Bijaya Ketan Panigrahi
Swagatam Das
Ponnuthurai Nagaratnam Suganthan
Pradipta Kumar Nanda (Eds.)

LNCS 7677

Swarm, Evolutionary, and Memetic Computing

Third International Conference, SEMCCO 2012
Bhubaneswar, India, December 2012
Proceedings

Springer

Lecture Notes in Computer Science 7677

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Bijaya Ketan Panigrahi
Swagatam Das
Ponnuthurai Nagaratnam Suganthan
Pradipta Kumar Nanda (Eds.)

Swarm, Evolutionary, and Memetic Computing

Third International Conference, SEMCCO 2012
Bhubaneswar, India, December 20-22, 2012
Proceedings

Springer

Volume Editors

Bijaya Ketan Panigrahi
Indian Institute of Technology
Department of Electrical Engineering
New Delhi 110016 , India
E-mail: bkpanigrahi@ee.iitd.ac.in

Swagatam Das
Indian Statistical Institute
Electronics and Communication Sciences Unit
Kolkata 700108, India
E-mail: swagatam.das@ieee.org

Ponnuthurai Nagaratnam Suganthan
Nanyang Technological University
School of Electrical and Electronic Engineering
Singapore 639798, Singapore
E-mail: epnsugan@ntu.edu.sg

Pradipta Kumar Nanda
Siksha "0" Anusandhan University
Department of Electronics and Telecom. Engineering
Institute of Technical Education & Research
Odisha 751030, India
E-mail: pknanda@iter.ac.in

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-35379-6 e-ISBN 978-3-642-35380-2
DOI 10.1007/978-3-642-35380-2
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012952995

CR Subject Classification (1998): F.1, I.2, H.3, F.2, I.4-5, J.3

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This LNCS volume contains the papers presented at the Third Swarm, Evolutionary and Memetic Computing Conference (SEMCCO 2012) held during December 20–22, 2012, at the Institute of Technical Education and Research, Siksha 'O' Anusandhan University, Bhubaneswar, Odisha, India. SEMCCO is regarded as one of the most prestigious international conference series, aiming to bring together researchers from academia and industry to report and review the latest progress in cutting-edge research with swarm, evolutionary, memetic computing, and other novel computing techniques such as neural and fuzzy computing, and thereby to explore new application areas, to design new bio-inspired algorithms for solving specific hard optimization problems, and finally to create awareness of these domains in a wider audience of practitioners.

SEMCCO 2012 received 310 paper submissions from 25 countries across the globe. After a rigorous peer-review process involving 990 reviews, 96 full-length articles were accepted for oral presentation at the conference. This corresponds to an acceptance rate of 31% and is intended for maintaining the high standards of the conference proceedings. The papers included in this LNCS volume cover a wide range of topics in swarm, evolutionary, memetic, and other intelligent computing algorithms and their real-world applications in problems selected from diverse domains of science and engineering.

The conference featured four distinguished keynote speakers. The lectures included Xin Yao's talk on "Cooperatively Coevolving Particle Swarms for Large Scale Optimization," Hisao Ishibuchi's talk on "Parallel Distributed Fuzzy Genetics-Based Machine Learning," Kumar Venayagamoorthy's talk on "Computational Intelligence Applications in Smart Grids," and Kalyanmoy Deb's lecture on "Multiobjective Evolutionary Algorithms."

We take this opportunity to thank the authors of all submitted papers for their hard work, adherence to the deadlines, and patience with the review process. The quality of a refereed volume depends mainly on the expertise and dedication of the reviewers. We are indebted to the Program Committee and Technical Committee members, who not only produced excellent reviews but also did so in the short timeframes they were given.

We would also like to thank our sponsors for providing all the logistical support and financial assistance. First, we are indebted to ITER Management and Administrations (faculty colleagues and administrative personnel of the School of Computer Science, School of Electronics Engineering, and School of Electrical Engineering) for supporting our cause and encouraging us to organize the conference at ITER, SOA University, Bhubaneswar, Odisha. In particular, we would like to express our heart-felt thanks to Manojranjan Nayak, President, Siksha O. Anusandhan Trust, for providing us with the necessary financial support and infrastructural assistance to hold the conference. Our sincere thanks are due to

R.P. Mohanty, Vice Chancellor, SOA University, R.K. Mishra, Dean, ITER, and P.K. Dash, Director, Research, for their continuous support. We thank Carlos A. Coello Coello and Nikhil R. Pal, the General Chairs, for providing valuable guidelines and inspiration to overcome various difficulties in the process of organizing this conference. We deeply regret the sad and untimely demise of one of our beloved colleagues – Satish Kumar of the Dayal Bag Educational Institute, Agra, India. Professor Kumar had agreed to deliver a keynote lecture on neuro-fuzzy systems at this conference. He is known to many of us through his marvelous book on *Neural Networks* published by Tata McGraw Hill. Apart from a great scientist and a prolific writer, he was a wonderful human being. A session at the conference was dedicated to the memory of Prof. Kumar.

We would also like to thank the participants of this conference, who considered the conference above all hardships. Finally, we would like to thank all the volunteers whose tireless efforts in meeting the deadlines and arranging every detail ensured that the conference ran smoothly. We hope the readers of this proceedings volume find the papers inspiring and enjoyable.

December 2012

Bijaya Ketan Panigrahi
Swagatam Das
P.N. Suganthan
P.K. Nanda

Organization

Chief Patron

Manojranjan Nayak, India

Patron

R.P. Mohanty, India

Honorary Chair

P.K. Dash

General Chairs

Nikhil R. Pal, India
Carlos A. Coello Coello, Mexico

General Co-chairs

Swagatam Das, India
B.K. Panigrahi, India

Program Chairs

R.K. Mishra, India
P.K. Nanda, India

Finance Chair

Manas Kumar Mallick, India

Steering Committee Chair

P.N. Suganthan, Singapore

Publicity Chairs

S.S. Dash, India
S.C. Satpathy, India
N.C. Sahoo, Malaysia

Special Session Chairs

Sanjoy Das, USA
Zhihua Cui, China
Samuelson Hong, Taiwan

Tutorial Chair

G. Panda, India

Organizing Secretariat

Debahuti Mishra
A.K. Jagadev, India

International Advisory Committee / Technical Review Committee

Almoataz Youssef Abdelaziz, Egypt
Athanasios V. Vasilakos, Athens
Alex K. Qin, France
Amit Konar, India
Anupam Shukla, India
Ashish Anand, India
Boyang Qu, China
Carlos A. Coello Coello, Mexico
Chilukuri K. Mohan, USA
Delin Luo, China
Dipankar Dasgupta, USA
D.K. Chaturvedi, India
Dipti Srinivasan, Singapore
Fatih M. Tasgetiren, Turkey
Ferrante Neri, Finland
Frank Neumann, Australia
Fayzur Rahman, Portugal
G.K. Venayagamoorthy, USA
Gerardo Beni, USA
Hai Bin Duan, China
Heitor Silvério Lopes, Brazil
Halina Kwasnicka, Poland
Hong Yan, Hong Kong
Javier Del Ser, Spain
Jane J. Liang, China
Janez Brest, Slovenia
Jeng-Shyang Pan, Taiwan
Juan Luis Fernández Martínez, Spain
Jeng-Shyang Pan, Taiwan
Kalyanmoy Deb, India
K. Parsopoulos, Greece
Kay Chen Tan, Singapore
Ke Tang, China
K. Shanti Swarup, India
Lakhmi Jain, Australia
Leandro Dos Santos Coelho, Brazil
Ling Wang, China
Lingfeng Wang, China
M.A. Abido, Saudi Arabia
M.K. Tiwari, India
Maurice Clerc, France
Meng Joo Er, Singapore
Meng-Hiot Lim, Singapore
M.F. Tasgetiren, Turkey
Namrata Khemka, USA
N. Puhan, India
Oscar Castillo, Mexico
Pei-Chann Chang, Taiwan
Peng Shi, UK
Qingfu Zhang, UK
Quanke Pan, China
Rafael Stubs Parpinelli, Brazil

Rammohan Mallipeddi, Singapore
Roderich Gross, UK
Ruhul Sarker, Australia
Richa Sing, India
Robert Kozma, USA
Ravipudi Venkata Rao, India
Suresh Sundaram, Singapore
S. Baskar, India
S.K. Udgata, India
S.S. Dash, India
S.S. Pattanaik, India
S.G. Ponnambalam, Malaysia
Saeid Nahavandi, Australia
Saman Halgamuge, Australia
Shizheng Zhao, Singapore
Sachidananda Dehuri, Korea
Samuelson W. Hong, Taiwan
X.Z. Gao, Finland
Yew Soon Ong, Singapore
Ying Tan, China
Yucheng Dong , China

Organizing Committee

Niva Das
Guru Prasad Mishra
Renu Sharma
Bibhu Prasad Mohanty
Badri Narayan Sahoo
Kabri Das
Biswa Mohan Acharya
Priyabrata Pattnaik
Sharmista Kar
Tapas Kumar Mohapatra
Sajia Hassan
Anuja Nanda
Sandeep Kumar Satapathy
Ambika Prasad Mishra
Manas Kumar Nanda
Sikha Mishra
Nibedan Panda
Shruti Mishra
Soumendra Mohanty

Table of Contents

Investigation of Mutation Schemes in Real-Parameter Genetic Algorithms 1
Debayan Deb and Kalyanmoy Deb

Discrete Harmony Search Algorithm for Dynamic FJSSP in Remanufacturing Engineering 9
Kaizhou Z. Gao, Ponnuthurai Nagaratnam Suganthan, and Tay Jin Chua

Multilevel Image Thresholding Based on Tsallis Entropy and Differential Evolution 17
Soham Sarkar, Swagatam Das, and Sheli Sinha Chaudhuri

Convergence and Boundary Estimation of the Particle Dynamics in Generalized Particle Swarm Optimization 25
Dipankar Maity and Udit Halder

Application of Differential Evolution with Best of Random Mutation Strategy on Asymmetric Location Only Synthesis of Broadside Circular Antenna Array 33
Sudipta Das, Durbadal Mandal, Sakti Prasad Ghoshal, and Rajib Kar

Performance Evaluation of Bacterial Foraging Optimization Algorithm for the Early Diagnosis and Tracking of Alzheimer's Disease 41
Tinu Varghese, R. Sheela Kumari, P.S. Mathuranath, and N. Albert Singh

Multi-sensor Satellite Image Analysis Using Niche Genetic Algorithm for Flood Assessment 49
J. Senthilnath, P.B. Shreyas, Ritwik Rajendra, S.N. Omkar, V. Mani, and P.G. Diwakar

Solution to Economic Load Dispatch Problem Based on FIREFLY algorithm and Its Comparison with BFO, CBFO-S and CBFO-Hybrid 57
Ananthanaryanan Rathinam and Ripunjoy Phukan

MESFET DC Model Parameter Extraction Using Adaptive Accelerated Exploration Particle Swarm Optimizer 66
Layak Ali, Samrat L. Sabat, and Siba K. Udgata

Kernel Group Method of Data Handling: Application to Regression Problems 74
Kalam Narendar Reddy and Vadlamani Ravi

An Efficient Algorithm for Gray Level Image Enhancement Using Cuckoo Search 82
Sanjay Agrawal and Rutuparna Panda

A Multi-objective Workspace Optimization of 3R Manipulator Using Modified PSO 90
Sumanta Panda, Debadutta Mishra, and B.B. Biswal

An Analysis of Genetic Algorithm Based Anycast Routing in Delay and Disruption Tolerant Networks 98
Éderson R. Silva and Paulo R. Guardieiro

Reactive Power Optimization Using Hybrid Cultural Algorithm 106
Bidishna Bhattacharya, Kamal Krishna Mandal, and Niladri Chakraborty

Techno-Economic Feasibility Analysis of Hybrid Renewable Energy System Using Improved Version of Particle Swarm Optimization 116
Bhimsen Tudu, Preetam Roy, Sajjan Kumar, Diptendu Pal, Kamal Krishna Mandal, and Niladri Chakraborty

Software Effort Prediction Using Fuzzy Clustering and Functional Link Artificial Neural Networks 124
Tirimula Rao Benala, Rajib Mall, Satchidananda Dehuri, and V.L. Prasanthi

Optimal Placement and Sizing of Distributed Generation in Radial Distribution System Using Differential Evolution Algorithm 133
Manas R. Nayak, Subrat K. Dash, and Pravat Kumar Rout

Clustering Algorithm Recommendation: A Meta-learning Approach 143
Daniel G. Ferrari and Leandro Nunes de Castro

A Clustering Particle Based Artificial Bee Colony Algorithm for Dynamic Environment 151
Subhodip Biswas, Digbalay Bose, and Souvik Kundu

A Selective Teaching-Learning Based Niching Technique with Local Diversification Strategy 160
Souvik Kundu, Subhodip Biswas, Swagatam Das, and Digbalay Bose

Efficient Dynamic Routing on Large Road Networks 169
Mohanasuram Geetha, G.M. Kadhar Nawaz Gulammohien, and S. Saravanan

Performance of Informative Differential Evolution Algorithm with Self Adaptive Re-clustering Technique on the Problems of Electromagnetism –The Linear Array Antenna Synthesis 181
Dipankar Maity, Udit Halder, and Sheli Sinha Chaudhuri

Differential Evolution with a Relational Neighbourhood-Based Strategy for Numerical Optimization 189
Souvik Kundu, Digbalay Bose, and Subhodip Biswas

A Simulated Annealing Heuristic for Minimizing Makespan in Parallel Machine Scheduling 198
Dipak Laha

Rule Extraction from DEWNN to Solve Classification and Regression Problems 206
Nekuri Naveen, Vadlamani Ravi, and Chillarige Raghavendra Rao

A New Improved Self Adaptive Particle Swarm Optimization Technique for Economic Load Dispatch 215
Kamal Krishna Mandal, Bidishna Bhattacharya, Bhimsen Tudu, and Niladri Chakraborty

Memetic Algorithm Based Task Scheduling Using Probabilistic Local Search 224
S. Padmavathi, S. MohitGolchha, and A. SeeniMohamed

Neighborhood Search Based Artificial Bee Colony Algorithm for Numerical Function Optimization 232
Anguluri Rajasekhar, Swagatam Das, Bijaya Ketan Panigrahi, and Manas Kumar Mallick

Path Generation and Obstacle Avoidance of an Autonomous Mobile Robot Using Intelligent Hybrid Controller 240
Prases Kumar Mohanty and Dayal R. Parhi

An Adaptive Memetic Algorithm for Multi-robot Path-Planning 248
Pratyusha Rakshit, Dhrubojyoti Banerjee, Amit Konar, and Ramadoss Janarthanan

Power Loss Minimization by the Placement of DG in Distribution System Using GA 259
Dasarathan Sattianadan, M. Sudhakaran, Subhransu Sekhar Dash, and K. Vijayakumar

Multipopulation Based Differential Evolution with Self Exploitation Strategy 267
Rupam Kundu, Rohan Mukherjee, and Shantanab Debchoudhury

Clustered Parent Centric Normal Cross-Over for Multimodal Optimization 276
Rohan Mukherjee, Rupam Kundu, and Swagatam Das

Stock Indices Prediction Using Radial Basis Function Neural Network 285
Minakhi Rout, Babita Majhi, Usha Manasi Mohapatra, and Rosalin Mahapatra

Gene Subset Selection for Cancer Classification Using Statsitical and Rough Set Approach 294
Asit Kumar Das and Soumen Kumar Pati

Support Vector Machine for Large Databases as Classifier 303
Rahul Kumar Sevakula and Nishchal K. Verma

Solution to Economic Load Dispatch Problem Based on BFO, CBFO-S and CBFO-H Algorithms and Its Advantages over the PSO 314
Ananthanaryanan Rathinam and Ripunjoy Phukan

Rough Set Based Fuzzy K-Modes for Categorical Data 323
Indrajit Saha, Jnanendra Prasad Sarkar, and Ujjwal Maulik

Teaching Learning Opposition Based Optimization for the Location of Median Line in 3-D Space 331
Anguluri Rajasekhar and Swagatam Das

Modified Onlooker Phase in Artificial Bee Colony Algorithm 339
Tarun Kumar Sharma, Millie Pant, and V.P. Singh

Complex-Valued Neuro-Fuzzy Inference System Based Classifier 348
Kartick Subramanian, Ramaswamy Savitha, Sundaram Suresh, and B.S. Mahanand

Neural Network Based Model Predictive Controller for Simplified Heave Model of an Unmanned Helicopter 356
Mahendra Kumar Samal, Sreenatha Anavatti, and Matthew Garratt

Hybrid Biogeography Based Simultaneous Feature Selection and Prediction of N-Myristoylation Substrate Proteins Using Support Vector Machines and Random Forest Classifiers 364
Shameek Ghosh, Nayana Ramachandran, C. Venkateshwari, and V.K. Jayaraman

Plant Leaf Disease Detection Using Gabor Wavelet Transform 372
Shitala Prasad, Piyush Kumar, Ranjay Hazra, and Ajay Kumar

Analysis of Vasculature in Human Retinal Images Using Particle Swarm Optimization Based Tsallis Multi-level Thresholding and Similarity Measures 380
Nadaradjane Sri Madhava Raja, Ganesan Kavitha, and Swaminathan Ramakrishnan

Optimal Reactive Power Compensation for Improvement of Steady State Voltage Stability Limit under Stressed System Condition Using BF Algorithm 388
Santi Behera and Manish Tripathy

Optimal Placement of Capacitors in Distribution Networks Using a Modified Teaching-Learning Based Algorithm 398
Ankita Mohapatra, Bijaya Ketan Panigrahi, Bhim Singh, and Ramesh Bansal

Gene Expression Programming Algorithm for Transient Security Classification 406
Almoataz Y. Abdelaziz, S.F. Mekhamer, H.M. Khattab, M.L.A. Badr, and Bijaya Ketan Panigrahi

PSO-Tuned Control Parameter in Differential Evolution Algorithm 417
Tapas Si, Nanda Dulal Jana, and Jaya Sil

On the Non Linear Dynamics of the Global Best Particle in Particle Swarm Optimization 425
Dipankar Maity, Udit Halder, Swagatam Das, and Bijaya Ketan Panigrahi

Adaptive Differential Evolution with Directional Information Based Search Moves 433
Satyajit Neogi, Deblina Das, and Swagatam Das

A Fuzzy Programming Method for Solving Multiobjective Chance Constrained Programming Problems Involving Log-Normally Distributed Fuzzy Random Variables 442
Animesh Biswas and Arnab Kumar De

Minimization of Side Lobe of Optimized Uniformly Spaced and Non-uniform Excited Time Modulated Linear Antenna Arrays Using Genetic Algorithm 451
Gopi Ram Hardel, Durbadal Mandal, Sakti Prasad Ghoshal, and Rajib Kar

Circular Antenna Array Design Using Novel Perturbation Based Artificial Bee Colony Algorithm 459
Digbalay Bose, Souvik Kundu, Subhodip Biswas, and Swagatam Das

Cooperative Co-evolutionary Teaching-Learning Based Algorithm with a Modified Exploration Strategy for Large Scale Global Optimization ... 467
Subhodip Biswas, Souvik Kundu, Digbalay Bose, and Swagatam Das

Brain Storming Incorporated Teaching–Learning–Based Algorithm with Application to Electric Power Dispatch ... 476
K.R. Ramanand, K.R. Krishnanand, Bijaya Ketan Panigrahi, and Manas Kumar Mallick

Associated and Assorted Recombination in SBX Operator for Problems with Linkages ... 484
Arnav Acharyya and Kalyanmoy Deb

Scalable Fuzzy Genetic Classifier Based on Fitness Approximation ... 492
Harihar Kalia, Satchidananda Dehuri, and Ashish Ghosh

Multi Objective Integrated Layout Design Problem ... 500
I. Jerin Leno, S. Saravana Sankar, and S.G. Ponnambalam

Dynamic Network Traffic Data Classification for Intrusion Detection Using Genetic Algorithm ... 509
Rahul Mitra, Sahisnu Mazumder, Tuhin Sharma, Nandita Sengupta, and Jaya Sil

2DOF PID Controller Tuning for Unstable Systems Using Bacterial Foraging Algorithm ... 519
K. Latha and V. Rajinikanth

Generation of Sufficient Cut Points to Discretize Network Traffic Data Sets ... 528
Sahisnu Mazumder, Tuhin Sharma, Rahul Mitra, Nandita Sengupta, and Jaya Sil

Parameters Optimization of Continuous Casting Process Using Teaching-Learning-Based Optimization Algorithm ... 540
Ravipudi Venkata Rao and Vivek D. Kalyankar

Genetic Algorithm Based Approach for Optimal Allocation of TCSC for Power System Loadability Enhancement ... 548
Almoataz Y. Abdelaziz, M.A. El-Sharkawy, M.A. Attia, and Bijaya Ketan Panigrahi

Wavelet-ANN Model for Nutrient Load Predictions in Rivers ... 558
Raj Mohan Singh

Performance Analysis and Design of Proportional Integral Derivative Controlled Automatic Voltage Regulator System Using Local Unimodal Sampling Optimization Technique ... 566
Pradeep Kumar Mohanty, Binod Kumar Sahu, Sidhartha Panda, Sanjeeb Kumar Kar, and Nandan Mishra

Particle Swarm Optimization Trained Auto Associative Neural Networks Used as Single Class Classifier 577
Vadlamani Ravi, Naveen Nekuri, and Manideepto Das

A Network Theoretic Analysis of Evolutionary Algorithms 585
Karthik Kuber, Stuart W. Card, Kishan G. Mehrotra, and Chilukuri K. Mohan

Characterization of Trabecular Architecture in Human Femur Radiographic Images Using Directional Multiresolution Transform and AdaBoost Model 594
Thomas Christy Bobby and Swaminathan Ramakrishnan

Missing Value Estimation of Microarray Data Using Similarity Measurement 602
Soumen Kumar Pati and Asit Kumar Das

A Strategy Pool Adaptive Artificial Bee Colony Algorithm for Dynamic Environment through Multi-population Approach 611
Digbalay Bose, Subhodip Biswas, Souvik Kundu, and Swagatam Das

Multistage Covariance Matrix Adaptation with Differential Evolution for Constrained Optimization 620
Shantanab Debchoudhury, Rohan Mukherjee, and Rupam Kundu

Evolutionary and Immune Algorithms Applied to Association Rule Mining 628
Danilo Souza da Cunha and Leandro Nunes de Castro

Improving Adaptive Differential Evolution with Controlled Mutation Strategy 636
Sayan Basu Roy, Mainak Dan, and Pallavi Mitra

Modified Particle Swarm Optimization with Switching Update Strategy 644
Rupam Kundu, Rohan Mukherjee, and Swagatam Das

Statistical Analysis Based Adjustment Method for Convergence Rate of Spiral Optimization 653
Kenichi Tamura and Keiichiro Yasuda

Neglect Benevolence in Human-Swarm Interaction with Communication Latency 662
Phillip Walker, Steven Nunnally, Mike Lewis, Andreas Kolling, Nilanjan Chakraborty, and Katia Sycara

A Comment on Bio-inspired Optimisation via GPU Architecture: The Genetic Algorithm Workload 670
Paula Prata, Paulo Fazendeiro, Pedro Sequeira, and Chandrashekhar Padole

Optimal Location and Sizing of DG for Congestion Management in Deregulated Power Systems 679
K. Vijayakumar and R. Jegatheesan

A Novel Strategy Adaptive Genetic Algorithm with Greedy Local Search for the Permutation Flowshop Scheduling Problem 687
Srinjoy Ganguly, Swahum Mukherjee, Debabrota Basu, and Swagatam Das

Estimation of Autocorrelation Space for Classification of Bio-medical Signals 697
Mihir Narayan Mohanty and Aurobinda Routray

Dimension Reduction Using Clustering Algorithm and Rough Set Theory 705
Shampa Sengupta and Asit Kumar Das

Connectivity Differences between Human Operators of Swarms and Bandwidth Limitations 713
Steven Nunnally, Phillip Walker, Michael Lewis, Andreas Kolling, Nilanjan Chakraborty, and Katia Sycara

Analysis of Emergent Selection Pressure in Evolutionary Algorithm and Machine Learner Offspring Filtering Hybrids 721
Mark Coletti and Guido Cervone

OpenCL Implementations of a Genetic Algorithm for Feature Selection in Periocular Biometric Recognition 729
Paulo Fazendeiro, Chandrashekhar Padole, Pedro Sequeira, and Paula Prata

Face Authentication Using Supervised Learning Techniques 738
Amioy Kumar, Rohan Gupta, Akshay Sharma, Bijaya Ketan Panigrahi, and M. Hanmandlu

Design of Non-uniform Circular Antenna Arrays Using Coordinated Bacterial Dynamics and Opposite Numbers 746
Jaydeep Ghosh Chowdhury, Aritra Chowdhury, Arghya Sur, and Swagatam Das

Process Plan and Scheduling Integration for Near Optimal Process Plans in Networked Based Manufacturing Using Controlled Elitist NSGA-II and Territory Defining Algorithms 754
V.K. Manupati, J.J. Thakkar, Priyabrata Mohapatra, Ajay Kumar, and M.K. Tiwari

Teaching Learning Based Optimization for Neural Networks Learning Enhancement 761
Suresh Chandra Satapathy, Anima Naik, and K. Parvathi

Large Scale Optimization Based on Co-ordinated Bacterial Dynamics and Opposite Numbers 770
Jaydeep Ghosh Chowdhury, Aritra Chowdhury, and Arghya Sur

A Strategy Adaptive Genetic Algorithm for Solving the Travelling Salesman Problem 778
Swahum Mukherjee, Srinjoy Ganguly, and Swagatam Das

Efficient Design of Cosine-Modulated Filter Banks Using Evolutionary Multi-objective Optimization 785
Md. Nasir, Soumyadip Sengupta, and Swagatam Das

Voltage Stability Constrained Optimal Power Flow Using Non-dominated Sorting Genetic Algorithm-II (NSGA II) 793
C. Nithya, J. Preetha Roselyn, D. Devaraj, and Subhransu Sekhar Dash

Markov Random Field Model Based Graduated Penalty Function for Reinforcing Ill-Defined Edges in Color Image Segmentation 802
Panda Sucheta and P.K. Nanda

Author Index 811

Investigation of Mutation Schemes in Real-Parameter Genetic Algorithms

Debayan Deb[1] and Kalyanmoy Deb[2]

[1] Department of Computer Science, Michigan State University,
East Lansing, MI 48824, USA
ronny3050@gmail.com
[2] Department of Mechanical Engineering, Indian Institute of Technology Kanpur,
208016, India
deb@iitk.ac.in

Abstract. In this paper, we investigate the effect of five different mutation schemes for real-parameter genetic algorithms (RGAs). Based on extensive simulation studies, it is observed that a mutation clock implementation is computationally quick and also efficient in finding a solution close to the optimum on four different problems used in this study. Moreover, parametric studies on the polynomial mutation operator identify a working range of values of these parameters. This study signifies that the long-suggested mutation clock operator should be considered as a valuable mutation operator for RGAs.

1 Introduction

Real-parameter evolutionary optimization has received a lot of attention over the years [1–4]. In a real-parameter genetic algorithm (RGA), the mutation operator is used primarily as a mechanism for maintaining diversity in the population [5, 6]. In contrast to a recombination operator, a mutation operator operates on only one individual at a time. In RGAs, several mutation operators are suggested – random mutation [7], Gaussian mutation [8], polynomial mutation [9, 10], and others. The effect is to perturb the current variable value to a neighboring value. While operated on multi-variable individual, one common strategy is to mutate each variable with a pre-specified probability.

Despite the existence of different mutation operators for perturbing a variable, the procedure of applying mutation operator to GA population members can be applied in many different ways. The effect of different mutation schemes on the performance of a real-parameter GA (RGA) is not investigated adequately yet. In this paper, we suggest and compare five different mutation schemes based on their effects on four different standard test problems. In all cases, we use the polynomial mutation for perturbing a variable. Similar studies can also be performed with other mutation operators.

In the remainder of the paper, we briefly describe the polynomial mutation operator. Thereafter, we present five different mutation schemes suggested here.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 1–8, 2012.

Simulation results of a real-parameter GA with identical selection and recombination operators but different mutation schemes are compared against each other and against no mutation. Interesting observations are made. Finally, the effect of polynomial mutation strength and probability are investigated for the winning mutation scheme. Conclusions of the study are then made.

2 Polynomial Mutation in Real-Parameter GAs

Deb and Agrawal [9] suggested a polynomial mutation operator with a user-defined index parameter (η_m). Based on a theoretical study, they concluded that η_m induces an effect of a perturbation of $O((b-a)/\eta_m)$ in a variable, where a and b are lower and upper bounds of the variable. They also found that a value $\eta_m \in [20, 100]$ is adequate in most problems that they tried. In this operator, a polynomial probability distribution is used to perturb a solution in a parent's vicinity. The probability distribution in both left and right of a variable value is adjusted so that no value outside the specified range $[a, b]$ is created by the mutation operator. For a given parent solution $p \in [a, b]$, the mutated solution p' for a particular variable is created for a random number u created within $[0, 1]$, as follows:

$$p' = \begin{cases} p + \bar{\delta}_L(p - x_i^{(L)}), \text{ for } u \leq 0.5, \\ p + \bar{\delta}_R(x_i^{(U)} - p), \text{ for } u > 0.5. \end{cases} \quad (1)$$

Then, either of the two parameters ($\bar{\delta}_L$ or $\bar{\delta}_R$) is calculated, as follows:

$$\bar{\delta}_L = (2u)^{1/(1+\eta_m)} - 1, \quad \text{for } u \leq 0.5, \quad (2)$$

$$\bar{\delta}_R = 1 - (2(1-u))^{1/(1+\eta_m)}, \quad \text{for } u > 0.5. \quad (3)$$

To illustrate, Figure 1 shows the probability density of creating a mutated child point from a parent point $p = 3.0$ in a bounded range of [1,8] with $\eta_m = 20$.

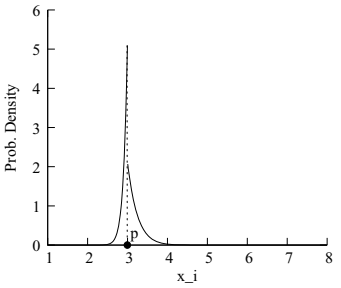


Fig. 1. Probability density function of creating a mutated child solution

3 Five Mutation Schemes

We discuss five different mutation schemes used in this study. We also compare all five schemes with Scheme 0 in which no mutation operator is used.

Scheme 1: Usual Mutation
This mutation scheme follows the usual method of mutating each and every variable one at a time with a pre-defined mutation probability p_m [5, 10]. Usually, $p_m = 1/n$ is used, so that on an average, one variable gets mutated per individual. A random number $u \in [0, 1]$ is created for every variable for an individual and if $u \leq p_m$ the variable is mutated using the polynomial mutation operator.

Scheme 2: Mutation Clock
The usual mutation scheme described above requires n random numbers to be created per individual. To reduce the computational complexity, Goldberg [5] suggested a *mutation clock* scheme for binary-coded GAs, in which once a bit is mutated, the next bit to be mutated (in the same or in a different individual) is determined by using an exponential probability distribution: $p(t) = \lambda \exp(-\lambda t)$, where λ is the inverse of the average occurrence of mutations. We implement here mutation clock, for the first time, to real-parameter GAs. With a mutation probability of p_m, on an average, one mutation will occur in $1/p_m$ variables. Thus, $\lambda = p_m$. For a mutation event, a random number $u \in [0, 1]$ is first chosen. Then, equating $u = \int_{t=0}^{l} p_m \exp(-p_m t)dt$, we obtain the next occurrence (l) of mutation as:

$$l = \frac{1}{p_m} \log(1 - u). \tag{4}$$

If k-th variable on i-th individual in the population is currently mutated, the next variable to be mutated is $((k + l) \bmod n)$-th variable of the $((k + l)/n)$-th individual from current individual in the population. At every generation, the initial variable to be mutated is calculated using $i = k = 1$. This operator should, on an average, require n times less number of random numbers than that required in Scheme 1.

Scheme 3: One Mutation per Solution
Here, we choose a random variable $i \in [1, n]$ and x_i is mutated using the polynomial mutation. Exactly, one mutation is performed per individual.

Scheme 4: Fixed Mutation
This mutation scheme is similar to Scheme 3, except that every variable is given an equal chance, thereby implementing a less noisy implementation of Scheme 3. For this purpose, variables are ordered in a random order in every generation and then variables are mutated using the polynomial mutation according to the sorted order of variables from first to last population member. After n variables are mutated in n top population members, the same order is followed from $(n+1)$-th population member. Again, exactly one mutation is performed per individual.

Scheme 5: Diversity based Mutation
In this mutation scheme, we put more probability for mutation to a variable that has less population-wise diversity. To implement, first the variance of values of each variable across the population members is computed and variables are sorted in ascending order of variance. Thereafter, an exponential probability distribution ($p(i) = \lambda \exp(-\lambda i)$ for $i \in [0, n - 1]$ is used. To make the above a probability distribution, λ is used by finding the root of the following equation for a fixed n: $\lambda \exp(-n\lambda) - \exp(-\lambda) - \lambda + 1 = 0$. Thereafter, for a random number $u \in [0, 1]$, the variable $(l + 1)$ that should be mutated is given:

$$l = \frac{1}{\bar{\lambda}} \log\left(1 - u(1 - \exp(-n\bar{\lambda}))\right). \tag{5}$$

For $n = 15$, $\bar{\lambda} = 0.168$ is found. This scheme makes one mutation per individual.

4 Results

We consider four different problems to investigate the effect of five suggested mutation schemes. The objective functions have minimum at $x_i = 0$ $(i = 1, \ldots, n)$ for the first three problems, and $x_i = 1$ $(i = 1, \ldots, n)$ for the fourth problem. However, in each case, we consider 15 variables $(n = 15)$, initialized and bounded within $x_i \in [-5, 10]$. This does not make the optimum exactly at the middle of the search space. We use the following GA parameter settings: (i) No. of real variables, $n = 15$, (ii) Population size = 150, (iii) SBX operator probability, $p_c = 0.9$, (iv) SBX operator index, $\eta_c = 2$, (v) Polynomial mutation operator probability, $p_m = 1/\text{n}$, (vi) Polynomial mutation operator index, $\eta_m = 20$, and (vii) Termination parameter $\epsilon_T = 0.01$. For each mutation scheme, we make 51 different runs starting from different initial populations, however the same set of initial populations are used for all mutation schemes. Mutation schemes are then compared with each other and with the zero mutation scheme.

4.1 Ellipsoidal Function

This function is unimodal and additively separable having $f_{ell}(\mathbf{x}^*) = 0$: $f_{ell}(x) = \sum_{i=1}^{D} i x_i^2$. The performance of different mutation schemes on the problem is shown in Table 1. While all mutation schemes perform successfully on 100% runs, GAs without mutation have failed to find the required solution in more than 50% of the runs. This amply suggests the importance of using a mutation scheme in RGAs. Figure 2 shows how the objective value is reduced with generation for all five mutation schemes. Clearly, Schemes 1 and 2 perform the best.

4.2 Schwefel's Function

The function is as follows: $f_{sch}(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)$. The performance of different mutation schemes on this problem is shown in Table 2. Figure 3 shows the

Table 1. Performance of different mutation schemes on ellipsoidal problem

		Scheme 0	Scheme 1	Scheme 2	Scheme 3	Scheme 4	Scheme 5
Run time	min.	**0.05**	0.07	0.06	0.11	0.10	0.15
	avg.	2.96	**0.08**	**0.08**	0.15	0.14	0.20
	med.	3.56	0.09	**0.08**	0.15	0.15	0.19
	max.	4.49	0.10	**0.09**	0.20	0.20	0.25
Generations	min.	**122.00**	157.00	149.00	277.00	247.00	311.00
	avg.	7,701.12	190.27	**188.84**	385.00	370.11	410.60
	med.	10,000.00	193.00	**185.00**	389.00	375.00	408.00
	max.	10,000.00	232.00	**226.00**	500.00	502.00	539.00
Mutations	min.	0.00	15,737.00	**15,321.00**	27,700.00	24,700.00	31,100.00
	avg.	0.00	**19,113.27**	19,629.78	38,500.00	37,011.76	41,060.78
	med.	0.00	19,259.00	**19,196.00**	38,900.00	37,500.00	40,800.00
	max.	0.00	23,256.00	**23,232.00**	50,000.00	50,200.00	53,900.00

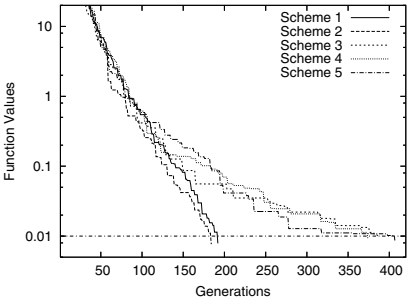


Fig. 2. Variation of number of function evaluations with generation for ellipsoidal problem

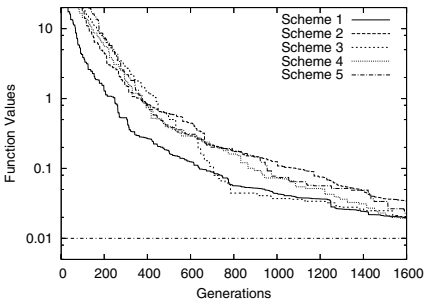


Fig. 3. Variation of number of function evaluations with generation for Schwefel's function

Table 2. Performance of different mutation schemes on Schwefel's problem

		Scheme 0	Scheme 1	Scheme 2	Scheme 3	Scheme 4	Scheme 5
Run time	min.	3.99	0.52	0.69	0.59	**0.47**	0.68
	avg.	4.72	1.27	1.26	**0.91**	0.94	1.04
	med.	4.96	1.22	1.23	**0.88**	0.94	1.03
	max.	5.38	2.60	1.84	1.85	**1.53**	1.66
Generations	min.	10,000.00	1,065.00	1,522.00	1,335.00	**1,060.00**	1,301.00
	avg.	10,000.00	2,559.57	2,788.29	2,053.10	2,146.77	**1,997.37**
	med.	10,000.00	2,490.00	2,718.00	1,979.00	2,122.00	**1,971.00**
	max.	10,000.00	5,307.00	4,076.00	4,210.00	3,470.00	**3,161.00**
Mutations	min.	0.00	106,891.00	159,104.00	133,500.00	**106,000.00**	130,100.00
	avg.	0.00	257,230.11	289,855.70	205,309.80	214,676.47	**199,737.25**
	med.	0.00	250,244.00	282,629.00	197,900.00	212,200.00	**197,100.00**
	max.	0.00	533,128.00	423,452.00	421,000.00	347,000.00	**316,100.00**

reduction in objective value with generation. All five schemes performs almost equally well for this problem.

4.3 Ackley's Function

Next, we consider Ackley's function: $f_{ack}(x) = 20+e-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right)$. The performance of different mutation schemes on this problem is shown in Table 3. Figure 4 shows the variation of objective value with generation. It is clear that Schemes 1 and 2 perform better than other three schemes, including the zero-mutation scheme.

4.4 Rosenbrock's Function

Rosenbrock's function is as follows: $f_{ros}(x) = \sum_{i=1}^{D-1}\left[100\left(x_{i+1}-x_i^2\right)^2 + (x_1-1)^2\right]$. Since this is a difficult problem to solve, we use $\epsilon_T = 15$. The performance of

Table 3. Performance of different mutation schemes on Ackley's function

		Scheme 0	Scheme 1	Scheme 2	Scheme 3	Scheme 4	Scheme 5
Run time	min.	**0.09**	0.11	0.10	0.53	0.41	0.58
	avg.	3.72	**0.14**	**0.14**	0.71	0.68	0.86
	med.	3.93	0.14	**0.13**	0.72	0.69	0.86
	max.	4.08	**0.18**	0.19	1.02	0.87	1.56
Generations	min.	**207.00**	224.00	225.00	1,220.00	969.00	1,132.00
	avg.	9,427.43	**291.88**	303.31	1,641.90	1,563.51	1,682.78
	med.	10,000.00	**291.00**	296.00	1,652.00	1,598.00	1681.00
	max.	10,000.00	**372.00**	429.00	2,364.00	2,031.00	3,066.00
Mutations	min.	0.00	**22,392.00**	23,523.00	122,000.00	96,900.00	113,200.00
	avg.	0.00	**29,349.04**	31,527.67	164,190.19	156,350.98	168,278.43
	med.	0.00	**29,068.00**	30,891.00	165,200.00	159800.00	168,100.00
	max.	0.00	**37,258.00**	44,327.00	236,400.00	203,100.00	306,600.00

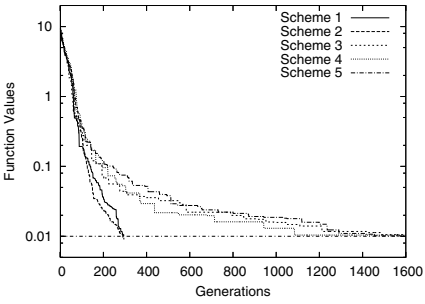


Fig. 4. Variation of f with generation for Ackley's function

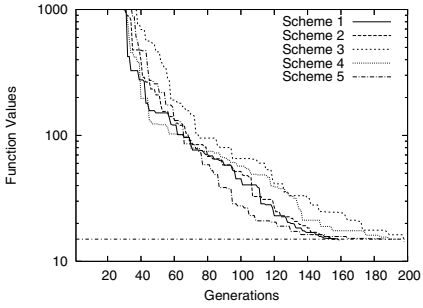


Fig. 5. Variation of f with generation for Rosenbrock's function

different mutation schemes on this problem is shown in Table 4. Figure 5 indicates that Schemes 1 and 2 perform slightly better.

On four different problems, it is observed that in terms of function evaluations, Schemes 1 and 2 perform better than the other three mutation schemes including the zero mutation scheme. Tables show that Scheme 2 requires much smaller number of computational time compared to Scheme 1. Hence, we recommend the use of mutation clock as an efficient mutation scheme for real-parameter GAs.

5 Parametric Studies with Mutation Index and Mutation Probability

Having established the superiority of mutation clock, we now perform a parametric study of the distribution index η_m for Scheme 2. Figure 6 shows that $\eta_m \approx 100$ performs the best on the ellipsoidal problem. Figure 7 shows $\eta_m \in [100, 150]$ produce best results for Schwefel's function. Similar plots for Ackley's and Rosenbrock's functions are found but for brevity we do present them here.

Table 4. Performance of different mutation schemes on Rosenbrock's problem

		Scheme 0	Scheme 1	Scheme 2	Scheme 3	Scheme 4	Scheme 5
Run time	min.	0.08	**0.05**	**0.05**	**0.05**	**0.05**	0.06
	avg.	4.63	0.24	**0.14**	0.29	0.27	0.30
	med.	5.14	0.08	**0.07**	0.09	0.08	0.10
	max.	5.51	3.09	**0.99**	2.45	2.77	2.21
Generations	min.	165.00	**102.00**	106.00	110.00	126.00	110.00
	avg.	9433.04	473.75	**309.12**	632.88	598.29	561.21
	med.	10,000.00	160.00	**159.00**	199.00	196.00	188.00
	max.	10,000.00	6,284.00	**2,163.00**	5,481.00	6,227.00	4,207.00
Mutations	min.	0.00	**10,148.00**	11,034.00	11,000.00	12,600.00	11,000.00
	avg.	0.00	47,593.15	**32,159.07**	63,288.23	59,829.41	56,121.57
	med.	0.00	**15,827.00**	16,544.0	19,900.00	19,600.00	18,800.00
	max.	0.00	631,346.00	**225,044.00**	548,100.00	622,700.00	420,700.00

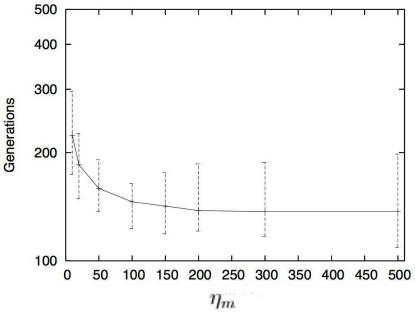


Fig. 6. Parametric study of η_m for Scheme 2 on ellipsoidal function

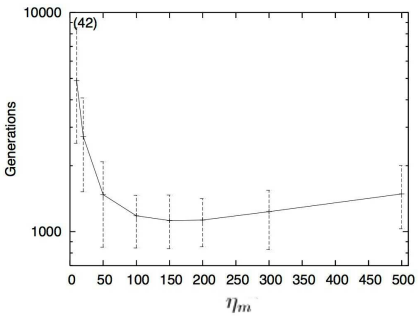


Fig. 7. Parametric study of η_m for Scheme 2 on Schwefel's function

Next, we perform a parametric study of mutation probability p_m with Scheme 2 and fix the mutation index as $\eta_m = 100$ (which is found to be near-optimal above). We use $p_m = k/n$ with $k = 0.01, 0.1, 0.5, 0.75, 1, 1.5, 2$, and 5. The rest of the GA parameters are kept the same as before. Figure 8 shows the number of generations needed for 51 runs for the ellipsoidal problem. Mutation probabilities of $0.01/n$ and $0.1/n$ are not found to produce a reasonable result. The figure shows that $0.5/n$ to $1.5/n$ perform the best. Thus, the usual practice of $p_m = 1/n$ is justified by our study. Similar results are found for other two functions as well. Thus, based on the above simulation results, we suggest the following parameter values for the polynomial mutation operator which is to be applied with SBX recombination operator with $\eta_c = 2$ and the binary tournament selection operator:

Mutation scheme: = Mutation clock [5],
Mutation index, η_m [9]: = $[100, 150]$,
Mutation probability, p_m:= $[0.5/n, 1.5/n]$, where n is the number of variables.

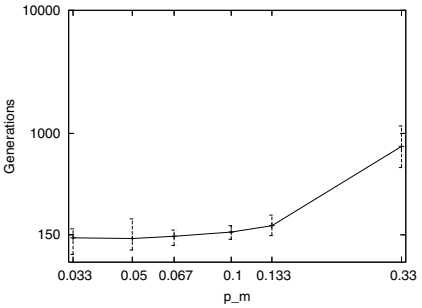


Fig. 8. Parametric study of p_m for Scheme 2 on the ellipsoidal function

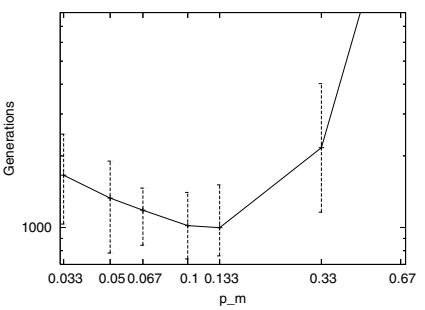


Fig. 9. Parametric study of p_m for Scheme 2 on Schwefel's function

6 Conclusions

Mutation operators are used for maintaining diversity in a GA. In this paper, we have suggested five different mutation schemes for real-parameter GAs. Based on their performance, we conclude the following: (i) any of the five mutation operators is better than not performing any mutation, (ii) the mutation clock operator which was suggested in eighties has been long forgotten and has been found here to be the fastest in terms of its execution time and best in terms of its performance. Parametric studies have also found a range of mutation index for the polynomial mutation ($\eta_m \in [100, 150]$) and mutation probability $p_m \in [0.5/n, 1.5/n]$ (where n is the number of variables) for its best performance. Similar studies can be performed with other real-parameter mutation operators.

References

1. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation Journal 9(2), 159–195 (2000)
2. Price, K.V., Storn, R., Lampinen, J.: Differential Evolution: A Practical Approach to Global Optimization. Springer (2005)
3. Kennedy, J., Eberhart, R.C., Shi, Y.: Swarm intelligence. Morgan Kaufmann (2001)
4. Deb, K., Agrawal, R.B.: Simulated binary crossover for continuous search space. Complex Systems 9(2), 115–148 (1995)
5. Goldberg, D.: Genetic algorthims in search, optimization, and machine learning. Addison-Wesley, Reading (1989)
6. Holland, J.H.: Adaptation in Natural and Artificial Systems. MIT Press, Ann Arbor (1975)
7. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer, Berlin (1992)
8. Schwefel, H.-P.: Collective phenomena in evolutionary systems. In: Checkland, P., Kiss, I. (eds.) Problems of Constancy and Change – the Complementarity of Systems Approaches to Complexity, pp. 1025–1033. Intl. Soc. for General Sys. Res, Budapest (1987)
9. Deb, K., Agrawal, S.: A niched-penalty approach for constraint handling in genetic algorithms. In: Proceedings of the Intl. Conf. on ANN and GAs, pp. 235–243. Springer (1999)
10. Deb, K.: Multi-objective optimization using evolutionary algorithms. Wiley, Chichester (2001)

Discrete Harmony Search Algorithm for Dynamic FJSSP in Remanufacturing Engineering

Kaizhou Z. Gao[1,2], Ponnuthurai Nagaratnam Suganthan[1], and Tay Jin Chua[3]

[1] School of EEE, Nanyang Technology University, Singapore
[2] School of Computer, Liaocheng University, Liaocheng, P.R. China
[3] Singapore Institute of Manufacturing Technology, Singapore
EPNSugan@ntu.edu.sg

Abstract. Remanufacturing is a growing branch of the traditional manufacturing industry. In this study, a discrete harmony search (DHS) algorithm is proposed for the dynamic flexible job shop scheduling problem (FJSSP) in remanufacturing. Firstly, the dynamic flexible job shop scheduling in remanufacturing engineering is described. Secondly, the harmony search algorithm is discretized for the dynamic flexible job shop scheduling problem. Thirdly, a new method for improving a new harmony is proposed based on the characteristics of dynamic FJSSP and solution representation. Finally, simulation experiments are carried out to test the proposed discrete harmony search algorithm. The results show the effectiveness of the proposed DHS algorithm in solving the disassembly scheduling problem in remanufacturing engineering.

Keywords: remanufacturing engineering, harmony search, dynamic flexible job shop scheduling.

1 Introduction

Lund [1] was the first one to define and discuss the remanufacturing problem. Krupp [2,3] perfected the concept of remanufacturing and predicted the future of remanufacturing engineering. In recent years, many researchers have focused on the remanufacturing scheduling problem. Guide [4] analyzed the capacity planning in remanufacturing environment. Guide [5] also examined the impact of product structure complexity on other managerial and operational decisions in a remanufacturing environment. Depuy [6] proposed a methodology for production planning within facilities involved in the remanufacturing of products. Teunter [7] developed a polynomial-time dynamic programming heuristic for the cost in manufacturing and remanufacturing engineering. Grubbstrom [8] proposed a model to analyze how labor costs, material costs and budget influence the optimal production decisions in remanufacture engineering. Denizel [9] considered the production planning when inputs have different and uncertain quality levels and discussed different decision variables in remanufacturing engineering.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 9–16, 2012.

Harmony search is one of the latest meta-heuristic methods presented by Geem et al [10] for solving optimization problems. Compared to the earlier meta-heuristics, the HS algorithm imposes fewer mathematical requirements and numerical comparisons show that the process of evolution in harmony algorithm is faster than GA [11, 12].

During the past few years, the HS algorithm has been successfully employed to solve problems, such as structural optimization [11], vehicle routing [13], water network design [14], multiple dam operation [15], Sudoku puzzle [16], multi-pass face-milling [17], discrete-time chaotic system [18], blocking flow shop scheduling problem [19], lot-streaming flow shop scheduling problem [20], no-wait flow shop scheduling problem [21-22], diesel generators in oil rig platforms [23] and discrete design variables.

The rest of this paper is organized as follows: Section 2 introduces the dynamic flexible job shop scheduling problem in remanufacturing engineering; Section 3 gives the details of the DHS algorithm for the dynamic flexible job shop scheduling problem in remanufacturing engineering; Section 4 is the simulation computation and results. We conclude the paper in Section 5.

2 Problem Description

The process of a typical remanufacturing operation is shown in Fig.1 [4]. Disassembly is an important part in remanufacturing engineering. After receiving, the products must be disassembled into parts for the next processing stage. The disassembly step is regarded as a flexible job shop scheduling problem. Each product has a sequence of parts. A part in one product can be disassembled by one machine out of a set of candidate machines. Each part of a product must be disassembled only on one machine at a time while each machine can disassemble only one part at a time. It is difficult to obtain the disassembly or remanufacturing time for one received product because most products were not manufactured for disassembly or remanufacturing. For different received products, the disassembly or remanufacturing time is different and indeterminate. The first operation is to assign every part of every product to one machine for disassembly and sequence the parts on each machine for minimum makespan. In remanufacturing process, if the processing time changes and makespan is larger, the subsequent operations may have to be rescheduled.

Makespan, denoted by C_M, is the maximal of completion time of products. The objectives of this paper are as Eq.1:

$$\text{Min } C_M = \max_{1 \le i \le n} \{c_i\} \tag{1}$$

where, c_i is the completion time of product i and n is the number of products.

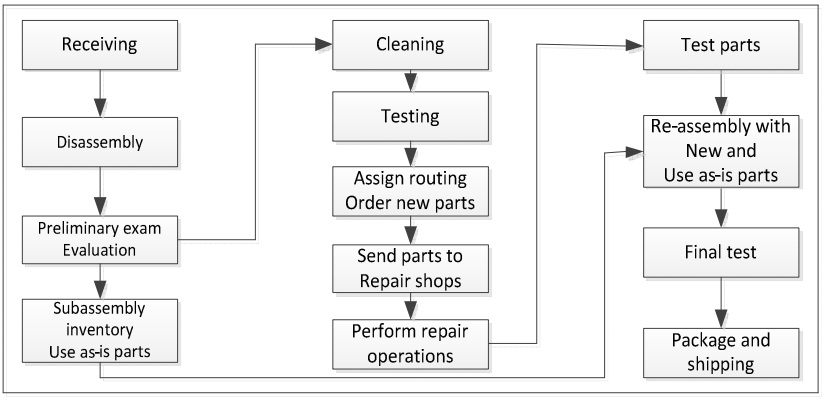


Fig. 1. The flow of a typical remanufacturing operation [4]

3 DHS Algorithm for the Dynamic FJSSP

3.1 Discrete Harmony Search

A solution in this paper consists of two vectors corresponding to the machine assignment and part scheduling sub-problems of the flexible job shop scheduling problem. Using harmony search terminology, a harmony is therefore composed of two parts:1) Machine assignment vector (hereafter called MA). (2) Part sequence vector (hereafter called PS).

Fig. 2(a) illustrates a machine assignment vector while Fig. 2(b) shows the corresponding part sequence. *Pi*,j is part *j* of product *i*. In MA, each element represents the machine selected for the corresponding part. In PS, the same elements represent the different parts of the same product. For example, the first 3 in MA means that machine 3 is selected for part *P*1,1, and the second 4 in PS is the second part of product 4.

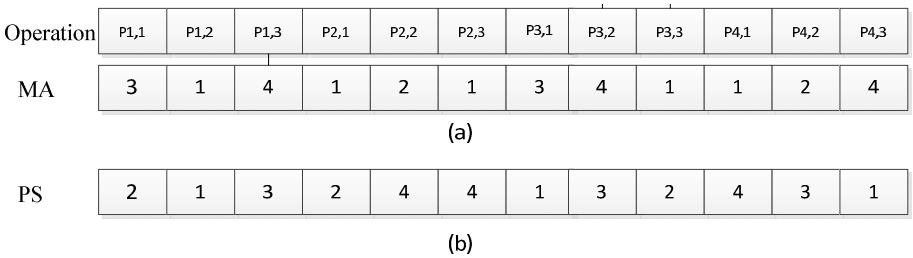


Fig. 2. Illustration of MA and PS

In our discrete harmony search algorithm, the quality of initial HM often affects the speed to a satisfactory solution. Therefore, it is a critical step to generate a good

quality initial HM. The initialization process includes machine assignment phase and part scheduling phase.

- Machine assignment procedure:

In this section, two heuristics are employed for initialing machine assignment.1) Random rule: For each part Pi,j, a randomly selected machine from the set of candidate machines will be placed at the corresponding position in MS vector. 2) Global min-processing time rule [9]: In this rule, we should find the global minimum processing time from the processing time table and fix the assignment. Then the processing time is added to corresponding machine. This rule considers the global workload among all machines and can help find the better makespan. The disadvantage is lack of diversity.

- Part scheduling procedure:

Once the machine assignment is fixed, the parts on each machine should be sequenced. The part scheduling component is obtained by the random rule: The part sequence is obtained by randomly shuffling the order of all parts on each machine.

3.2 Improving a New Harmony

In DHS algorithm, each harmony includes machine assignment and part sequence. In an MA vector, each element represents a machine selected for disassembly corresponding part. However, multi elements have the same value and represent the different parts of the same product in PS vector. From different harmonies, the elements at the same positions are for the same part in MA vector while the elements at the same position may mean the part of different products in PS vector. Therefore, we improve a new harmony for the machine assignment vector and part sequence vector separately. The process of improvising the new MA vector is shown in Fig.3. According to the problem feature of dynamic FJSSP and the coding strategies, we use crossover operators for the PS vector of a new harmony. In this study, we proposed a new crossover operator based on order crossover. We obtain two new harmonies from the current two harmonies. The proposed crossover operator works for the PS vector as follows:

Step1: Generate a random number *R* from 1 to product number;
Step2: Copy the values from the PS vector of Harmony1 to the corresponding positions in New Harmony1 where the values are less than or equal to *R*.
Step3: Copy the values from the PS vector of Harmony2 to the corresponding positions in New Harmony2 where the values are larger than *R*.
Step4: From the PS vector of Harmony2, copy the values which don't appear in New Harmony1 to the unfixed positions in New Harmony1 from left to right according to the order of the sequence in Harmony2.
Step5: From the PS vector of Harmony1, copy the values which don't appear in New Harmony2 to the unfixed positions in New Harmony2 from left to right according to the order of the sequence in Harmony1.

```
Procedure: Improving the new MA vector for new harmony
    For each product i
        For each part j of product i
            If Rand1(0-1)<HMCR                    // memory consideration
                select a machine k for Pi,j from the same position in harmony memory randomly
                If rand2(0-1)<PAR                 // pitch adjustment
                    If k is the last machine in the set of candidate machines
                        select the first machine k' in the set of candidate machines for Pi,j
                    else
                        select the next machine knext of machine k in the set of candidate machines for Pi,j
                Endif
            Else
                randomly select a machine kr from the set of dandidate machines for Pi,j   // a random selection
        Endfor
    Endfor
```

Fig. 3. The process of improvising the MA part of a new harmony

3.3 Dynamic Scheduling

In the remanufacturing process, the quality of received products is different. So, the processing time of parts is also different. If the processing time of one part becomes larger than pre-estimated and the makspan is also larger than pre-optimized, the following parts in the remanufacturing process will be rescheduled using proposed DHS algorithm for minimizing makespan.

4 Simulation and Computational Results

4.1 Experimental Setup

To test the performance of the proposed DHS algorithm, simulation and experiment are carried out in this section. The data of one instance is shown in Table 1. In Table1, p1-p6 are 6 products received for remanufacturing while the oij denotes the operation j of product i. M1-M5 are the machines for remanufacturing products. The values are the processing time of parts on corresponding machines and the values are the normal processing time. The level of parts which changes its processing time are calculated as Eq.2:

$$\alpha = \frac{\rho}{\xi} \times 100\% \tag{2}$$

where ρ is the number of parts with processing time larger than pre-estimated, ξ is the total number of parts of all products. The DHS algorithm were coded in C++.net and run on Intel 2.8GHz PC with 1 GB memory. In this paper, the parameters are fixed as follows: HMS=20, HMCR=0.95 and PAR=0.5. α is set two different levels, α=0.05 and α=0.1.

4.2 Simulation and Experiment Results

Before commencing the remanufacturing, the proposed DHS algorithm has calculated the makespan, $C_M = 27$, of a instance. For each α, we randomly select the operations to increase processing time and correspondingly added value of processing time. The simulation of remanufacturing process is run for 10 independent replications.The simulation results of α=0.05 and α=0.1 are shown in Table 1 and Table 2.

In Table 1 and Table 2, the first column is the parts which increase processing time. The second column is the added value of processing time. And the third column is the added value of makespan while no dynamic scheduling in remanufacturing process.The next column is the increase in percentage of makespan with no dynamic scheduling. The column second to last is the added value of makespan with dynamic scheduling in remanufacturing process. The final column is the increasing percentage of makespan with dynamic scheduling. It can be seen from Table 1 that the results with dynamic scheduling is better than that without dynamic scheduling. In 10 independent replications, the mean increase in percentage of makespan with dynamic scheduling is 3.33% while the result without dynamic scheduling is 5.92%. The similar conclusion can be obtained from Table 2. We can see from Table 2 that the

Table 1. Simulation results with α=0.05

Oi,j	Add1	Add2	imp (%)	Add3	imp (%)
(6,3)	3	2	7.40	1	3.70
(1,4)	1	0	0.00	0	0.00
(2,4)	2	1	3.70	0	0.00
(4,1)	3	3	11.11	1	3.70
(5,3)	3	2	7.40	2	7.40
(6,2)	2	2	7.40	1	3.70
(6,5)	4	1	3.70	1	3.70
(3,4)	1	1	3.70	1	3.70
(6,4)	3	1	3.70	1	3.70
(4,2)	2	3	11.11	1	3.70
Mean	2.4	1.6	5.92	0.9	3.33

Table 2. The simulation results of α=0.1

Oi,j	Add1	Add2	*imp* (%)	Add3	*imp* (%)
(5,1)(3,1)(1,2)	6	3	11.11	2	7.40
(3,2)(6,4)(1,5)	6	2	7.40	2	7.40
(5.3)(5.4)(3,4)	8	3	11.11	3	11.11
(3,1)(4,6)	5	3	11.11	3	11.11
(5,1)(1,2)(3,4)	5	3	11.11	2	7.40
(1,1)(4,3)(2,3)	5	3	11.11	3	11.11
(4,3)(4,6)(1,5)	8	6	22.22	4	14.81
(6,1)(6,2)(3,5)	7	5	18.51	5	18.51
(1,5)(5,2)(6,4)	4	2	7.40	2	7.40
(4,1)(6,4)(1,5)	5	2	7.40	1	3.70
Mean	5.9	3.2	11.85	2.7	10.00

mean increase in percentage of makespan with dynamic scheduling is 11.85% while the result without dynamic scheduling is 10.00%. In a nutshell, it can be concluded that our proposed DHS algorithm is effective for solving the dynamic FJSSP in remanufacturing engineering.

5 Conclusions

To the best of our knowledge, this is the first report to propose a DHS algorithm for the dynamic flexible job shop scheduling problem in remanufacturing engineering. A novel approach for improving a new harmony is proposed based on the problem characteristics and solution representation. The simulation and experiment are carried for evaluating the performance of the novel proposal. In future, we will improve the search ability of discrete harmony search algorithm and further investigate the local search method. We will also research more dynamic scheduling problem in remanufacturing engineering and other field.

Acknowledgements. This research is partially supported by Remanufacturing Scheduling Systems Program of A*Star in Singapore under Grant 112 290 4021.

References

1. Lund, R.I.: Remanufacturing. Technology Review 87(2), 18–23 (1984)
2. Krupp, J.A.: Structuring bills of material for automotive remanufacturing. Production and Inventory Management Journal 34(4), 46–52 (1993)
3. Krupp, J.A.: Core obsolescence forecasting in remanufacturing. Production and Inventory Management Journal 33(2), 12–17 (1992)
4. Guide Jr., V.D.R., Srivastava, R., Spencer, M.S.: A evaluation of capacity planning techniques in a remanufacturing environment. International Journal of Production Research 35(1), 67–82 (1997)
5. Guide Jr., V.D.R., Srivastava, R., Kraus, M.E.: Product structure complexity and scheduling of operations in recoverable manufacturing. International Journal of Production Research 35(11), 3179–3200 (1997)
6. DePuy, G.W., Usher, J.S., Walker, R.L., Taylor, G.D.: Production planning for remanufactured products. Production Planning & Control 18(7), 573–583 (2007)
7. Teunter, R.H., Bayindir, Z.P., Van Den Heuvel, W.: Dynamic lot sizing with product returns and remanufacturing. International Journal of Production Research 44(20), 4377–4400 (2006)
8. Grubbström, R.W., Tang, O.: Optimal production opportunities in a remanufacturing system. International Journal of Production Research 44(18), 3953–3966 (2006)
9. Denizel, M., Fergusonand, M., Souza, G.C.: Multiperiod remanufacturing planning with uncertain quality of inputs. IEEE Transactions on Engineering Management 57(3), 394–404 (2010)
10. Geem, Z.W., Kim, J.H., Loganathan, G.V.: A new heuristic optimization algorithm: harmony search. Simulation 76(2), 60–68 (2001)
11. Lee, K.S., Geem, Z.W., Lee, S.H., Bae, K.W.: The harmony search heuristic algorithm for discrete structural optimization. Eng. Optimiz. 37(7), 663–684 (2005)

12. Mahdavi, M., Fesanghary, M., Damangir, E.: An improved harmony search algorithm for solving optimization problems. Appl. Math. and Comput. 188(2), 1567–1579 (2007)
13. Geem, Z.W., Lee, K.S., Park, Y.: Application of harmony search to vehicle routing. Am. J. Appl. Sci. 2(12), 1552–1557 (2005)
14. Geem, Z.W.: Optimal cost design of water distribution networks using harmony search. Eng. Optimiz. 38(3), 259–280 (2006)
15. Geem, Z.W.: Optimal Scheduling of Multiple Dam System Using Harmony Search Algorithm. In: Sandoval, F., Prieto, A.G., Cabestany, J., Graña, M. (eds.) IWANN 2007. LNCS, vol. 4507, pp. 316–323. Springer, Heidelberg (2007)
16. Geem, Z.W.: Harmony Search Algorithm for Solving Sudoku. In: Apolloni, B., Howlett, R.J., Jain, L. (eds.) KES 2007, Part I. LNCS (LNAI), vol. 4692, pp. 371–378. Springer, Heidelberg (2007)
17. Zarei, O., Fesanghary, M., Farshi, B., Saffar, R.J., Rafar, M.R.: Optimization of multi-pass face-milling via harmony search algorithm. J. Mater. Process. Tech. 209(5), 2386–2392 (2009)
18. Coelho, L.S., Bernert, D.L.A.: An improved harmony search algorithm for synchronization of discrete-time chaotic system. Chaos Solitons Fract 41(5), 2526–2532 (2009)
19. Wang, L., Pan, Q.K., Tasgetiren, M.F.: Minimizing the total flow time in a flow shop with blocking by using hybrid harmony search algorithms. Expert Syst. Appl. 37(12), 7929–7936 (2010)
20. Pan, Q.K., Suganthan, P.N., Liang, J.J., Tasgetiren, M.F.: A local-best harmony search algorithm with dynamic sub-harmony memories for lot-streaming flow shop scheduling problem. Expert Syst. Appl. 38(4), 3252–3259 (2011)
21. Gao, K.Z., Pan, Q.K., Li, J.Q.: Discrete harmony search algorithm for the no-wait flow shop scheduling problem with total flow time criterion. Int. J. Adv. Manuf. Technol. 56, 683–692 (2011)
22. Gao, K.Z., Pan, Q.K., Li, J.Q., et al.: A hybrid harmony search algorithm for the no-wait flow shop scheduling problems. Asia-Pacific J. of Operational Research 29(2), 1250012 (23 pages) (2012)
23. Yadav, P., Kumar, R., Panda, S.K., Chang, C.S.: An improved harmony search algorithm for optimal scheduling of the diesel generators in oil rig platforms. Energ. Convers. Manage. 52(2), 893–902 (2011)
24. Geem, Z.W.: Novel Derivative of Harmony Search Algorithm for Discrete Design Variables. Applied Mathematics and Computation 199(1), 223–230 (2008)

Multilevel Image Thresholding Based on Tsallis Entropy and Differential Evolution

Soham Sarkar[1], Swagatam Das[2], and Sheli Sinha Chaudhuri[3]

[1] Electronics and Communication Engineering Department, RCC Institute of Information Technology, Kolkata – 700015, India
[2] Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata – 700108, India
[3] Electronics and Telecommunication Engineering Department, Jadavpur University, Kolkata – 700032, India
sarkar.soham@gmail.com, swagatamdas19@yahoo.co.in, shelism@rediffmail.com

Abstract. Image segmentation is known as one of the most critical task in image processing and pattern recognition in contemporary time, for this purpose Multi Level Thresholding based approach has been an acclaimed way out. Endeavor of this paper is to focus on obtaining the optimal threshold points by using Tsallis Entropy. In this paper, we have incorporated a Differential Evolution (DE) based technique to acquire optimal threshold values. Furthermore, results are compared with two state-of-art algorithms- a. Particle Swarm Optimization (PSO), and b. Genetic Algorithm (GA). Several image quality assessment indices are applied for the performance analysis of the outcome derived by applying the proposed algorithm.

Keywords: Multilevel Image Segmentation, Tsallis Entropy, Differential Evolution, MSSIM, WPSNR.

1 Introduction

IMAGE segmentation, the process of discriminating objects from its background in pixel level, has become the utmost component of image analysis. Over the years segmentation is being applied as a basic step for several computer vision applications like feature extraction, identification, image registration etc. Image segmentation done via bi-level thresholding that subdivides the image into two homogenous regions, based on texture, histogram, edge etc., uses only one threshold value.

In the year 2004, bi-level maximum Tsallis entropy (MTE) based image segmentation was proposed by Portes de Albuquerque et al [1]. Proposed technique of image segmentation is based on work done by Tsallis et. al. [2], which is an extended version of Havrda and Charva't paper in 1967 [3]. Later multi-level image segmentation gained popularity for its ability for sub-dividing the image into more than one segment. It makes the image more useful for the later analysis and studies. However, the computation complexity of these methods had increased to a significant

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 17–24, 2012.

amount [4]. Now, to reduce the computational time, several multi–level Tsallis entropy based image segmentation techniques are being proposed by using some state-of-art metaheuristics of recent time, like Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Bacterial Foraging Algorithm (BFA) [5]. In this paper Differential Evolution (DE) has been used to find the maximum Tsallis entropy for accurate and faster computation. DE is no doubt, a powerful and real parameter optimizer of current time [6, 7]. It has been shown that DE can outperform GA and PSO when it is used for multi-level thresholding based image segmentation problems [8].

The rest of the paper includes the basic concept of Tsallis entropy -Section 2. Section 3 describes multi-level image thresholding. A brief introduction of Differential Evolution (DE) is given in Section 4. The experimental results and comparative performance are presented in Section 5. Lastly the paper is concluded in Section 6.

2 Tsallis Entropy

Let $P = (p_1, p_2, p_3 , \ldots\ldots , p_n) \in \Delta_n$, where

$$\Delta_n = \left\{ (p_1, p_2, \ldots , p_n) \middle| p_i \geq 0,\ i = 1,2,\ldots, n,\ n \geq 2,\ \sum_{i=1}^{n} p_i = 1 \right\}$$

is a set of discrete finite *n*-ary probability distributions. Havrda and Charva't defined entropy of degree α as [3]:

$$H_n^{\alpha} = \frac{1}{1 - 2^{1-\alpha}} \left[1 - \sum_{i=1}^{n} p_i^{\alpha} \right]. \tag{1}$$

In 1988, Independently Tsallis proposed a one parameter generalization of the Shannon entropy as [16]

$$H_n^{\alpha}(P) = \frac{1}{1 - \alpha} \left[1 - \sum_{i=1}^{n} p_i^{\alpha} \right], \tag{2}$$

where α is a real positive parameter not equal to one. Both (1) and (2) have similar expression except the normalization factor. The Tsallis entropy is non-extensive in such a way that for a statistical independent system, the entropy of the system is defined by the following pseudo additive entropic rule [1]

$$H_T^{\alpha}(A + B) = H_T^{\alpha}(A) + H_T^{\alpha}(B) + (1 - \alpha) H_T^{\alpha}(A) H_T^{\alpha}(B) \tag{3}$$

For an image the entire distribution is divided into classes, one for object (class A) and another for background (class B).Then the priori Tsallis for each distribution can be defined as

$$H_A^{\alpha}(t) = \frac{1}{\alpha - 1} \left[1 - \sum_{i=0}^{t} \left(\frac{P(i)}{P_A(t)} \right)^{\alpha} \right], \tag{4}$$

And

$$H_B^{\alpha}(t) = \frac{1}{\alpha - 1}\left[1 - \sum_{i=t+1}^{L-1}\left(\frac{P(i)}{P_B(t)}\right)^{\alpha}\right], \tag{5}$$

where

$$P_A(t) = \sum_{i=0}^{t} P(i) \;\; and \; P_B(t) = \sum_{i=t+1}^{L-1} P(i)$$

The optimum threshold value can be determined by:

$$\varphi_{\alpha}(t) = Arg\ max([H_o^{\alpha}(t) + H_b^{\alpha}(t) + (1-\alpha)H_o^{\alpha}(t)H_b^{\alpha}(t)]) \tag{6}$$

3 Multi-level Tsallis Entropy

The Tsallis global thresholding method can be further extended by using more than two classes e.g. class A, class B and class C [9, 10, 11]. The entropy of the system for threshold values t_1 and t_2 is defined by the following pseudo additive entropic rule

$$\varphi_{\alpha}(t) = Argmax([H_A^{\alpha}(t) + H_B^{\alpha}(t) + H_C^{\alpha}(t) + (1-\alpha)H_A^{\alpha}(t)H_B^{\alpha}(t)H_C^{\alpha}(t)]), \tag{7}$$

where

$$H_A^{\alpha}(t) = \frac{1}{\alpha - 1}\left[1 - \sum_{i=0}^{t1}\left(\frac{P(i)}{P_A(t)}\right)^{\alpha}\right], \qquad P_A(t) = \sum_{i=0}^{t1} P(i)\,,$$

$$H_B^{\alpha}(t) = \frac{1}{\alpha - 1}\left[1 - \sum_{i=t1+1}^{t2}\left(\frac{P(i)}{P_B(t)}\right)^{\alpha}\right], \quad P_B(t) = \sum_{i=t1+1}^{t2} P(i),$$

$$H_C^{\alpha}(t) = \frac{1}{\alpha - 1}\left[1 - \sum_{i=t2+1}^{L-1}\left(\frac{P(i)}{P_C(t)}\right)^{\alpha}\right], \quad P_C(t) = \sum_{i=t2+1}^{L-1} P(i).$$

A more generalized equation can be reformed from (7) for n threshold values:

$$\begin{aligned}\varphi_{\alpha}(t_1, t_2, \ldots, t_n) = Argmax([&H_A^{\alpha}(t) + H_B^{\alpha}(t) + \cdots + H_n^{\alpha}(t) \\ &+ (1-\alpha).H_A^{\alpha}(t)H_B^{\alpha}(t).\ldots.H_n^{\alpha}(t)]),\end{aligned} \tag{8}$$

where no of segmentation level would be n+1. For the ease of computation two dummy threshold $t_0 \equiv 0$, $t_{n+1} \equiv L-1$ and $t_0 < t_1 < \cdots < t_n < t_{n+1}$, are being incorporated. DE is used to find the threshold values by maximizing equation (8).

4 Differential Evolution (DE)

DE, a population-based global optimization algorithm, was proposed by Storn in 1997. The i^{th} individual (parameter vector) of the population at generation (time) t is a D-dimensional vector containing a set of D optimization parameters:

$$\overrightarrow{Z_i}(t) = [Z_{i,1}(t), Z_{i,2}(t), \ldots\ldots, Z_{i,D}(t)] \tag{9}$$

In each generation to change the population members $\overrightarrow{Z_i}(t)$ (say), a *donor* vector $\overrightarrow{Y_i}(t)$ is created. It is the method of creating this donor vector that distinguishes the various DE schemes. In one of the earliest variants of DE, now called DE/rand/1 scheme, to create $\overrightarrow{Y_i}(t)$ for each i^{th}member, three other parameter vectors (say the r_1, r_2, and r_3-th vectors such that $r_1, r_2, r_3 \in [1, NP]$ and $r_1 \neq r_2 \neq r_3$) are chosen at random from the current population. The donor vector $\overrightarrow{Y_i}(t)$ is then obtained multiplying a scalar number *F* with the difference of any two of the three. The process for the j^{th} component of the i^{th} vector may be expressed as,

$$\overrightarrow{Y_{i,j}}(t) = Z_{r1,j}(t) + F.\left(Z_{r2,j}(t) - Z_{r3,j}(t)\right) \tag{10}$$

A 'binomial' crossover operation takes place to increase the potential diversity of the population. The binomial crossover is performed on each of the D variables whenever a randomly picked number between 0 and 1 is within the Cr value. In this case the number of parameters inherited from the mutant has a (nearly) binomial distribution. Thus for each target vector $\overrightarrow{Z_i}(t)$, a trial vector $\overrightarrow{R_i}(t)$is created in the following fashion:

$$\begin{aligned} R_{i,j}(t) &= Y_{i,j}(t) \quad if\ rand_j(0,1) \leq Cr\ or\ j = rn(i) \\ &= Z_{i,j}(t) \quad if\ rand_j(0,1) > Cr\ or\ j \neq rn(i) \end{aligned} \tag{11}$$

For *j = 1, 2, ….., D* and $rand_j(0,1) \in [0,1]$ is the j^{th} evaluation of a uniform random number generator. $rn(i) \in [1, 2, \ldots\ldots, D]$ is a randomly chosen index to ensures that $\overrightarrow{R_i}(t)$ gets at least one component from $\overrightarrow{Z_i}(t)$. Finally 'selection' is performed in order to determine which one between the target vector and trial vector will survive in the next generation i.e. at time $t = t + 1$. If the trial vector yields a better value of the fitness function, it replaces its target vector in the next generation; otherwise the parent is retained in the population:

$$\left.\begin{aligned} \overrightarrow{Z_i}(t+1) &= \overrightarrow{R_i}(t) \quad if\ \ f\left(\overrightarrow{R_i}(t)\right) \leq f\left(\overrightarrow{Z_i}(t)\right) \\ &= \overrightarrow{Z_i}(t) \quad if\ \ f\left(\overrightarrow{R_i}(t)\right) > f\left(\overrightarrow{Z_i}(t)\right) \end{aligned}\right\} \tag{12}$$

where $f(.)$ is the function to be minimized.

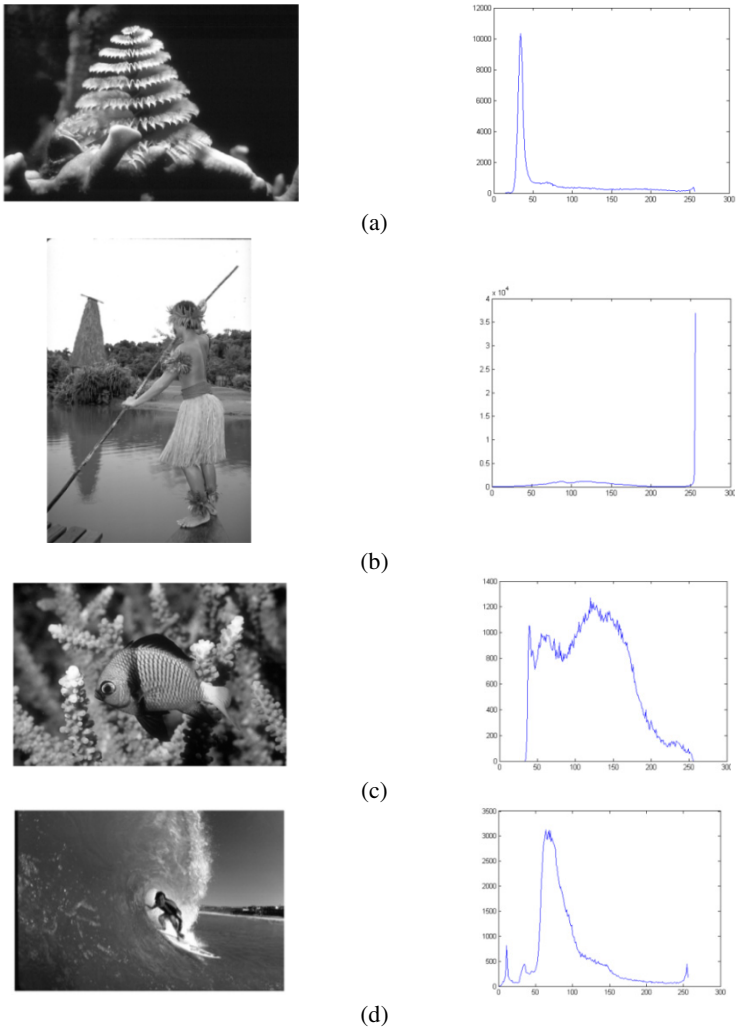
5 Experimental Results

The simulations are performed with MATLAB R2011b in a workstation with Intel® Core™ i3 3.2 GHz processor. For testing and analysis, 4 images are used from the Berkeley Segmentation Dataset and Benchmark (obtainable from http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/). The original gray scale images, their 1-D histograms are shown in Figure 1. The segmented greyscale image is formed by using a generalized equation.

$$f_S(x,y) = \left\{k * \left\lfloor \frac{L-1}{n+1} \right\rfloor \quad if\ (t_{k-1}, s_{k-1}) < f(x,y) \leq (t_k, s_k), \right. \tag{13}$$

where k = 1, 2, … ,n+1.

The performance of DE based method is compared with other efficient global optimization techniques like GA and PSO. In case of DE, the following parametric setup is used for all the test images: $Cr = 0.9,\ F = 0.5$. Best possible parametric setup is also maintained for GA and PSO. The used parametric setup for PSO is: $C_1 = C_2 = 2$ and w = 0.9 and for GA: crossover probability = 0.85 and mutation probability = 0.1 Results are reported as the mean of the objective functions of 50 independent runs. Each run contains 200 generations. The value of α is set to 0.3 .Through detail analysis it is found that Tsallis entropy based multi-level segmentation performs efficiently for α's value below 0.4.



(a) (b) (c) (d)

Fig. 1. Test Images and their histograms (a) 12074 (b) 101087 (c) 209070 (d) 300091

Table 1 shows the required computational time for each levels of image "12074", using DE, GA and PSO. Results clearly show that DE requires lesser computation time than the other algorithms. In addition, mean objective function value (f_{mean}) and standard deviation (f_{std}), shown in Table 2, establish the superiority of DE over modern day's state-of-art global optimization techniques. (For convenience, best results are shown in bold letters.)

Table 1. Average Computational time in Seconds (Image No. 12074) for 200 generation in a single run

No. Of Levels	DE	PSO	GA
2	**1.1019**	1.1595	1.6945
3	**1.4901**	1.9173	2.4987
4	**1.5390**	2.2348	3.4223

Table 2. Mean objective function value (f_{mean}) and standard deviation (f_{std}) of "12074"

L	DE		PSO		GA	
	f_{mean}	f_{std}	f_{mean}	f_{std}	f_{mean}	f_{std}
2	**1.1889e+004**	**0.0**	1.1886e+004	2.5793	1.1880e+004	7.640
3	**1.8243e+005**	**0.0**	1.8165e+005	6.3079e+002	1.8236e+005	45.1361
4	**1.8436e+006**	**0.0**	1.8040e+006	1.7922e+004	1.8311e+006	5.4529e+003

Different threshold values of test images, obtained by using meta-heuristics, are given in Table 3. Fig. 2 shows the segmented gray level test images. Weighted Peak Signal to Noise Ratio (WPSNR) [14] and Mean Structural Similarity Index Measurement (MSSIM) [15] (between original grayscale image and segmented grayscale image) are indicated to establish the betterment of results.

Table 3. Threshold values acquired by using DE, GA, PSO

Image	No. of Levels	Threshold values											
		DE				PSO				GA			
12074	2	95	175			95	175			99	177		
	3	78	137	196		77	138	189		79	136	196	
	4	68	115	162	209	64	112	156	207	70	118	163	209
101087	2	68	140			67	140			67	140		
	3	51	105	159		51	105	159		54	109	162	
	4	42	85	130	174	46	87	133	177	41	82	128	173
209070	2	109	185			109	185			110	187		
	3	90	146	201		91	146	198		90	150	202	
	4	79	123	168	211	79	122	165	209	80	126	169	211
300091	2	99	173			99	173			99	173		
	3	53	115	181		53	117	184		53	117	184	
	4	53	102	153	203	53	102	151	205	53	103	154	203

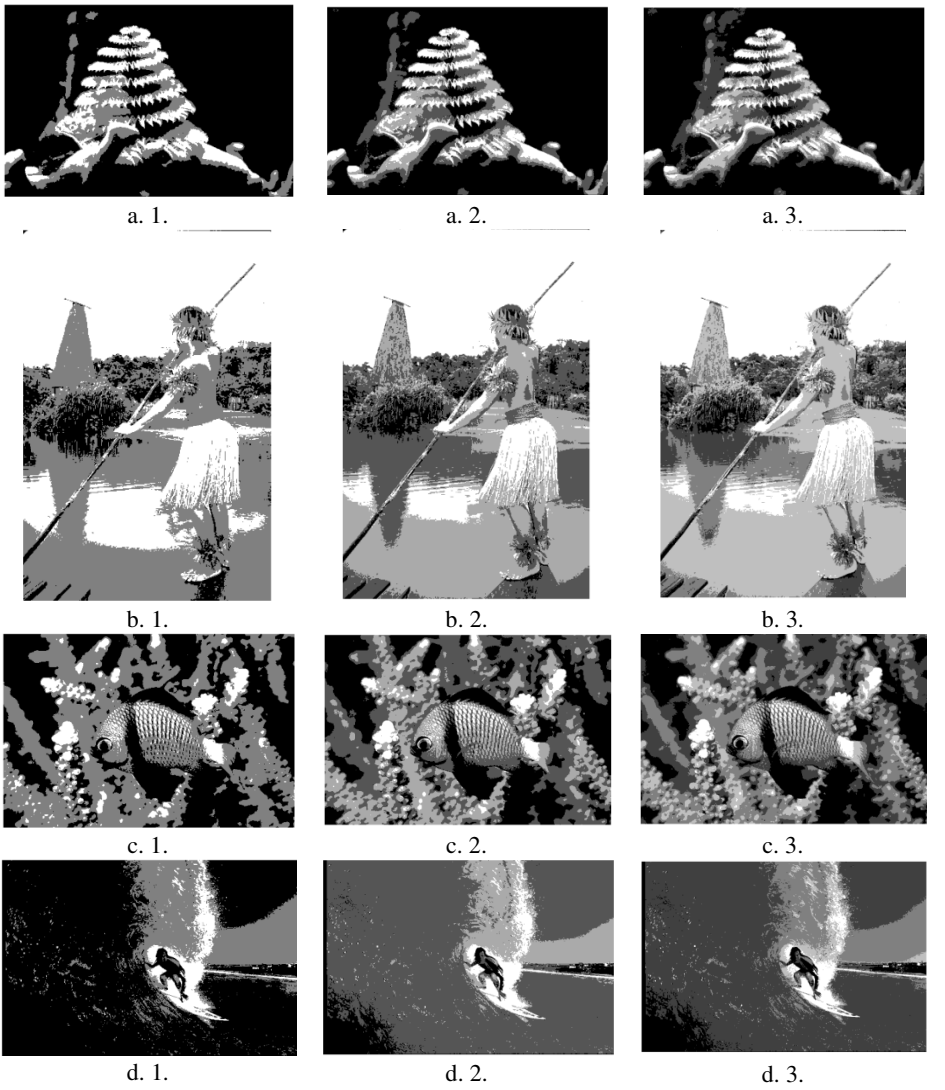


Fig. 2. Segmented images obtained by MTE-DE method ((a. 1.), (b. 1.), (c. 1.), (d. 1.) 3-level thresholding, (a. 2.), (b. 2.), (c. 2.), (d. 2.) 4-level thresholding, (a. 3.), (b. 3.), (c. 3.), (d. 3.) 5-level thresholding

Table 4. WPSNR and MSSIM of test images using DE

Name of Images	WPSNR(dB)			MSSIM		
	2	3	4	2	3	4
12074	22.0073	23.1569	23.9040	0.1770	0.2283	0.2694
101087	21.7986	24.0456	26.1610	0.6749	0.7495	0.7762
209070	20.0278	21.8441	22.8655	0.3302	0.4363	0.4999
300091	18.6277	27.7601	30.0469	0.2010	0.6748	0.7080

6 Conclusion

In this paper we have proposed a scheme based on differential evolution for Multi-Level Thresholding by using MTE. DE approach has definitely increased the speed and accuracy of our selected algorithm. This technique was applied to various real images; the results demonstrated the efficiency of working of the algorithm, and the feasibility of our proposal. This finding could encourage further researches, still 2-D histogram based approaches and fuzzy based approach could be implemented to achieve better performance.

References

[1] Portes de Albuquerque, M., Esquef, I.A., Mello, A.R.G.: Image thresholding using Tsallis entropy. Pattern Recognition Letters 25, 1059–1106 (2004)
[2] Tsallis, C.: Possible generalization of Boltzmann-Gibbs statistics. Journal of Statistical Physics 52, 479–487 (1988)
[3] Havrda, J., Charvát, F.: Quantification methods of classification processes: Concept of structural α-entropy. Kybernetica (Prague) 3, 95–100 (1967)
[4] Kittler, J., Illingworth, J.: Minimum error thresholding. Pattern Recognition 19, 41–47 (1986)
[5] Hammouchea, K., Diaf, M., Siarry, P.: A comparative study of various meta-heuristic techniques applied to the multilevel thresholding problem. Engineering Applications of Artificial Intelligence 23(5), 676–688 (2010)
[6] Storn, R., Price, K.V.: Differential Evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report TR-95-012, ICSI (1995), `http://http.icsi.berkeley.edu/~storn/litera.html`
[7] Das, S., Suganthan, P.N.: Differential evolution – a survey of the state-of-the-art. IEEE Transactions on Evolutionary Computation 15(1), 4–31 (2011)
[8] Sarkar, S., Patra, G.R., Das, S.: A Differential Evolution Based Approach for Multilevel Image Segmentation Using Minimum Cross Entropy Thresholding. In: Panigrahi, B.K., Suganthan, P.N., Das, S., Satapathy, S.C. (eds.) SEMCCO 2011, Part I. LNCS, vol. 7076, pp. 51–58. Springer, Heidelberg (2011)
[9] Sathya, P.D., Kayalvizhi, R.: PSO-based Tsallis thresholding selection procedure for image segmentation. International Journal of Computer Applications 5(4), 39–46 (2010)
[10] Sathya, P.D., Kayalvizhi, R.: Modified bacterial foraging algorithm based multilevel thresholding for image segmentation. Engineering Applications of Artificial Intelligence 24, 595–615 (2011)
[11] Zhang, Y., Wu, L.: Optimal multi-level thresholding based on maximum Tsallis entropy via an artificial bee colony approach. Entropy 13, 841–859 (2011)
[12] Miyahara, M., Kotani, K., Algazi, V.R.: Objective picture quality scale (PQS) for image coding. IEEE Trans. on Communications 46(9), 1215–1226 (1998)
[13] Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E.: Image quality assessment: From error visibility to structural similarity. IEEE Trans. Image Processing 13(4), 600–612 (2004)

Convergence and Boundary Estimation of the Particle Dynamics in Generalized Particle Swarm Optimization

Dipankar Maity and Udit Halder

Dept. of Electronics and Telecomm. Engineering, Jadavpur University, Kolkata, India
{dipankarmaity1991,udithalder99}@gmail.com

Abstract. Concept of the particle swarms emerged from a simulation of the collective behavior of social creatures and gradually evolved into a powerful global optimization technique, now well-known as the Particle Swarm Optimization (PSO). A vast amount of analytical studies on various aspects of the PSO dynamics like stability, convergence, explorative power, sampling distribution and so on can be found in the literature. The boundary of the swarm is still as a challenging research interest. The upper boundary restricts the swarm members within a sub-region of the whole search space. Higher the upper boundary, higher is the diversity. This paper investigates mainly the diversity of the swarm in terms of the upper boundary of the swarm.

Keywords: Particle Swarm Optimization, Convergence, Exploration, diversity.

1 Introduction

The concept of function-optimization by means of a particle swarm was introduced by Kennedy and Eberhart [1] in 1995. The velocity and the position update equation at the t+1th generation is given by the following equations-

$$\boldsymbol{v}_i(t+1) = \omega \boldsymbol{v}_i(t) + a \cdot rand_1(\boldsymbol{l}_i(t) - \boldsymbol{x}_i(t)) + b \cdot rand_2(\boldsymbol{g}_i(t) - \boldsymbol{x}_i(t)), \quad (1)$$

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1). \quad (2)$$

The first term of the velocity updation formula represents the inertial velocity of the particle. ω is called the "inertia factor". Here, $\boldsymbol{l}_i(t)$ is the best previous position found by the i^{th} particle and $\boldsymbol{g}_i(t)$ is the best position discovered by the whole population at t^{th} iteration. '*a*' and '*b*' are the acceleration coefficients reflecting the weighting of stochastic acceleration terms that pull each particle toward *pbest* and *gbest* positions. Venter and Sobieski [2] termed '*a*' as "self-confidence" and '*b*' as "swarm confidence". Local neighborhood models [3] (or *lbest*) were proposed for PSO long ago, where each particle has access to the information corresponding to its immediate neighbors, according to a certain swarm topology. The velocity and the position update equation for *lbest* topology at the t+1th generation is given by the following equations-

$$\boldsymbol{v_i}(t+1) = \omega \boldsymbol{v_i}(t) + a \cdot rand_1(\boldsymbol{l_i}(t) - \boldsymbol{x_i}(t)) + b \cdot rand_2(\boldsymbol{n_i}(t) - \boldsymbol{x_i}(t)). \quad (3)$$

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 25–32, 2012.

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1), \tag{4}$$

where $\boldsymbol{n}_i(t)$ is the neighborhood best particle found by the $\boldsymbol{x}_i(t)$. A comprehensive knowledge on the foundation and application of PSO can be found in [4-6].

Ozcan and Mohan [7] considered single particle moving towards fixed attracters which was extended to a multi particle and multidimensional space in [8]. Clerc and Kennedy did a stability analysis in [9]. Recently Fernández-Martínez and García Gonzalo [10] found a continuous form of *gbest* PSO.

All of the analysis discussed so far is on the *gbest* PSO. A preliminary probabilistic analysis of the particle interaction and information exchange in an *lbest* PSO was addressed by Ghosh *et al*. [11]. But none of the analysis addressed the boundary of the swarm trajectory of *lbest* PSO. We believe that analysis of *lbest* PSO will help us to develop efficient optimizers based on *lbest* swarms.

2 Analytical Treatment

2.1 Error Dynamics in Generalized lbest PSO

Let the global optimum to be found be denoted with $\boldsymbol{x}^*$. If none of the particles are successful (till now) to find the optimum, the position of each particle is given as:

$$\boldsymbol{x}_i(t) = \boldsymbol{x}^* + \boldsymbol{e}_i(t), \tag{5}$$

where $\boldsymbol{e}_i(t)$ represents the error of the i^{th} particle at time $t = t$. The PSO dynamics can be generalized [10] and the velocities and positions update equations are:

$$\boldsymbol{v}_i(t+\Delta t) = \left(1-(1-\omega)\Delta t\right)\cdot \boldsymbol{v}_i(t) + a\cdot rand_1\cdot \Delta t(\boldsymbol{l}_i(t)-\boldsymbol{x}_i(t)) + b\cdot rand_2\cdot \Delta t(\boldsymbol{n}_i(t)-\boldsymbol{x}_i(t)). \tag{6}$$

$$\boldsymbol{x}_i(t+\Delta t) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+\Delta t)\cdot \Delta t\,. \tag{7}$$

From equation (6) and (7) we get:

$$\ddot{\boldsymbol{x}}_i(t) + (1-\omega)\dot{\boldsymbol{x}}_i(t) + c\boldsymbol{x}_i(t) = a\cdot rand_1\cdot \boldsymbol{l}_i(t) + b\cdot rand_2\cdot \boldsymbol{n}_i(t), \tag{8}$$

where $\dot{\boldsymbol{x}}_i(t) = \lim_{\Delta t\to 0}\left(\boldsymbol{x}_i(t)-\boldsymbol{x}_i(t-\Delta t)\right)/\Delta t$, $\ddot{\boldsymbol{x}}_i(t) = \lim_{\Delta t\to 0}\left(\boldsymbol{x}_i(t+\Delta t)-2\boldsymbol{x}_i(t)+\boldsymbol{x}_i(t-\Delta t)\right)/(\Delta t)^2$, and $c = a\cdot rand_1 + b\cdot rand_2$. Now, if we substitute $\boldsymbol{x}_i(t) = \boldsymbol{x}^* + \boldsymbol{e}_i(t)$ in (8), we get,

$$\ddot{\boldsymbol{e}}_i(t) + (1-\omega)\dot{\boldsymbol{e}}_i(t) + c\boldsymbol{e}_i(t) = a\cdot rand_1\cdot \boldsymbol{l}_i(t) + b\cdot rand_2\cdot \boldsymbol{n}_i(t) - c\boldsymbol{x}^*. \tag{9}$$

By letting $E(\boldsymbol{e}_i(t)) = \boldsymbol{\mu}(t)$ and applying expectation operator on both sides of equation (9) and by interchanging the order of differentiation and expectation, we obtain:

$$\ddot{\boldsymbol{\mu}}(t) + (1-\omega)\dot{\boldsymbol{\mu}}(t) + c'\boldsymbol{\mu}(t) = \boldsymbol{\theta}(t) - c'\boldsymbol{x}^*, \tag{10}$$

where $\boldsymbol{\theta}(t) = E(a\cdot rand_1\cdot \boldsymbol{l}_i(t) + b\cdot rand_2\cdot \boldsymbol{n}_i(t))$ and $c' = E(c) = (a+b)/2$. Equation (10) is a second order differential equation of $\boldsymbol{\mu}(t)$ and it has a general solution and a particular

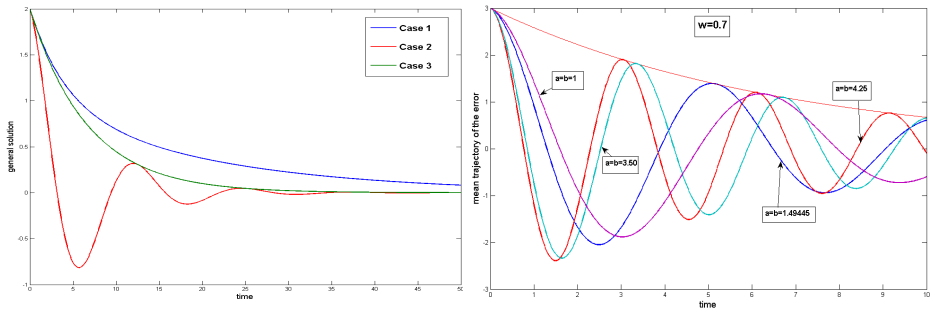
solution. Let $\mu(t)=\mu_g(t)+\mu_p(t)$, where $\mu_g(t)$ and $\mu_p(t)$ are the general and particular solutions of equation (10) respectively. In (10) the term $(1-\omega)\dot{\mu}(t)$ is the damping term of the stochastic process $\mu(t)$. If this term is absent then the general solution of the equation will be oscillating and will never converge to any stable point. Therefore $\omega=1$ is not acceptable to have convergence. If damping co-efficient is negative then also $\mu(t)$ will not converge rather it will diverge to large values eventually. So $\omega>1$ is also not acceptable and to have a converging behavior of $\mu(t)$ we must choose $\omega<1$. Now if we try to get the general solution of equation (10), we will get

$$\mu_g(t) = \left(C_1.e^{\lambda_1 t} + C_2.e^{\lambda_2 t}\right),$$

Where C_1 and C_2 depends on the initial conditions, putting $\lambda_1,\lambda_2 = \frac{-(1-\omega)\pm\sqrt{(1-\omega)^2-4c'}}{2}$, the $\mu_g(t)$ term can be rewritten as: $\mu_g(t)=e^{-(1-\omega)t/2}\left(C_1.e^{(\sqrt{(1-\omega)^2-4c'})t/2}+C_2.e^{-(\sqrt{(1-\omega)^2-4c'})t/2}\right)$.

The value of $(1-\omega)^2-4c'$ determines the nature of the $\mu_g(t)$ and the following three cases may arise:

i) Case 1: $(1-\omega)^2-4c'>0$, ii) Case 2: $(1-\omega)^2-4c'=0$, iii) Case 3: $(1-\omega)^2-4c'<0$.



(a) $\mu_g(t)$ vs. time for all 3 cases (b) Case 3 with ω=0.7 with three sets of *a,b*

Fig. 1. Variation of the general solution with time

For all the three cases the graphical plots of how the general solution varies with time are shown in Figure 1. From Figure 1 (b) we can quantitatively say that if '*a*' and '*b*' are high, then for small influence of social and cognitive component, high '*a*' and '*b*' magnify its magnitude and show more oscillation in the mean error trajectory. Similarly for small values of '*a*' and '*b*', the error trajectory comes to a stagnation after a small number of iterations. In both the cases, 1 and 2, $(1-\omega)^2-4c'\geq 0$ and thus, $4c'\leq(1-\omega)^2$ implies $c'\leq(1/4)$. This leads a small value of '*a*' and '*b*'. For Case 3, $4c'\geq(1-\omega)^2$. So, for case 3 the value of c' and hence the values of '*a*' and '*b*' are high and higher values give better diversity in the swarm. In order to further

elaborate on this point, we shall take Case 3 for further analysis. Substituting $4c' - (1-\omega)^2 = 4\delta^2$ in the expression for $\mu_g(t)$ we get,

$$\boldsymbol{\mu}_g(t) = e^{-(1-\omega)t/2}(\boldsymbol{C}\cos(\delta t + \Theta)). \tag{11}$$

Where $\boldsymbol{C}$ and Θ depends on the initial conditions. The effect of choosing different '*a*' and '*b*' on convergence of the general solution is already shown in Figure 1(b). Now we try to find the particular solution of the second order differential equation (10) in the following way, where α and β are roots of the equation $D^2 + (1-\omega)D + c' = 0$.

$$\boldsymbol{\mu}_p(t) = \frac{1}{(D-\alpha)(D-\beta)}.(\boldsymbol{\theta}(t) - c'\boldsymbol{x}^*)$$

$$\boldsymbol{\mu}_p(t) = \frac{a}{(D-\alpha)(D-\beta)}E(rand_1 \cdot \boldsymbol{l}_i(t)) + \frac{b}{(D-\alpha)(D-\beta)}E(rand_2 \cdot \boldsymbol{n}_i(t)) - \boldsymbol{x}^*. \tag{13}$$

Now let $E(\boldsymbol{l}_i(t)) = \boldsymbol{L}(t)$ and $E(\boldsymbol{n}_i(t)) = \boldsymbol{N}(t)$. Also let $\boldsymbol{l}_i(t) = \boldsymbol{L}(t) + \boldsymbol{d}_i(t)$ so that $E(\boldsymbol{d}_i(t)) = 0$. Similarly let $\boldsymbol{n}_i(t) = \boldsymbol{N}(t) + \boldsymbol{d}_i'(t)$ and $E(\boldsymbol{d}_i'(t)) = 0$. Again, we have $E(rand_1 \cdot \boldsymbol{L}(t)) = \boldsymbol{L}(t) \cdot E(rand_1) = \boldsymbol{L}/2$. Same also holds for $E(rand_1 \cdot \boldsymbol{N}(t))$. Hence,

$$\boldsymbol{\mu}_p(t) = \frac{1}{2(D-\alpha)(D-\beta)}(a \cdot \boldsymbol{L}(t) + b \cdot \boldsymbol{N}(t)) + \frac{1}{(D-\alpha)(D-\beta)}E(rand_1 \cdot a \cdot \boldsymbol{d}_i(t) + rand_2 \cdot b \cdot \boldsymbol{d}_i'(t)) - \boldsymbol{x}^*. \tag{14}$$

By letting $\boldsymbol{L}(t) = \boldsymbol{x}^* + \boldsymbol{K}_1(t)$ and $\boldsymbol{N}(t) = \boldsymbol{x}^* + \boldsymbol{K}_2(t)$ we get from above equation:

$$\boldsymbol{\mu}_p(t) = \frac{1}{2(D-\alpha)(D-\beta)} \cdot (a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)) + \frac{1}{(D-\alpha)(D-\beta)} \cdot E(a \cdot rand_1 \cdot \boldsymbol{d}_i(t) + b \cdot rand_2 \cdot \boldsymbol{d}_i'(t)). \tag{15}$$

Now $\boldsymbol{d}_i(t)$ is the deviation of $\boldsymbol{l}_i(t)$ from its mean, $\boldsymbol{L}(t)$. The position of local best at *t*-th generation of each particle in the swarm does not depend on the random number generated at *t*+1-th generation. So $\boldsymbol{d}_i(t)$ and $rand_1$ are uncorrelated and we get

$$\mathrm{Cov}(\boldsymbol{d}_i(t), rand_1) = E(\boldsymbol{d}_i(t) \cdot (rand_1)) - E(\boldsymbol{d}_i(t)) \cdot E(rand_1).$$

$$E(rand_1 \cdot \boldsymbol{d}_i(t)) = E(rand_1(t)) \cdot E(\boldsymbol{d}_i(t)) = 0 \text{ .as } E(\boldsymbol{d}_i(t)) = 0.$$

Similarly we can prove that $E(rand_2 \cdot \boldsymbol{d}_i'(t)) = 0$. From equation (15) we can obtain:

$$\boldsymbol{\mu}_p(t) = \frac{1}{2(D-\alpha)(D-\beta)}(a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)).$$

$$\Rightarrow |\boldsymbol{\mu}_p(t)| \le \left|\frac{1}{2(\alpha-\beta)}\right| \left(\left|e^{\alpha t}\right| \int \left|e^{-\alpha t}\right| \left|a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)\right| dt + \left|e^{\beta t}\right| \int \left|e^{-\beta t}\right| \left|a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)\right| dt \right). \tag{16}$$

α and β being the roots of the equation $D^2 + (1-\omega)D + c' = 0$, they are complex conjugate for the case 3 and thus we get from equation (16):

$$|\boldsymbol{\mu}_p(t)| \le \left|\frac{1}{\alpha-\beta}\right| \cdot e^{\mathrm{Re}(\alpha)t} \cdot \int e^{-\mathrm{Re}(\alpha)t} \left|a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)\right| dt. \tag{17}$$

Now we can write $|\boldsymbol{K}_1(t)| = |E(\boldsymbol{l}_i(t) - \boldsymbol{x}^*)| \le E\left(|\boldsymbol{l}_i(t) - \boldsymbol{x}^*|\right)$, and $|\boldsymbol{K}_2(t)| \le E\left(|\boldsymbol{n}_i(t) - \boldsymbol{x}^*|\right)$. $\boldsymbol{l}_i(t)$ is the local best position and in the vicinity of the global optimum, as the particle comes closer to the global optimum the local best position is updated otherwise it remains unchanged. Thus $|\boldsymbol{K}_1(t)|$ never degrades with time; same for $|\boldsymbol{K}_2(t)|$ also. Thus we can say $|a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)|$ is either a decreasing function with time or remains constant (when the swarm does not find any better position than the initial position). Now $|\boldsymbol{l}_i(t) - \boldsymbol{x}^*| \le |\boldsymbol{l}_i(0) - \boldsymbol{x}^*| = |\boldsymbol{x}(0) - \boldsymbol{x}^*| = |\boldsymbol{\mu}(0)|$ and similarly $|\boldsymbol{n}_i(t) - \boldsymbol{x}^*| \le \boldsymbol{\mu}(0)$. By this way we get $|a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)| \le |a + b| \cdot |\boldsymbol{\mu}(0)|$. Substituting this expression of $|a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)|$ in equation (17) we get,

$$|\boldsymbol{\mu}_p(t)| \le \left|\frac{1}{\alpha - \beta}\right| \cdot e^{\mathrm{Re}(\alpha)t} \cdot \int e^{-\mathrm{Re}(\alpha)t} |a + b| \cdot |\boldsymbol{\mu}(0)| \, dt.$$

$$|\boldsymbol{\mu}_p(t)| \le \left|\frac{a+b}{\delta}\right| \cdot \frac{|\boldsymbol{\mu}(0)|}{1-\omega}. \quad \left[\mathrm{Re}(\alpha) = -\frac{1-\omega}{2} \text{ and } |\alpha - \beta| = 2\delta\right] \tag{18}$$

So without making any impractical assumption, we succeeded to conclude that mean swarm trajectory will remain bounded with in the upper limit given by $|a + b||\boldsymbol{\mu}(0)|/\left((1-\omega)|\delta|\right)$ for any value of *t*, whatever is the objective function. This upper limit is high for higher values of '*a*' and '*b*'. Now from equation (17) we again get

$$|\boldsymbol{\mu}_p(t)| \le \left|\frac{1}{\alpha - \beta}\right| \cdot e^{\mathrm{Re}(\alpha)t} \cdot \int e^{-\mathrm{Re}(\alpha)t} |a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)| dt.$$

Now we define upper bound $(\boldsymbol{u}(t))$ of the mean swarm trajectory at time *t* as

$$\boldsymbol{u}(t) = \left|\frac{1}{\alpha - \beta}\right| \cdot e^{\mathrm{Re}(\alpha)t} \cdot \int e^{-\mathrm{Re}(\alpha)t} |a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)| dt.$$

$$\dot{\boldsymbol{u}}(t) = \left|\frac{1}{\alpha-\beta}\right| \cdot \left[\mathrm{Re}(\alpha) \cdot e^{\mathrm{Re}(\alpha)t} \cdot \int e^{-\mathrm{Re}(\alpha)t} |a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)| dt + |a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)|\right] \cdot \tag{19}$$

As we said before that $|a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)|$ is either a constant function or a decreasing function of time so the derivative of this is always either zero or negative for any value of *t*. We define here $\frac{d}{dt}\left(|a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)|\right) = -\boldsymbol{\Psi}(t)$ where $\boldsymbol{\Psi}(t) \ge 0 \quad \forall t$. Using this we are going to prove that the upper boundary of the swarm trajectory shrinks with time.

$$\int e^{-\mathrm{Re}(\alpha)t} |a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)| dt = 2 \cdot \left[|a \cdot \boldsymbol{K}_1(t) + b \cdot \boldsymbol{K}_2(t)| \cdot \frac{e^{-\mathrm{Re}(\alpha)t}}{1-\omega} + \int \boldsymbol{\Psi}(t) \cdot \frac{e^{-\mathrm{Re}(\alpha)t}}{1-\omega} dt\right] \tag{20}$$

From equation (19) and equation (20) we get

$$\Rightarrow \dot{\boldsymbol{u}}(t) = -\left|\frac{1}{\alpha - \beta}\right| \cdot e^{-(1-\omega)t/2} \cdot \int \boldsymbol{\Psi}(t) \cdot e^{(1-\omega)t/2} dt. \tag{21}$$

Now we have $\Psi(t) \geq 0 \ \forall t$ and $e^{(1-\omega)t/2}$ are always positive and increasing function. So $\int \Psi(t) \cdot e^{(1-\omega)t/2} dt$ is an increasing function and it will attain positive values, and $e^{-(1-\omega)t} / |\alpha - \beta|$ is also always positive, and thus $\dot{\boldsymbol{u}}(t) \leq 0 \ \forall t$. Now if we think of the general solution of the swarm then from equation (11) we see that $\boldsymbol{\mu}_g(t)$ tends to zero and it is almost zero after some time. So for high values of *t*, $\boldsymbol{\mu}_p(t)$ is $\boldsymbol{\mu}(t)$ as $\boldsymbol{\mu}_g(t)$ is zero. So the upper bound of $\boldsymbol{\mu}_p(t)$ determines the upper bound of $\boldsymbol{\mu}(t)$. Hence we conclude that the upper bound of the swarm trajectory is a decreasing function of time. By this way we can say that the swarm converges with time.

2.2 Error Dynamics in Generalized gbest PSO: A Comparative View

This section continues the analysis with *gbest* PSO and studies the nature of convergence and diversity of the swarm. A similar analysis leads us to a particular solution:

$$\boldsymbol{\mu}_p(t) = \frac{1}{(D-\alpha)(D-\beta)} (\boldsymbol{\varphi}(t) - c' \cdot \boldsymbol{x}^*) \tag{22}$$

$$\Rightarrow \boldsymbol{\mu}_p(t) = \frac{a}{(D-\alpha)(D-\beta)} \cdot E(rand_1 \cdot \boldsymbol{l}_i(t)) + \frac{1}{2 \cdot (D-\alpha)(D-\beta)} (b \cdot \boldsymbol{g}(t)) - \boldsymbol{x}^*. \tag{23}$$

Now let $E(\boldsymbol{l}_i(t)) = \boldsymbol{L}(t)$ and consequently $\boldsymbol{l}_i(t) = \boldsymbol{L}(t) + \boldsymbol{d}_i(t)$, where $E(\boldsymbol{d}_i(t)) = 0$. So by a similar way that of *lbest* PSO we can show that equation (23) reduces to

$$\boldsymbol{\mu}_p(t) = \frac{1}{2(D-\alpha)(D-\beta)} \cdot (a \cdot \boldsymbol{L}(t) + b \cdot \boldsymbol{g}(t)) - \boldsymbol{x}^* \tag{24}$$

Here by letting $\boldsymbol{L}(t) = \boldsymbol{x}^* + \boldsymbol{K}_3(t)$ and $\boldsymbol{g}(t) = \boldsymbol{x}^* + \boldsymbol{K}_4(t)$, and substituting we can derive,

$$\boldsymbol{\mu}_p(t) = \frac{1}{2(D-\alpha)(D-\beta)} \cdot (a \cdot \boldsymbol{K}_3(t) + b \cdot \boldsymbol{K}_4(t)) \tag{25}$$

$$|\boldsymbol{\mu}(t)| \leq \left| \frac{1}{\alpha - \beta} \right| \cdot e^{\mathrm{Re}(\alpha)t} \cdot \int e^{-\mathrm{Re}(\alpha)t} |a \cdot \boldsymbol{K}_3(t) + b \cdot \boldsymbol{K}_4(t)| dt. \tag{26}$$

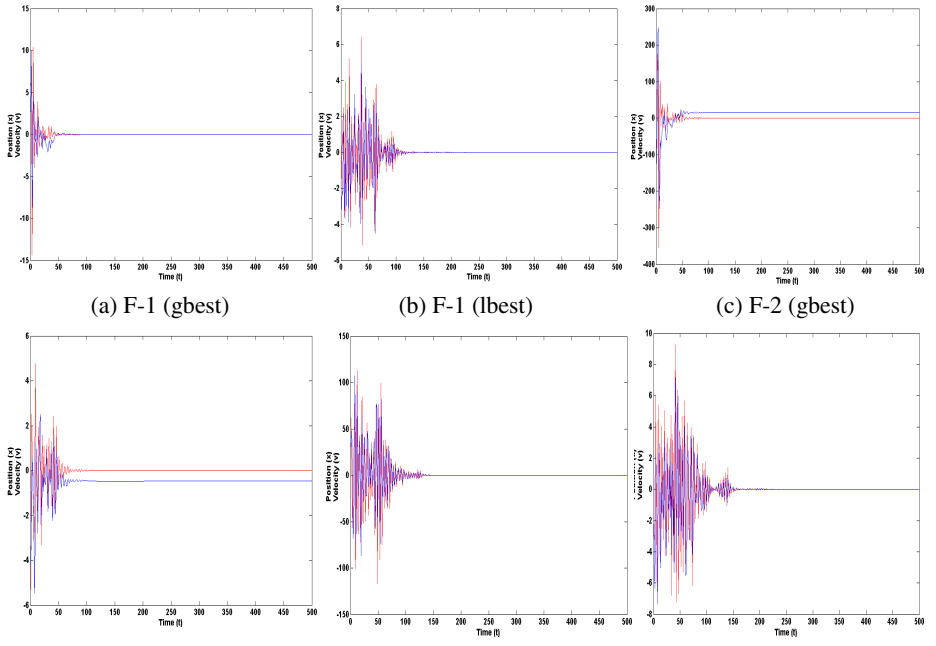
$$|\boldsymbol{\mu}(t)| \leq \left| \frac{1}{\delta} \right| \cdot \frac{\left(a \cdot |\boldsymbol{\mu}(0)| + b \cdot |\boldsymbol{K}_4(t)|_{\max} \right)}{1 - \omega}. \tag{27}$$

We can notice easily that equation (17) and equation (26) are same except in the fact that equation (17) contains $|\boldsymbol{K}_1(t)|$ and $|\boldsymbol{K}_2(t)|$ whereas equation (26) contains $|\boldsymbol{K}_3(t)|$ and $|\boldsymbol{K}_4(t)|$. Here also we can say that, $|\boldsymbol{K}_3(t)|$ and $|\boldsymbol{K}_4(t)|$ are either decreasing function or remains constant. So all the analysis done after equation (17) also hold here and we can infer that the upper bound of the mean swarm trajectory is also decreasing and swarm approaches to the global optimum as times goes. The only difference in the convergence property that is found in equation (26) for *gbest* with the equation (17) is that $|\boldsymbol{K}_4(t)|$ is present in (26) instead of $|\boldsymbol{K}_2(t)|$ as in (17).

Though both $|K_1(t)|$ and $|K_3(t)|$ denotes the same thing, we gave different notations only to distinguish between the models *lbest* and *gbest*. $|K_4(t)|_{\max}$ is the maximum error of the global best particle from the global optimum. So when the particles are in the attraction basin of the optimum, $|K_4(t)|_{\max} \le |g(0) - x^*|$ and hence $|K_4(t)|_{\max} < |\mu(0)|$. So we can infer here that the upper bound of the mean swarm trajectory is lower for *gbest* case than *lbest* case. Larger upper bound of the mean swarm trajectory conveys that the particles are distributed over a large search space, whereas smaller upper bound restricts the particles to be confined in a smaller region. Thus, the *lbest* PSO possesses better explorative power than the *gbest* PSO.

3 Simulation Results

The following plots give the nature of the mean swarm trajectory and mean velocity component with time on some benchmark functions. Here we have plotted the first component of mean swarm trajectory and mean velocity. These graphs successfully demonstrate that diversity introduced in *lbest* PSO is more than *gbest* PSO. In these graphs (Figure 2), red colored graphs denote the first component of mean velocity while the blue colored graph denotes the first component of mean. From all the graphs given in figure 2 we can easily notice that the *lbest* PSO has better oscillatory nature



(a) F-1 (gbest) (b) F-1 (lbest) (c) F-2 (gbest)

(d) F-2 (lbest) (e) F-3 (gbest) (f) F-3 (lbest)

Fig. 2. Variation of the first component of mean velocity and the mean position with time for the three benchmark functions; F-1: Sphere, F-2 Ackley's and F-3 Griewank function

(diversity) than *gbest* best. Again from these graphs one can notice that the amplitude of oscillation decreases that indicates the decrease in the upper boundary of the swarm trajectory as explained in the section following equation (21). The lesser amplitude of oscillation for *gbest* PSO than *lbest* PSO establishes the fact that upper boundary of swarm trajectory is larger for *lbest* PSO than *gbest* PSO.

4 Conclusion

In this paper we have done a comparative analysis of the *lbest* and *gbest* PSO on their convergence and boundary of diversity. In this paper we have successfully shown that the swarm trajectory remains bounded in a certain; so, if the global minimum remains outside this range, it is impossible theoretically to find the global optimum. Also we have shown the upper boundary of the swarm trajectory for *gbest* is smaller than that for *lbest* PSO. This signifies the probability that an optimum will remain within the upper boundary of the mean swarm trajectory is larger for *lbest* than *gbest* PSO.

References

1. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proc. IEEE Int. Conf. Neural Netw. (ICNN)., vol. 4, pp. 942–1948 (November 1995)
2. Venter, G., Sobieski, J.S.: Particle swarm optimization. AIAA Journal 41(8), 1583–1589 (2003)
3. Bratton, D., Kennedy, J.: Defining a standard for particle swarm optimization. In: IEEE Swarm Intelligence Symposium (2007)
4. Engelbrecht, A.P.: Fundamentals of Computational Swarm Intelligence. John Wiley & Sons (2006)
5. Clerc, M.: Particle Swarm Optimization. ISTE Publications (2008)
6. Hendtlass, T.: The particle swarm algorithm. In: Computational Intelligence: A Compendium, pp. 1029–1062 (2008)
7. Ozcan, E., Mohan, C.K.: Analysis of a simple particle swarm optimization system. In: Proc. Intell. Eng. Syst. Through Artificial. Neural Netw. (ANNIE), vol. 8, pp. 253–258 (October 1998)
8. Ozcan, E., Mohan, C.: Particle swarm optimization: surfing the waves. In: Proc. Congr. Evol. Comput., pp. 1939–1944. IEEE Service Center, Piscataway (1999)
9. Clerc, M., Kennedy, J.: The particle swarm - explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. on Evolutionary Computation 6(1), 58–73 (2002)
10. Fernández-Martinez, J.L., Garcia-Gonzalo, E.: Stochastic stability analysis of the linear continuous and discrete PSO models. IEEE Trans. on Evolutionary Computation 15(3), 405–423 (2011)
11. Ghosh, S., Kundu, D., Suresh, K., Das, S., Abraham, A., Panigrahi, B.K., Snášel, V.: On some properties of the lbest topology in particle swarm optimization. In: Ninth International Conference on Hybrid Intelligent Systems (HIS, 2009), Shenyang, China, August 12–14 (2009)

Application of Differential Evolution with Best of Random Mutation Strategy on Asymmetric Location Only Synthesis of Broadside Circular Antenna Array

Sudipta Das[1], Durbadal Mandal[1], Sakti Prasad Ghoshal[2], and Rajib Kar[1]

[1] Department of Electronics and Communication Engineering
[2] Department of Electrical Engineering
National Institute of Technology, Durgapur
Durgapur, Burdwan, India
{sudipta.sit59,durbadal.bittu,spghoshalnitdgp,
rajibkarece}@gmail.com

Abstract. Differential Evolution is a class of Evolutionary Optimization algorithms. Antenna array pattern synthesis for interference suppression without significant loss in gain has become a centre of attraction to many researchers. This paper aims at overall sidelobe performance improvement of broadside uniformly excited circular antenna array by finding near-appropriate locations of elements. For this work, Differential Evolution with Best of Random mutation strategy is utilized.

1 Introduction

An Evolutionary Algorithm is a computer program that fetches the idea of biological evolution strategy to reach a probable numerical solution. It maps biological Selection, Crossover and Mutation procedures into Selection Crossover and Mutation tools.

Differential Evolution (DE) [1] is a type of Evolutionary Algorithm that employs mutation, crossover and then selection to find a probable solution to a problem. Since development of the algorithm, it dragged attraction of researchers and scientists, and thus a numerous variety of this algorithm has been proposed [2]-[4]. Due its speed and simplicity it has drawn concentration of engineers, researchers and scientists, and consequently, DE has found a wide variety of applications [5]-[6].

This paper uses Differential Evolution Algorithm incorporating Best of Random mutation strategy (DEBoR) for circular antenna array [7]-[17] synthesis problem of improvement of sidelobe performance and peak directivity [7].

This paper proceeds as follows: A brief introduction to the RGA in section 2, problem statement in section 3, description of platform specification for program execution and results of simulation in section 4, and finally, conclusion in section 5.

2 The Differential Evolution Algorithm with Best of Random Mutation Strategy

Differential Evolution with Best of Random Mutation strategy is described in [4], [5], [17]. Thus, due to limitation of page, it is not described here in detail.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 33–40, 2012.

3 The Problem Statement

Circular Antenna Array is probably the most versatile kind of planar array, yet it is simple. This paper considers a broadside circular antenna array structure lying on *x*-*y* plane without central element feeding. For simplicity and versatility of application, the far-field space pattern is taken for modification. Only the generalized array factor suffices to describe the far-field space pattern. For a uniformly excited circular array as considered in this paper the array factor is given below [7].

$$AF(\theta,\phi) = \sum_{n=1}^{N} e^{jka\sin\theta\cos(\phi-\phi_n)} \tag{1}$$

where

N = the number of elements on the structure;

$k = 2\pi/\lambda$, λ being the wavelength of operation;

a is the radius of the array aperture;

θ is the azimuth angle;

ϕ is the elevation angle; and

ϕ_n is the angular location of n^{th} element from *x*-axis on *x*-*y* plane.

A generalized circular antenna array structure looks as shown in Fig. 1:

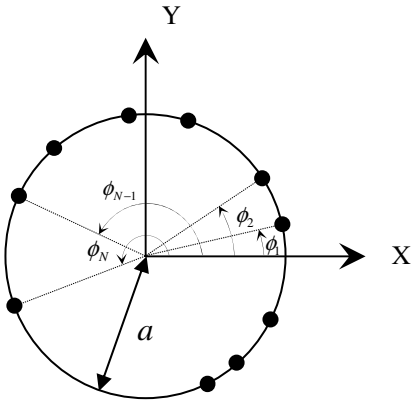


Fig. 1. Geometry of *N*-element asymmetric circular array of isotropic radiators placed on *x*-*y* plane(Top View)

A uniform circular antenna array has uniform current and location profile maintained over the array aperture. It has high peak sidelobe causing the maximum interference in the received signal from unintended direction. The goal in the present case is taken as the significant reduction of the relative maximum interference level (relative peak sidelobe level) or SLL of antenna array's response pattern. The solution

is required to depart from the initial pattern in terms of the SLL only. The First Null Beamwidth (BWFN) is not required to spread significantly; otherwise it will introduce external noise through the main lobe. Further, as the nulls in the radiation profile denote negligible interference points, the solution is required to retain the nulls of the initial pattern as far as possible. The *x*-*z* plane $\left(\phi = 0^{o}\right)$ is considered for pattern optimization.

Only the angular locations ($\phi_n, n \in [1, N]$) profile of the elements on the circular aperture are taken as the variables. The radius of the initial aperture is chosen so that arc distance of any neighbor element-pair lies always in the range $[0.5\lambda, \lambda]$ so as to avoid mutual coupling and possibility of grating lobe appearance in the radiation pattern. Thus $a = 3\lambda / \{8\sin(\pi / N)\}$ is chosen [8]. With large number of elements, the above relation changes to

$$a = \frac{3N}{8\pi}\lambda \tag{2}$$

Uniform array design selected this way, has the uniform inter-element arc distance of 0.75λ. Directivity of each array is numerically calculated using the method as specified by [7] that is found correct result up to two decimal places for this problem.

The cost function (CF) is designed to make the problem a minimization problem. It is designed in such a way that reduction of *SLL* in both upper and lower bands without significant increment in the *BWFN* and loss of peak directivity causes lowering of cost function. In the present case, CF is designed as follows:

$$CF = \frac{\left\{ \begin{array}{l} \dfrac{SLL_{initial}}{SLL_{current}} + \sum_{i} \left| AF(\theta_i, \phi_n) \right|^2 \\ + \left| BWFN_{initial} - BWFN_{current} \right| \end{array} \right\}}{D_0} \tag{3}$$

where,

$SLL_{initial}$ and $SLL_{current}$ are the relative peak sidelobe levels of the initial (uniform) and current geometry.

D_0 = The Peak Directivity of the array as found in current iteration in absolute scale;

If the current iteration yields the lower *SLL*, the 1st term will get reduced and hence, CF will be reduced. So, the first term of numerator in (3) is used for minimizing the *SLL*.

The second term in the numerator of (3) [9] is used to reduce overall sidelobe level in each iteration by imposing nulls everywhere outside the main beam.

In (3), the two beamwidths, $BWFN_{current}$ and $BWFN_{initial}$ basically refer to the computed first null beamwidths in radian for the non-uniform angular positions case and for uniform angular positions (initial case), respectively. So, this difference term of (3) [10]-[12] restricts the spreading of the main beam as far as possible.

With higher value of D_0, CF in (3) is minimized. The DEBoR employed for optimizing the angular locations of the elements results in the minimization of CF.

4 Platform Description and Result Analysis

4.1 Platform Specification

Since all the programs are more or less platform dependent, the platform specification is necessary for conducting a test and commenting on the results. The programming was written in MATLAB language using MATLAB 7.5 on core (TM) 2 duo processor, 2.99 GHz with 1 GB RAM.

4.2 Simulation Results

Parameters for the DEBoR are set after many test runs. Population size of vectors is 120. Maximum number of iterations is 400. Best scaling Factor for mutation is selected as 0.49. Best Binary crossover is chosen with Crossover Probability as 0.35.

Optimization simulation is run with the location vectors for constructing the radiation pattern that provides the lowest *SLL* without hampering D_0. No progressive inter-element phasing is assumed. The excitation profile is assumed to be uniform, and the radius is expressed in terms of λ. ϕ_n is expressed in degrees (^{o}). D_0 is dimensionless.

The simulations are carried out for the sets of 16-, 18- and 20-element circular arrays. Best of five independent runs are used in this paper. Results are correct up to two decimal places.

Table 1 records the parameters SLL, BWFN and D_0 as found from the uniform array, *i.e.*, the initial design.

Table 2 records the results for location only synthesis without any constraint on the arc distance element, *i.e.*, arc distances may attain values beyond the limit $[0.5\lambda, \lambda]$. It records corresponding optimal locations and resultant SLL, BWFN and D_0 for all sets.

Table 3 records the parameters for location only synthesis with the constraint on the arc distances within $[0.5\lambda, \lambda]$. It records optimal angular locations, SLL, BWFN and D_0 for the location only synthesis for all sets.

Fig. 2 depicts the radiation profiles of the 20-elements antenna arrays with uniform, unconstrained and constrained arc distances over the array aperture. Convergence profile of DEBoR for this case is plotted in Fig. 3.

Table 1. Parameters for Uniform Circular Array Sets

No. of Elements	Radius a/λ	Initial *SLL* (dB)	Initial *BWFN* (°)	Initial D_0
16	1.91	-7.90	23.75	28.53
18	2.15	-7.90	20.75	23.67
20	2.39	-7.90	18.25	34.32

Table 2. Parameters for Unconstrained Location Only Synthesis of Circular Array Sets

No. of Elements	Optimal Angular Locations (°)	Final *SLL* (dB)	Final *BWFN* (°)	Final D_0
16	30.00 60.00 80.47 93.97 110.85 140.85 170.85 184.35 214.35 231.86 245.36 258.86 272.36 285.86 299.36 327.84	-14.11	29.30	27.10
18	26.67 43.03 55.03 73.94 85.94 97.94 109.94 121.94 148.61 174.77 201.44 228.11 240.11 252.11 264.11 276.11 288.11 302.16	-15.82	28.30	18.37
20	10.80 33.43 57.43 74.38 87.96 98.87 109.67 120.47 131.27 155.27 174.04 193.37 217.37 235.72 251.86 262.66 273.46 284.26 295.056 319.06	-15.36	23.70	32.79

Table 3. Parameters for Constrained Location Only Synthesis of Circular Array Sets

No. of Elements	Optimal Angular Locations (°)	Final *SLL* (dB)	Final *BWFN* (°)	Final D_0
16	30.00 53.45 71.42 86.42 101.42 116.42 135.50 165.50 195.50 225.50 254.01 269.43 285.00 300.00 330.00 360.00	-13.46	27.25	33.16
18	26.67 53.33 67.22 81.10 94.99 109.73 136.40 163.06 176.95 203.61 228.53 245.95 259.84 273.72 287.60 306.67 333.33 360.00	-12.77	23.75	23.26
20	24.00 43.58 67.58 79.68 91.77 103.86 119.97 139.08 163.08 187.08 211.08 235.08 247.17 260.55 272.65 284.74 296.83 312.00 336.00 360.00	-13.38	21.75	34.68

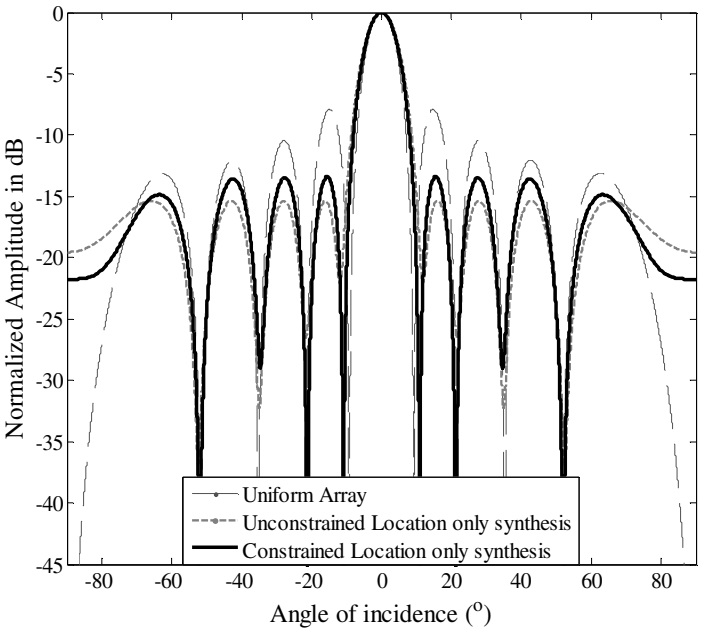


Fig. 2. Radiation Patterns for the 20-element Circular Antenna Arrays optimized for angular locations

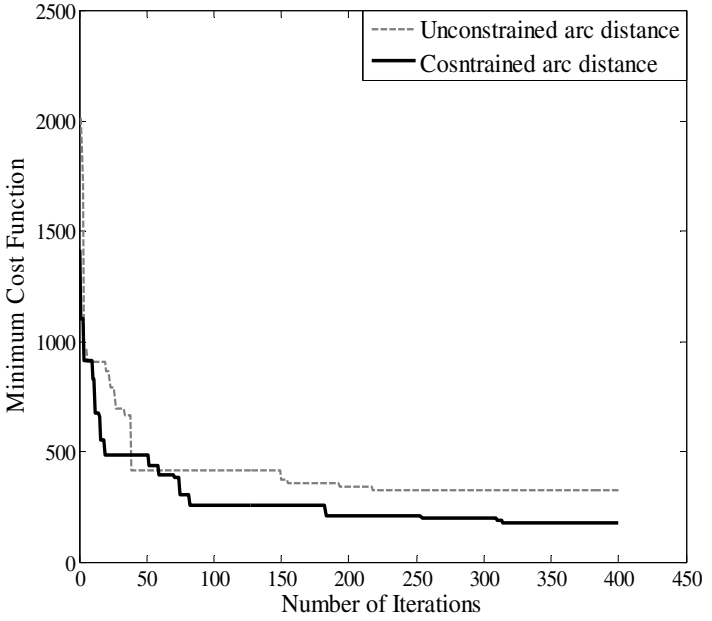


Fig. 3. Convergence Profile of DEBoR for pattern optimizing of the 20- element circular antenna arrays by angular locations

4.3 Analysis of Results and Discussion

The Tables reveal that the radiation characteristics are dependent on the element locations on the circular structure. Good *SLL* lowering is possible for such arrays without the increase in size. Table 2 and Table 3 show that all sets, simulations for both unconstrained and constrained arc distances improve the *SLL* performance as compared to corresponding uniform structures as in Table 1, but, constrained arc distance approach leads the unconstrained one in terms of preserving *BWFN* and Nulls and improving D_0 as far as possible. Consequently, constrained arc distance approach suffers from poorer *SLL* performance. Fig. 2 compares the 20-element structures as optimized and tabulated in Table 2 and Table 3. From these Tables it can be found that except for 18-element set, the controlled arc distance approach has improved the peak directivity to a good extent. For 18-element set the directivity is hampered a little. Whereas, all sets for unconstrained approach are weaker in preserving peak directivity. Minimum and Maximum arc distances for 20-element set for unconstrained approach can be found as 0.45λ and 2.16λ, whereas, for constrained approach has provided 0.50λ and λ.

The linear distance and the arc distance of any two elements of circular array are related to each other in terms of the angular separation and the radius [7, 8]. The inter-element linear distance or inter-element arc distance (for large number of elements) of an array contributes significantly to the pattern. Too nearby placed elements cause inter-element coupling that ultimately results in thickening of the main beam, thus lowering the directivity [15], [16]. Again, if the elements are placed with a large inter-element distance, risk of construction of grating lobe increases. This leads to deterioration of the *SLL*.

Fig. 3 compares the convergence profiles of DEBoR for constrained and unconstrained 20-element arrays. It denotes that the solution update for 20-element circular antenna array was complete within 230 iterations for unconstrained approach, while for constrained approach it took further steps and about 320 iterations to get it's near optimal solution parameters.

It may be interpreted from continuously reducing CF that radiation patterns are getting continuously improved as iteration progresses for both cases. Further, constrained arc distance approach has yielded comparatively lower CF and more improvement in overall *SLL* performance.

5 Conclusions

Choice of inter-element arc distance is important for improving the antenna performance. This work reveals that constrained inter-element arc distance in circular antenna array results in lower BWFN, deeper Nulls and higher peak directivity, but less SLL, as compared to the case of un-constrained inter-element arc distance. Differential Evolution with Best of Random Mutation strategy has made is able to make a good approximation of the required locations of the elements of the circular array without hampering the array size.

References

1. Storn, R., Price, K.: Differential Evolution-A simple & Efficient Adaptive Scheme for Global Optimization over Continuous Spaces, International Computer Science Institute, Bekerly, CA, Technical report-TR-95-102 (1995)
2. Omran, M.G.H., Salman, A., Engelbrecht, A.P.: Self-adaptive Differential Evolution. In: Hao, Y., Liu, J., Wang, Y.-P., Cheung, Y.-M., Yin, H., Jiao, L., Ma, J., Jiao, Y.-C. (eds.) CIS 2005. LNCS (LNAI), vol. 3801, pp. 192–199. Springer, Heidelberg (2005)
3. Das, S., Abraham, A., Chakraborty, U.K., Konar, A.: Differential Evolution Using a Neighborhood-based Mutation Operator. IEEE Transactions on Evolutionary Computation 13(3), 526–553 (2009)
4. Lin, C., Quing, A.: Synthesis of Unequally Spaced Antenna Arrays by a New Differential Evolution Algorithm. International Journal on Communication Networks Information Security (IJCNIS) 1(1), 20–25 (2009)
5. Roscca, P., Oliveri, G., Massa, A.: Differential Evolution as Applied to Electromagnetics. IEEE Antennas & Propagation Magazine 53(1), 38–49 (2011)
6. Kurup, D.G., Himdi, M., Rydberg, A.: Synthesis of uniform amplitude unequally spaced antenna arrays using the differential evolution algorithm. IEEE Trans. Antennas Propagat. 51(9), 2210–2217 (2003)
7. Balanis, C.A.: Antenna theory analysis and design, 3rd edn. John Willey and Son's Inc., New York (2005)
8. Das, S., Mandal, D.: Synthesis of Broadside Uniform Circular Antenna Array with low on-Surface Scanning. In: The Proceedings of IEEE Indian Antenna Week 2011, Kolkata, pp. 1–4 (December 2011)
9. Khodier, M.M., Christodoulou, C.G.: Linear Array Geometry Synthesis With Minimum Sidelobe Level and Null Control Using Particle Swarm Optimization. IEEE Trans. Antennas Propagat. 53(8), 2674–2679 (2005)
10. Das, S., Mandal, D.: Low Current Tapered Non-Uniform Linear Array Synthesis with Decreasing Sidelobe Using NPSO. In: The Proceedings of IEEE Indian Antenna Week 2011, Kolkata, pp. 10–14 (December 2011)
11. Mandal, D., Ghoshal, S.K., Das, S., Bhattacharjee, S., Bhattacharjee, A.K.: Improvement of Radiation Pattern for Linear Antenna Arrays Using Genetic Algorithm. In: Proceedings of the 2010 IEEE International Conference on Recent Trends in Information, Telecommunication and Computing, pp. 126–129 (2010)
12. Das, S., Bhattacharjee, S., Mandal, D.: Improvement of Far Field Radiation Pattern of Linear Array Antenna Using Genetic Algorithm. ICTACT Journal of Communication Technology 1(1), 23–31 (2010)
13. Das, S., Bhattacharjee, S., Mandal, D.: Sidelobe Level and Null Control of Asymmetric Linear Antenna Array using Genetic Algorithm. International Journal of Artificial Intelligence and Computational Research 2(1), 7–12 (2010)
14. Das, S., Bhattacherjee, S., Mandal, D.: Optimal Angular Locations of Elements for Asymmetric Circular Array Antennas. In: IEEE Symposium on Industrial Electronics & Applications (ISIEA 2010), Penang, Malaysia, pp. 681–685 (October 2010)
15. Elliot, R.S.: Beamwidth and Directivity of Large Scanning Arrays, First of Two Parts. The Microwave Journal, 53–60 (January 1963)
16. Elliot, R.S.: Beamwidth and Directivity of Large Scanning Arrays, Last of Two Parts. The Microwave Journal, 74–82 (January 1964)
17. Das, S., Bhattacharya, M., Sen, A., Mandal, D.: Linear Antenna Array synthesis with Decreasing Sidelobe and Narrow Beamwidth. ACEEE International Journal on Communication 3(1), 10–14 (2012)

Performance Evaluation of Bacterial Foraging Optimization Algorithm for the Early Diagnosis and Tracking of Alzheimer's Disease

Tinu Varghese[1], R. Sheela Kumari[2], P.S. Mathuranath[3], and N. Albert Singh[4]

[1] Noorul Islam University, Kumara Coil, Thuckalay, Tamilnadu
tinuannevarghese@gmail.com
[2] Sree Chitra Tirunal Institute for Medical Science and Technology, Trivandrum, Kerala
sheela82nair@gmail.com
[3] Department of Neurology, Sree Chitra Tirunal Institute
for Medical Science and Technology, Trivandrum, Kerala
mathu@sctimst.ac.in
[4] Noorul Islam University, Kumara coil, Thuckalay, Tamilnadu
albertsingh@rediffmail.com

Abstract. Alzheimer's disease (AD) is a complex progressive brain disorder. The concept of Mild Cognitive impairment (MCI) is considered as a subtle but measurable disorder that is greater than the normal aging controls. In this paper we explored the ability of specifically designed and trained Artificial Neural Network (ANN) combined with bacterial foraging optimization (BFO) algorithm, to discriminate between the MRI of patients with AD and their age matched controls. The proposed approach employs feature extraction using Gabor filter and discrimination based on BFO tuned ANN. Due to the progressive nature of AD, This study aims to identify the structural characteristics at baseline and over a period of two years that could serve as accurate predictors of future development of MCI in the NCI and AD in the MCI patients .We report the results of the classification accuracies on both training and test images are up to 92%.

Keywords: Alzheimer's disease, Mild Cognitive Impairment, Artificial Neural network, Bacterial Foraging Optimization, Magnetic Resonance Imaging.

1 Introduction

Dementia is a chronic syndrome, characterized by a progressive, global deterioration in intellect including memory, learning, orientation language, comprehension and judgment due to disease of the brain[1]. In 2010 dementia India reports estimated that over 3.7 million people are affected by dementia in our country. Alzheimer's disease (AD) is the commonest type of dementia .It is a progressive and irreversible disease. It usually occurs after the age of 65. Neurofibrillary tangles and amyloid plaques are the histopathological hallmark of AD and are associated with neuronal loss and brain volume reductions[2]. The concept of MCI is a midway between normal aging and

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 41–48, 2012.

very early AD. It provides a window for intervention in the preclinical stage of dementia and thereby for possible prevention of dementia [3]. Most commonly used diagnosis of AD is made by clinical, neuropsychological and neuroimaging assessments[4]. Brain pathologies that cause to AD all start well in advance of the onset of clinical signs and symptoms[5]. Early diagnosis of AD allows time to plan for the future and to treat patients before marked deterioration occurs. MR imaging technique must be consistently differentiates AD from normal aging in individual scans[6].

This work presents an investigation of the potential of BFO for classification of registered magnetic resonance images of the human brain. The main aim of this work is to segment regions of interest in MRI images which consist of groups of similar cells indicating some form of features in the human brain. This approach permits the segmentation of image volumes based on training and test sets selected on a single slice.We present BFO based segmentation method for extracting the features using Gabor filter. The Gabor filter features embedded in 3D medical images. Finally, the segmentation results obtained both with the Gabor filter feature selection and the BFO classifiers.

This paper presents current work on back propagation neural network simulate the chemotactic behavior of bacterial foraging algorithm and its application to optimize the parameters of a neural network. This segmentation which results in the subdivision of an entire image into its constituent regions such as Grey matter (GM), White Matter (WM) and Cerebrospinal fluid (CSF). In research, different machine learning approaches such as neural network and fuzzy are used for the classification of different stages of AD. In the preclinical stage of AD, atrophy can be originated in the hippocampus, temporal lobe and entorhinal cortex[7].Due to the progressive nature of AD, the longitudinal MRI should be able to detect the progressive structural changes occurring within an individual and improve the diagnostic accuracy.

The purpose of our longitudinal study is to examine the structural characteristics of certain specific brain regions at baseline and structural changes in them overtime that could serve as accurate predictors of future development of MCI in the NCI and AD in the MCI patients. Our first aim was to clarify the differences between cerebral atrophy due to normal aging, MCI and AD. The second objective is to identify the atrophy exists before the onset of dementia or during the MCI stage.

2 Materials and Methods

2.1 Samples and Recruitment

All Participants in this study were selected in the Sree Chitra Tirunal Institute for Medical Science and Technology (SCTIMST), Trivandrum, Kerala dementia clinic. The subjects were underwent clinical examination, detailed neuropsychological and neuroimaging evaluation at baseline and after one year.30 MCI patients were selected with a clinical diagnosis of probable MCI according to the NINCDS/A-DRDA criteria. The patients ranged in age from 52 to 75 years and the average score of Mini Mental State Examination (MMSE) was 23 ±3. 20 healthy volunteers group matched to patients on age and education and MMSE score of 28±2.

2.2 MRI Acquisition and Preprocessing

Whole brain MRI scans were obtained on Siemens Magnetom-Avanto SQ engine, 1.5T MR Scanner. Whole brain volume was acquired by the 3D flash spoiled gradient echo sequence using standard parameters.TR=11msec, TE=4.95, flip angle=150, slice thickness=1mm, matrix=256x256, 112 axial plane images were made to cover the whole brain. The images were post processed in the fully equipped Brain mapping unit of Cognitive and Behavior Neurology Section (CBNS).

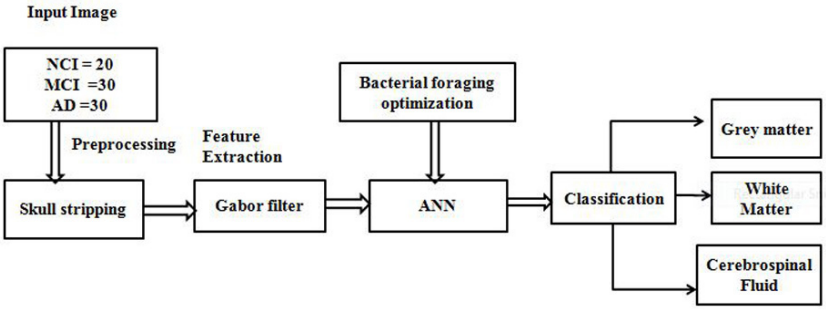


Fig. 1. Methodology of our proposed algorithm

Our method consists of three stages as shown in fig.1 skull stripping, feature extraction and back propagation based neural network classifications. Data pre-processing technique have a major role in MR brain imaging applications. Skull stripping removes the non cerebral tissues such as skull, scalp, vein etc from the original MR images. A single hidden layer BPNN is adopted with sigmoid neurons in the hidden layer and linear neuron in the output layer.MR brain image segmentation which results in the subdivision of an entire image into its constituent regions such as GM, WM and CSF.

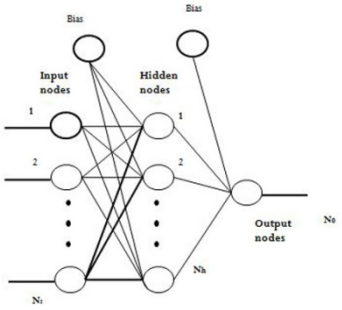


Fig. 2. Simple Architecture of ANN

2.3 Texture Feature Extractions

The purpose of texture analysis is to classify or segment different texture regions in an image. Specifically this work deals with 2D Gabor filter texture segmentation of

an image [8]. In this paper, we propose to exploit the concept of Gabor filter texture segmentation to segment medical images containing various anatomical structures that are belonging to MR imaging modalities.

2.4 BFO Based Training of ANN

The BFO Algorithm is a swarm intelligence artificial neural network technique which simulates the foraging policies of the E. coli bacteria as a distributed optimization process. This swarm based algorithm proposed in 2002 by Kevin M Passino[9]. Chemotaxis behavior is the key idea of bacterial foraging .The E.coli bacteria and salmonella locomotion is achieved by a set of tensile flagella. Swimming and tumbling are the basic operations performed by a bacterium at the time of foraging. The E.coli bacteria rotate their flagella in counter clockwise directions to move forward rotate also called as swimming. While the movement of bacteria in clockwise directions the flagellum causes the bacterium randomly tumble itself in a new directions and then swims again[10]. The alternate processing of swimming and tumbling enables the bacterium search for nutrients in random directions. The BFO algorithms consists of four principal mechanisms, namely, Chemotaxis, swarming, reproduction, elimination-dispersal.

Chemotaxis. Chemotaxis process simulates the locomotion of the E.coli bacteria through swimming and tumbling via flagella. Consider θ^i (j, k, l) represents i^{th} bacterium at the j^{th} chemotactic, k^{th} reproductive and l^{th} elimination and dispersal .In the computational chemotactic processing step, the movement of the bacterium can be represented as

$$\theta^i(j+1,k,l) = \theta^i(j,k,l)+\mathrm{C}(\mathrm{i}) \times \ \Delta(i)/\sqrt{\Delta^T(i)\Delta(i)} \tag{1}$$

Swarming. It is desired that the bacterium has searched the optimum path of food should try to attract other bacteria so that they reached the desired place more rapidly. Swarming makes the bacteria aggregate in to groups and hence move as concentric pattern of groups with high bacterial density. The cell to cell signaling in E.coli swarm may be represented as

$$\begin{aligned} J_{cc}(\theta, P(j,k,l) &= \sum_{i=1}^{S} J_{cc}(\theta, \theta^i(j,k,l) \\ &= \sum_{i=1}^{S} \left[-\mathrm{d}_{\mathrm{attract}} \exp\left(\mathrm{w}_{\mathrm{attract}} \sum_{m=1}^{p} (\theta_{\mathrm{m}} - \theta^{\mathrm{i}}{}_{\mathrm{m}})^2\right)\right] \\ &+ \sum_{i=1}^{S} \left[\mathrm{h}_{\mathrm{repellant}} \exp\left(-\mathrm{w}_{\mathrm{repellant}} \sum_{m=1}^{p} (\theta_{\mathrm{m}} - \theta^{\mathrm{i}}{}_{\mathrm{m}})^2\right)\right] \end{aligned} \tag{2}$$

where Jcc(θ,P (j,k,l)) is the objective function , S is the total number of bacteria ,p is the number of variables to be optimized .

Reproduction. The least healthy bacteria considering half of the bacterial populations are ultimately eliminated. Remaining the other healthiest bacteria split in to two bacteria which are placed in the location of the least healthy bacteria[11]. Ultimately this makes the population of bacteria is constant. This step can be evaluated by the following equations

$$J_{health}^{i} = \sum_{j=1}^{N_{c+1}} \mathrm{J(i, j, k, l)} \tag{3}$$

Elimination and Dispersal. In this algorithm some bacteria to get eliminated and dispersal process may place bacteria near good food sources. Elimination and dispersion steps in the BFO ensure that the bacteria do not get trapped in to a local optimum instead of the global optima. [12].

2.4.1 Operation of Bacterial Foraging

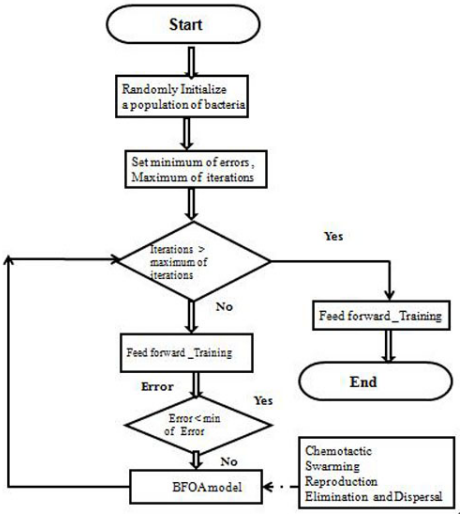


Fig. 3. Training of ANN model with BFO

3 Experimental Results and Discussions

In the proposed BFO tuned ANN method, ANN is applied for evolving fully connected feedforward neural network and is optimized with best network architecture by optimizing the number of neurons in the hidden layer (Nh), the learning rate (Lr) and the momentum factor (Mc). The range of the optimization process is defined by two range arrays

$$R\ min = \{Nh_{min,} Lr_{min,} Mc_{min,}\},\ R\ max = \{Nh_{max,} Lr_{max,} Mc_{max,}\} \quad (4)$$

Let f be the activation function and is defined as the sum of the weighted input plus the bias and is represented as

$$y_k^p = f(s_k^p),\ s_k^p = \sum_j w_{j,k}\ y_j^p + \theta_k \quad (5)$$

y_k^p is the output of the kth neuron when a pattern p is fed , $w_{j.k}$ is the weight from the jth neuron and θ_k is the bias value of the kth neuron in the hidden layer and it is defined by tangent activation function,

$$\tanh(x) = e^x - e^{-x} / e^x + e^{-x} \quad (6)$$

The fitness functions sought for optimal training is the Mean Square Error (MSE) formulated as

$$MSE = \sum_{p \in T} \sum_{k=1}^{No} (t_k^p - y_k^{p,o})^2 \quad (7)$$

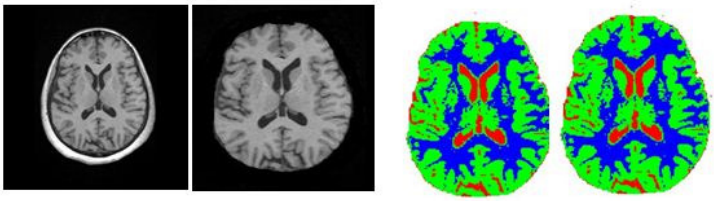
Where t_k^p is the desired output, $y_k^{p,o}$ is the actual output from the k^{th} neuron in the output layer 'o', for the pattern p in the training set. With the framed fitness function the BFOANN algorithm automatically evolve a best solution.

For the classification experiments, the training set consists of 1025 feature set and is used for training. The optimization BFOANN classifier is performed with the learning rate and the momentum constant varied from 0 to 1 and the hidden neurons varied from 31 to 200.Using the proposed algorithm an optimized ANN is achieved with Nh=152,Lr=0.2571,and Mc=0.8963.The parameters are optimally selected thereby adjusting the connection weights and analyze the classification performance depends on the Nh , Lr and Mc.

BFO Parameters.

Dimensions of the search space	**3**
Number of bacteria used for searching the total region (S)	**20**
The number of iterations taken in a chemotactic steps	**20**
Swimming length (N_s)	**4**
The number of reproduction (N_{re})	**4**
The number of elimination and dispersal events (N_{ed})	**2**
Elimination and dispersion probability (P_{ed})	**0.25**

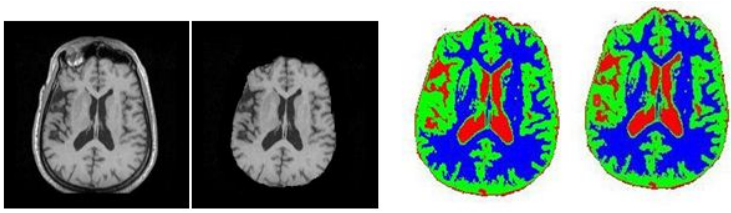
3.1 Base Line vs. Follow Up Evaluation of Controls (NCI)



(a) (b) (c) (d)

Fig. 4. Segmentation of a baseline and follow up MR brain images (NCI): (a) Original baseline image, (b) skull stripped baseline image, Segmentation of (c) baseline and (d) follow up images using ANN BFO algorithm

3.2 Base Line vs. Follow Up Evaluation of MCI Patients



(a) (b) (c) (d)

Fig. 5. Segmentation of a baseline and follow up MR brain images (MCI): (a) Original baseline image, (b) skull stripped baseline image, Segmentation of (c) baseline and (d) follow up images using ANN BFO algorithm

3.3 Normal Aging versus AD Classifications Using BFO

(a) NCI (b) MCI (c) AD

Fig. 6. Segmentation of MR brain images using ANN BFO algorithm: (a) NCI, (b) MCI (c) AD

From the validation results, BFOANN has shown good performance in improving ANN learning in terms of correct classification percentage. In this paper we have

developed a novel classifier to distinguish normal and abnormal brain MRIs. The results show that our method obtained 92% classification accuracy on both training and test images.

4 Conclusions

In this paper, we have developed a computer aided diagnosis system for assisting the early detection of AD. We conclude that the discrimination between the normal aging and AD appears to be a trend towards accelerated volume loss in grey matter and other regions of the brain. BFO ANN Segmentation algorithm can be used to distinguish normal and abnormal brain MRIs. In addition predictive values of MCI within 12 months and after 12 months are significantly different. The method obtained up to 92% classification accuracy on both training and test images of the selected data set.

References

1. Burns, A., Michael, Z.: Mild cognitive impairment in older people. Lancet 360, 1963–1965 (2002)
2. Richard, J.P., Anne, M.F., David, M.H.: Multimodal Technique for diagonosis and progonosis of AD. Nature 461 (2009)
3. Mathuranath, P.S., Mathew, R.: Role of subjective memory complaints in defing MCI. Neurobiology of Aging 25, 74–79 (2004)
4. Barber, R.C.: Biomarkers for early detection of Alzheimers disease. JAOA 8, 110–119 (2010)
5. Barbro, R., Monica, N., Helle, W.: Investigating poor insight in AD: A survey research approaches. Dementia 6, 44–61 (2007)
6. Rahul, S.D., Howard, J.C.: Automated MRI measures identify individuals with MCI and AD. Brain 132, 2048–2057 (2009)
7. Devanand, D.P., Bansal, R., Liu, J.: MRI hippocampal and entorhinal cortex mapping in predicting conversion to AD. Neuroimage 60, 1622–1629 (2012)
8. Rajaei, A., Rangarajan, L.: Medical image segmentation using linear combination of gabor filtered images. International Journal of Machine Intelligence 3, 212–216 (2011)
9. Passino, K.M.: Biomimicry of Bacterial foraging for distributed optimization and Control. IEEE Control Systems Magazine, 52–67 (2002)
10. Das, S., Biswas, A., Dasgupta, S., Abraham, A.: Bacterial Foraging Optimization Algorithm: Theoretical Foundations, Analysis, and Applications. Foundations of Computational Intelligence Volume 3. SCI, vol. 203, pp. 23–55. Springer, Heidelberg (2009)
11. Passino, K.M.: Bacterial Foraging Optimization. International Journal of Swarm Intelligence Research 1, 1–16 (2010)
12. Maitra, M., Amitava: A noval technique for multilevel optimal magnetic resonance brain image thresholding using BFO. Measurement 10, 1124–1134 (2008)
13. Al-Hadi, I.A.A., Hashim, S.Z.M., Shamsuddin, S.M.H.: BFO algorithm for neural network learning enhancement. In: Int. Conference on Hybrid Intelligent Systoms, vol. 11, pp. 200–205 (2011)

Multi-sensor Satellite Image Analysis Using Niche Genetic Algorithm for Flood Assessment

J. Senthilnath[1,*], P.B. Shreyas[2], Ritwik Rajendra[2], S.N. Omkar[1], V. Mani[1], and P.G. Diwakar[3]

[1] Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India
[2] Department of Electronics and Communication Engineering, National Institute of Technology Karnataka, Surathkal, India
[3] Earth Observation System, ISRO Headquarters, Bangalore, India
snrj@aero.iisc.ernet.in

Abstract. In this paper, cluster splitting and merging algorithms are used for flood assessment using LISS-III (before flood) and SAR (during flood) images. Bayesian Information Criteria (BIC) is used to determine the optimal number of clusters. Keeping this constraint, the cluster centers are generated using the cluster splitting techniques, namely Mean Shift Clustering (MSC), and Niche Genetic Algorithm (NGA). The merging method is used to group the data points into their respective classes, using the cluster centers obtained from the above techniques. These techniques are applied on the LISS-III and SAR image. Further, the resultant images are overlaid to analyze the extent of the flood in individual land classes. A performance comparison of these techniques (MSC and NGA) is presented. From the results obtained, we deduce that the NGA is efficient.

1 Introduction

Floods cause extensive loss of life, land and property, and hence it is imperative that an effective flood assessment model be developed to help gather information about the occurrence and damage caused by floods. Present methods employ satellite image classification to extract flood prone regions [1].

Optical sensors of the satellites are used to obtain the image of the region prior to the flood. But such sensors are dependent on illumination by the sun, and there is also a possibility of misinterpretation of land cover map under the cloudy weather conditions [2]. However, Synthetic Aperture Radar (SAR) images have the capability of distinguishing between the land cover even during the unfavorable weather conditions because of its active sensor system which uses dipolar effect to acquire the image [3]. Hence, SAR images are used to analyze the land during flood. The optical image is classified according to the various distinct types of land cover [4] and SAR images help to determine the flooded and unflooded regions [5].

* Corresponding author.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 49–56, 2012.

Different methods have been developed to cluster data sets by splitting and merging. Broadly, they can be classified into parametric and non-parametric methods. In parametric methods such as K-means [6], a prior assumption of the number of clusters is required. In non-parametric methods such as Mean Shift Clustering (MSC) (which use single point for locating local maxima) [7], [8], no prior assumptions is made on the number of clusters.

Recently, researchers are interested in locating multiple local optima of a given multi-modal function in a *d*–dimensional search space. For this purpose nature inspired techniques [9] are used. In Niche Genetic Algorithm (NGA) [10], Genetic Algorithm [11] is modified by using niching technique [12] to locate the multiple modes within the search space. Brits et al. [13] developed Niche Particle Swarm Optimization (NPSO), which is a variant of Particle Swarm Optimization (PSO) [14], [15]. Many researchers have successfully applied niche technique to obtain local optima [16], [17].

In this paper, we use cluster splitting and merging techniques namely MSC and NGA, for flood assessment using satellite images. The images used are LISS-III (before flood) and SAR (during flood). LISS-III image is used to analyze the condition prior to the flood. On the other hand, SAR image is used to assess condition during the flood. We classify these images by using cluster splitting and merging techniques. The main challenge here is to optimally split the cluster centers and group (merge) the data set into their respective classes. The care has been taken to split the complex large data set into a number of cluster centers by satisfying Bayesian Information Criteria (BIC) [18], which is commonly used in model selection. We combine the classified LISS-III and SAR images to estimate the extent of land cover affected by the flood. A comparative study of these techniques is done to analyze their performance.

2 Methodology

Clustering involves sub-division of the data set into clusters based on some similarity metrics. In this study, MSC and NGA algorithms are applied. These techniques make use of kernel functions for locating maxima for a given set of discrete data points.

2.1 Splitting Methodology

For large data sets, it is difficult to determine the number of clusters required, as it depends on the distribution of the given data. Bayesian Information Criteria (BIC) is a model fitting approach, which provides the optimal number of clusters. The splitting of data set using BIC into number of clusters is given by

$$BIC = L(\theta) - 0.5*k_j*\log(n) \tag{1}$$

where $L(\theta)$ is the log-likelihood measure for the data set, k_j is the number of free parameters for the specific number of clusters and n is the number of data points for a given data set.

Mean Shift Clustering

Let us have n data points for MSC. Each data point contains d-dimensional feature space given by $x_1, x_2, x_3 \ldots x_n$. A Gaussian kernel is used to average each data point with its neighbouring points. The kernel is given by

$$K(r) = e^{-\|r\|^2} \tag{2}$$

where r is the Euclidean distance between any two data points. The points in the d-dimensional space are treated as a probability density function. This implies if a point is a local maxima then it indicates dense region. A gradient ascent procedure is performed on the local estimated density until it converges, and the modes of the distribution are given by the stationary points. The mean shift for a point x_i is given by

$$m(x_i) = \frac{\sum_{j=1}^{n} x_i g\left(\left\|\frac{x - x_i}{h}\right\|^2\right)}{\sum_{j=1}^{n} g\left(\left\|\frac{x - x_i}{h}\right\|^2\right)} \tag{3}$$

where h is the bandwidth (controls density within the kernel) and $g=-k'(x)$, x is the mean of given n data points. A complete discussion about various aspects of MSC and its applications are given in [7]. Here, care is taken by setting the bandwidth h of the kernel function in MSC such that it splits the data set into maximum number of clusters, by satisfying BIC.

Niche Genetic Algorithm

Genetic Algorithm (GA) is used in many applications to obtain global optimum solution. But there are many cases where we need local optimum solutions. So we modify the GA such that we can use it for such problems also. This is called niche GA. Initially chromosomes are randomly distributed. Fitness value of all chromosomes is calculated using equations given below:

$$factor = \sqrt{\sum_{i=1}^{d} (\max(a_i) - \min(a_i))^2} \tag{4}$$

$$\rho = \sum_{p=1}^{n} dist(agent, a(p)) \tag{5}$$

$$fitness = e^{-\rho / factor} \tag{6}$$

where d is number of dimensions, n is number of samples, a is data samples and $dist$ function gives Euclidean distance.

Using the fitness value of each chromosome we select set of best chromosomes which is used for next generation. Number of pre-crossover and mutated chromosomes depends on the fraction which we decide. Here we apply reproduction for best 40% and the remaining 60% undergo crossover and mutation. Using matrix of Euclidean distance between chromosomes and group of datasets which comes under sub-swarm radius are determined. Modified fitness value is calculated using equation (8), (9), so that we can retain different characteristics. Later we apply genetic operations like crossover and mutation on dataset which are within a niche.

$$\text{measure} \quad = 1 - dist \ / \sigma \tag{7}$$

where 'dist' is distance between best fitted data in that niche and the data for which we are calculating. 'σ' is threshold for a niche which is user defined. Here it is set to 0.5.

$$Mod\_fitness = fitness/measure \tag{8}$$

After crossover and mutation of each chromosome, fitness value of all chromosomes with their new positions is calculated using equation (7). The algorithm is implemented using the following steps:

Step 1. Initialization – randomly choose chromosomes
Step 2. Find fitness value of each chromosome using equation (7).
Step 3. Sort all the chromosomes according to their fitness values and select best 40% for next generation.
Step 4. Find out all niches and note all the chromosomes within that niche. Calculate modified fitness of all data within that niche using (8) and (9).
Step 5. Apply genetic operations like crossover and mutation.
Step 6. Find new fitness values of all chromosomes according to their new positions using equation (7).
Step 7. Sort all the chromosomes according to their fitness values and select best 60% for next generation.
Step 8. For next generation follow steps 2 to 7 and continue this until they are converged.

2.2 Merging Methodology

After obtaining cluster centers from MSC and NGA, data points are merged to its closest cluster centre. Labelling of the cluster is done using voting methodology. In voting method classification of clusters is done using original ground truth of the image. If the majority of the data points belong to a particular class then the whole cluster is labelled as that class. For example if there are 30 data points in a cluster and if 25 data points belong to class-X and remaining 5 belong to class-Y then that cluster is labelled as class-X. But if the distribution of number of data points among classes is uneven, say class-X has 1000 points and class-Y has only 70 points then this method may not work. Say in a cluster there are 80 points belonging to class-X and 40 points belonging to class-Y. Voting method will classify it as class-X. But we should observe that majority of the class-Y points is in this cluster but yet it is misclassified. Hence we keep a threshold for smaller classes. If the number of data points belonging

to smaller class in a cluster is more than that threshold then no matter which class has the highest vote; it is classified as that smaller class. In our study, we observed that there was an uneven distribution of number of data points among classes in LISS-III image. In comparison with other class labels, urban class data set is very less.

3 Result and Discussions

In this section, we compare the performance of the cluster splitting and merging techniques in extracting the classes of the LISS-III and SAR images. First, we describe the main characteristics of the satellite imagery.

3.1 Satellite Image Acquisition

The study area includes the regions affected by the Kosi river flood in Bihar during 2008. We focus on mainly the Bhagalpur district, where the Kosi tributary merges with Ganges, and assess the extent of damage caused. The region is contained within the co-ordinates: 25°39'30.76"N - 86°25'54.24"E and 25°12'34.42"N – 87°17'21.70"E. The districts affected by the flood and covered in this image include Bhagalpur, Khagaria, Katihar and Madhepur. Total area under study is 3248.9 km^2.

We use LISS-III image to assess the land cover prior to the flood. It was collected on April 11, 2008 from RESOURCESAT-2. For flood delineation, we use SAR image *C*-band dataset obtained from RADARSAT-2 on August 27, 2008 during the flood. The spatial resolution of the LISS-III image is 23.5 meters per pixel and SAR image is 30 meters per pixel. The image dataset used is of size 1614x3645 pixels. The SAR image is separated into two classes, namely flooded and non-flooded. The LISS-III image is classified into 4 different land cover types' namely urban land, fallow land, water and vegetation. Initially we use BIC to determine the optimal number of clusters required for the images. We observe that the optimal number of clusters is 9 for SAR image and 100 for LISS-III image.

We evaluate the performance of MSC and NGA on LISS-III and SAR image using the classification matrix of size n x n, where n is the number of classes. A value $A_{i,j}$ in this matrix indicates the number of samples of class i which have been classified into class j. For an ideal classifier, the classification matrix is diagonal. However, we get off-diagonal elements due to misclassification. Accuracy assessment of these image classification techniques is done using User's Accuracy (UA), Producer's Accuracy (PA), kappa co-efficient [19] and Overall Accuracy (OA) [9].

3.2 SAR Image Classification

To apply the MSC algorithm to the SAR image, we observe that the optimal bandwidth value is 15, as it yields 9 clusters. We then merge them into 2 classes, corresponding to flooded (A_1) and unflooded region (A_2).

The most favorable parameter values for NGA after different runs are as follows: Number of agents (samples considered for clustering) = 2000, Maximum no of iterations = 10, Sub-swarm radius (to find agents in niche) = 30 and Window size (used for merging) = 20.

Using the above parameters, the method yielded 9 cluster centers. We compare the performance of these techniques in classifying the SAR image, and the results are tabulated in Table 1. From this table, we observe that the two techniques all manage to effectively pick the flooded and unflooded regions, although some portion of flooded region has been incorrectly classified as unflooded. This is caused by the change in the dielectric value of the inundated surface, hence appears as unflooded region. In comparison with MSC, NGA gives better PA, UA, OA and Kappa.

Table 1. Comparison of classification performance on SAR, LISS-III and overlaid image

	SAR image (set A)				LISS-III image (set B)				Overlaid image (set C)			
	MSC		NGA		MSC		NGA		MSC		NGA	
	PA	UA	PA	UA	PA	UA	PA	UA	PA	UA	PA	UA
1	87	96.4	92.6	95.2	31.8	6	51.2	36.7	32.1	5.3	56.9	33.4
2	95.4	83.6	93.2	89.8	71.5	84.9	85.6	85.4	69	69.1	79.3	74.7
3	-	-	-	-	68.8	80.7	90.6	81.5	68.8	80.8	90.6	81.5
4	-	-	-	-	63.5	68.8	70.6	77.6	57.7	61.6	65.5	73.2
5	-	-	-	-	-	-	-	-	26.46	5.51	42.8	35.7
6	-	-	-	-	-	-	-	-	61.43	83.63	79.6	82.9
7	-	-	-	-	-	-	-	-	56.32	62.19	63.7	68.7
OA	90.4		92.9		68		81.2		61.9		75.8	
Kappa	0.807		0.853		0.459		0.645		0.518		0.679	

3.3 LISS-III Image Classification

To apply the MSC algorithm to the LISS-III image, we observe that the optimal bandwidth value is 6.1, as it yields 100 clusters. Further, we merge them into 4 classes, namely urban land (B_1), fallow land (B_2), water (B_3) and vegetation (B_4).

The most favorable parameter values for NGA after different runs are as follows: Number of agents (samples considered for clustering) = 2000, Maximum no of iterations) = 10, Sub-swarm radius (to find agents in niche) = 25 and Window size (used for merging) = 5.

Using the above parameters, the method yielded 100 cluster centers. We compare the performance of these techniques in classifying the LISS-III image, and the results are tabulated in Table 1. We observe that NGA provides far superior results in comparison with the MSC. In all the techniques, the accuracy is low for the class having less data points namely class B_1, which represents urban region. Since we use hierarchical clustering, the cluster centers of this smaller data points in a class are not picked, and the points get misclassified into the larger data points of a class. But we have modified the voting methodology by using threshold which is explained earlier in section 2.2. If the threshold chosen is very low, then small data points in the classes will be classified with greater accuracy on the cost of larger data points of the class. Pixels belonging to larger class will be misclassified to this small class. So we have to carefully choose the threshold such that we balance the individual accuracies of both small and larger data points in the class. Because of this NGA has picked class B_1 with greater accuracy. From Table 1 we can observe that, for LISS-III image also NGA outperforms MSC.

3.4 Overlaid Image

Further, the LISS-III and SAR image, obtained using MSC and NGA, are overlaid to obtain the resultant image that gives the flood extent in each class of the LISS-III image. It consists of 7 classes, namely unflooded urban land (C_1), unflooded fallow land (C_2), water (C_3), unflooded vegetation (C_4), flooded urban land (C_5), flooded fallow land (C_6) and flooded vegetation (C_7). The image obtained by overlaying the results of NGA on LISS-III and SAR images, is shown in Fig. 1.

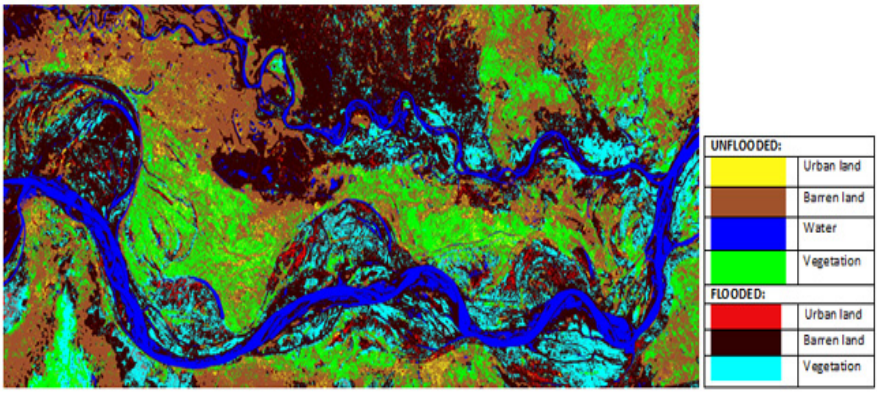


Fig. 1. Overlaid image of SAR and LISS-III image using NGA

We compare the results of these techniques in terms of accuracy of the overlaid image, and the results are tabulated in Table 1. Here we have 7 classes, corresponding to the flooded and unflooded portion of the land cover types. We observe that the flooded portion of a particular land class tends to get misclassified to its corresponding unflooded class. Also, the class having less data points, namely class C_1 and C_5 corresponding to unflooded and flooded urban regions, tend to get misclassified into different land cover types. This is because their cluster centres are not picked, and get merged with the larger classes. NGA provides more optimal results, and picks all the classes with greater accuracy than MSC.

4 Conclusion

In this paper, we have presented the classification methods for flood assessment problem. The satellite images used for this purpose are LISS-III (before flood) and SAR (during flood). We adopt MSC and NGA algorithms for splitting the data set by satisfying BIC and voting method is used to merge the data set. We varied the parameters in these techniques, to give the optimal number of clusters. In MSC, we analyzed the effect of bandwidth on number of clusters. In NGA, the prominent parameters are number of agents and sub-swarm radius, and their effect to generate optimal number of clusters. We observe that the performance of NGA is superior to that of MSC, as verified statistically using producer's accuracy, user's accuracy, overall accuracy and Kappa coefficient.

References

1. Duc, V.B.: Advantage of the remote sensing data utilization in studying inundation risks in terms of Land-use. In: IEEE Int. Conf. on Geoscience and Remote Sens. Symp., pp. 279–282 (2006)
2. Zhou, C.H., Luo, J.C., Yang, C.J., Li, B.L., Wang, S.L.: Flood monitoring using multi-temporal AVHRR and RADARSAT imagery. Photogrammetric Engineering and Remote Sensing 66(5), 633–638 (2000)
3. Freeman, A., Durden, S.: A three component scattering model for polarimetric SAR data. IEEE Tran. of Geoscience and Remote Sens., 36963–36973 (1998)
4. Shamaoma, H., Kerle, N., Alkema, D.: Extraction of flood-modelling related base-data from multi-source remote sensing imagery. In: ISPRS Mid-Term Symp. Remote Sens.: From Pixels to Processes, May 8-11. ITC, Enschede (2006)
5. Moungjin, L., Seongwoo, J., Soojyoung, M., Juongsun, W.: Detecting Flooded Location Using SAR Data and Assessment of Post-Flooded Condition. In: ACRS Proceedings (2008)
6. Hartigan, J.A., Wong, M.A.: A k-means clustering algorithm. J. Royal Statistical Society. Series C (Applied Statistics) 28(1), 100–108 (1979)
7. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Trans. Pattern Anal. Mach. Intell. 24(5), 603–619 (2002)
8. Fukunaga, K., Hostetler, L.: The estimation of the gradient of a density function, with applications in pattern recognition. IEEE Trans. Inf. Theory 21(1), 32–40 (1975)
9. Senthilnath, J., Omkar, S.N., Mani, V., Tejovanth, N., Diwakar, P.G., Shenoy, B.A.: Hierarchical Clustering Algorithm for Land Cover Mapping Using Satellite Images. IEEE Journal of Selected Topics in Appl. Earth Observations and Remote Sens. 5(3), 762–768 (2012)
10. Goldberg, D.E., Richardson, J.: Genetic algorithms with sharing for multimodal function optimization. In: Grefenstette, J.J. (ed.) Genetic Algorithms and their Application, pp. 41–49. Lawrence Erlbaum, Hillsdale (1987)
11. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading (1989)
12. Deb, K., Goldberg, D.E.: An investigation of niche and species-formationin genetic function optimization. In: Schaffer, J.D. (ed.) Proc. 3rd Int. Conf. Genetic Algorithms, San Mateo, CA, pp. 42–50 (1989)
13. Brits, R., Engelbrecht, A.P., Van den Bergh, F.: A niching particle swarm optimizer. In: Proc. 4th Asia-Pacific Conf. Simulated Evolutionand Learning, pp. 692–696 (2002)
14. Coello, C.A.C., Pulido, G.T., Lechuga, M.S.: Handling multiple objectives with particle swarm optimization. IEEE Trans. Evolutionary Computation 8(3), 256–279 (2004)
15. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proc. IEEE Int. Conf. Neural Networks, pp. 1942–1948 (1995)
16. Das, S., Maity, S., Qu, B.-Y., Suganthan, P.N.: Real-parameter evolutionary multimodal optimization - A survey of the state-of-the-art 1(2), 71–88 (2011)
17. Qu, B.-Y., Suganthan, P.N., Liang, J.J.: Differential Evolution with Neighborhood Mutation for Multimodal Optimization. Accepted by IEEE Trans. on Evolutionary Computation, doi:10.1109/TEVC.2011.2161873
18. Schwarz, G.: Estimating the dimension of a model. The Annals of Statistics 6(2), 461–464 (1978)
19. Rosenfield, G.H., Fitzpatrick-Lins, K.: A coefficient of agreement as a measure of thematic classification accuracy. Photogrammetric Engineering and Remote Sensing 52(2), 223–227 (1986)

Solution to Economic Load Dispatch Problem Based on FIREFLY Algorithm and Its Comparison with BFO,CBFO-S and CBFO-Hybrid

Ananthanaryanan Rathinam and Ripunjoy Phukan

Dept. of Electrical and Electronics Engineering
SRM University, Chennai (603203), India
a_rathinam@yahoo.com, ripun000@gmail.com

Abstract. The Economic Load Dispatch problem decides the minimum economic cost of production in a given power system, while keeping the environmental constraints and the load demand to an agreed level without compromising the generator ratings. Analysis on this application have been made using Bacteria Foraging Optimization (BFO) algorithm, where the bacterial motion is incorporated as a search algorithm in finding the optimum values for economic generation. An improvisation on this algorithm is the co-operative bacteria foraging optimization using serial decomposition (CBFO-s) and CBFO-hybrid that combines BFO and CBFO-s. Firefly Algorithm (FA) is a multi-modal meta-heuristic algorithm that uses the firefly's flash as a signal on the flies of opposite sex. The use of FA, guarantees minimized economic costs along with zero deviation from the target load in comparison to BFO and its variants.

Keywords: Valve Point effects, emission constraints, chemotactic steps, elimination & dispersal, foraging, Economic Load Dispatch.

1 Introduction

In Electrical power systems, the problem of unit commitment and economic generation are together considered as arduous and time consuming. The added effects of both are studied in a superficial approach termed as the Economic Load Dispatch Problem (ELD) [1-2].The overall scope of the problem gets elevated on adding multiple generators to the system. Since the inception of Artificial Intelligence (AI) in solving ELD [3], the scope of the solution has enhanced with decrease in computation times and balanced equilibrium solutions. One such approach was using neural networks [4]. A grueling approach to ELD was made when both the generation and transmission costs were considered for optimization [5]. But, eventually a polarized approach was recommended for a better convergence. However, these equilibrium points are not necessarily steady state points and are bound to change in the dynamic model and real time situation [6]. The idea of using heuristics to solve these problems stems from the fact that many local optimum points are generated within the solution space, and eventually the computations become restricted around this local point. With this in mind, several other techniques were brimming up in the 80s, and one of

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 57–65, 2012.

these methods was the Bacteria Foraging Optimization (BFO). This algorithm was eventually tuned to make the step size adapt as per the problem nature [7]. Eventually, to avoid lags in computation, the nested looping feature was brought into the algorithm with best iteration count for faster response [8]. Yet, another effective approach was adopted, namely the Co-operative Bacteria Foraging Algorithm involving serial decomposition in search space [9-10]. The hybrid algorithm, on the other hand, integrates the use of BFO and the serial decomposition together in a reduced search space. All these methods prove to be effective on a small scale only.

The use of FA came into picture by 2009 [11]. It is a novel approach that mimics the behavior of fireflies. It is a meta-heuristic algorithm that uses communication among fireflies as a means of motion to optimal points. The proposed algorithm was demonstrated on IEEE 30 and 118 bus systems [12-13]. Firefly was also successfully implemented in optimal location and size of distributed generation systems [14].

This paper aims to compare the effect of this algorithm on the ELD problem and compare with the BFO and CBFO based results. The subsequent sections deal with the ELD formulation followed by the firefly algorithm description. The system results are eventually summarized and their convergence is plotted.

2 Economic Dispatch Formulation

2.1 Objective Function

The primary concern of economic dispatch is the minimization of fuel cost involved with production, keeping in check with the generator rated power limit. Total cost is taken as the objective function and the fuel cost of a thermal generation unit is given as a second order polynomial as shown.

$$F_i(Pg_i)=a_i+b_i(Pg_i)+c_i(Pg_i)^2 \quad (1)$$

Where; a_i ,b_i , and c_i are constants (cost coefficients of i^{th} unit)
Pg_i is power output of i^{th} generator

2.2 Equality Constraints

Any power system needs to match the generator output with the demand, and also compensate for the losses. The mathematical formulation for this is given in equation,

$$\sum_{i=1}^{n} P\,g_i = Pd_i + Pl_i \quad (2)$$

Where; Pg_i is Power generation of i^{th} unit.
Pd_i is the Power demand of i^{th} unit.
Pl_i is net Power loss of i^{th} unit.

2.3 Inequality Constraints

Generation units have lower and upper bounds of rated power. Once the generator output exceeds the permissible limits, the generator suffers from over-voltage and

insulation damage. Excess load leads to under-voltage condition in generators, eventually causing load shedding problems. So, depending on the upper and lower boundaries of generator output, the optimization algorithm is tuned to become a constrained optimization problem. These bounds are designated by inequality constraints as shown in equation (3)

$$Pg_{i,min} < Pg_i < Pg_{i,max} \tag{3}$$

Where; $Pg_{i,min}$ is the minimum generator output
$Pg_{i,max}$ is the maximum generator output

2.4 Ramp Rate Limits

While starting and shutting down generators, transient response in steam flow is seen and accounted for in ELD problems. It also comes into picture, during increase or decrease of power generation. The objective function corresponding to the combined effect of load dispatch and ramp rate effect becomes like equation (4)

$$Fi\ (Pg_i) = a_i + (b_i).(Pg_i) + (c_i).(Pg_i^2) + (e_i).\sin\ (f_i \times (Pg_{i,min} - Pg_i)) \tag{4}$$

Where; e_i and f_i is the generator coefficients associated with valve point effects.

2.5 Emission Constraints

This effect is taken into consideration with regard to the harmful effects of environmental pollution and its control. This encompasses a market level problem for electric companies and needs to be checked. It is accomplished by introducing a separate equation in addition to the original ELD equation as shown in 5.

$$Fi\ (Pg_i) = a_i + (b_i).(Pg_i) + (c_i).(Pg_i^2) + \exp(d_i x Pg_{i,min}) \tag{5}$$

Where; d_i = emission constant
$Pg_{i,min}$ = lower power limit

3 Firefly Algorithm

3.1 Fireflies

This algorithm has been inspired by the synchronized flashing behavior of glowworms found in South East Asia. The foremost assumption of this algorithm is that every firefly is unisexual in nature. This means that every firefly can mate with every other firefly. The glowing nature of fireflies determines their mating abilities. The brighter the firefly the better chances for it to mate. In simple terms, it has a better fitness value than its neighbor. For minimization purpose we assume that the firefly with lesser intensity moves towards the firefly with higher intensity.

3.2 Algorithm

The flashing nature of fireflies is incorporated in the form of an algorithm as follows. First the intensity of brightness is calculated for every firefly separately. Mathematically, the brightness intensity is given by the equation (6).

$$I=I_o\,(e^{-\gamma\, r}) \tag{6}$$

Where I_o = intensity at the point r = 0
r being the distance between two fireflies

The intensity is calculated on a relative basis (i.e. between two fireflies). As can be seen, the intensity decreases as the distance increases due to exponential component. For the purpose of minimization, a firefly with lesser intensity would move towards the one with higher intensity. The movement of the firefly is dependent on the attractiveness feature stated below in equation (7).

$$\beta(r) = \beta_o\, e^{-\gamma\, r\wedge 2} \tag{7}$$

Where; β_o = attractiveness at r = 0

So, the movement of fireflies in one dimension can be analyzed as per the equation,

$$x_{i\,(new)} = x_{i\,(old)} + (\beta_o\, e^{-\gamma\, r\wedge 2}).(x_{j\,(old)} - x_{i\,(old)}) + \alpha.(\mathrm{rand}(1)) \tag{8}$$

Where; x_i is the position of firefly with lesser fitness
x_j is the position of firefly with higher fitness
α is the randomization parameter
rand is a function used for generating a random number in the domain(0,1)

3.3 Constrained Firefly Algorithm

The generalized firefly algorithm as given in [11] has been modified to suit the constraint requirements of the economic load dispatch. Care has been taken to avoid the overshoot of load and generated power values by introducing adjustment routines (Appendix A, equations 9 and 10). The flowchart representation of FA is given under Appendix B.

4 Test Systems and Results

The following experiments were simulated in MATLAB, R2008b using M-File programming. Simulations were done for BFO, CBFO-h, CBFO-s and firefly algorithm. The Co-operative algorithm [14] was coded in M-file and the PSO based results were obtained from Table 2.8 of [15]. For our analysis, we have taken a 3 gen.system (not shown)and 6 gen. system with 700 MW and 800 MW demand.

4.1 A 6 Generator System Was Considered and Experimentation Was Done as under

Load Demand: 700 MW		Load Demand: 800 MW	
Gen 1	67.5 MW	Gen 1	67.5 MW
Gen 2	60 MW	Gen 2	60 MW
Gen 3	120 MW	Gen 3	120 MW
Gen 4	102.5 MW	Gen 4	102.5 MW
Gen 5	207.5 MW	Gen 5	207.5 MW
Gen 6	142.5 MW	Gen 6	242.5 MW

Random data was taken for FA within permissible limits of generator ratings as shown

10MW<Gen 1<125MW
10MW<Gen 2<150MW
35MW<Gen 3<225MW
35MW<Gen 4<210MW
130MW<Gen 5<325MW
125MW<Gen 6<315M

Table 1. Data of cost estimates

(a) Without valve point effects and emission constraints					
load demand (MW)	**PSO (Rs.)**	**BFO (Rs.)**	**CBFO-S (Rs.)**	**CBFO-H (Rs.)**	**Firefly (Rs.)**
700	36987	36185	**36128**	36154	38600.31
800	42114	41701	41020	**40822**	41355.7

(b) With valve point effects only			
Load demand (MW)	**BFO (Rs.)**	**CBFO-s (Rs.)**	**Firefly (Rs.)**
700	**36355**	36514	36570.03
800	41435	41468	**40828.16**

(c) With emission constraints only			
Load demand (MW)	**BFO (Rs.)**	**CBFO-s (Rs.)**	**Firefly (Rs.)**
700	**36414**	36447	36484.2
800	**41077**	41146	42313.8

(d) With valve point effects and emission constraints				
Load demand (MW)	**BFO (Rs.)**	**CBFO-s (Rs.)**	**CBFO-H (Rs.)**	**Firefly (Rs.)**
700	**35758**	36171	35774	37768.38
800	**40750**	40940	40843	40869.38

Table 2. Unit allocationusing FA

Demand (MW)	Gen 1 (MW)	Gen 2 (MW)	Gen 3 (MW)	Gen 4 (MW)	Gen 5 (MW)	Gen 6 (MW)
(a) Without valve point effects and emission constraints						
700	117.74	14.463	147.79	42.668	212.85	164.47
800	18.638	134.33	141.86	69.021	173.18	262.93
(b) With valve point effects only						
700	27.847	47.633	117.43	104.28	212.92	189.87
800	27.142	46.196	138.45	177.34	270.11	143.75
(c) With emission constraints only						
700	18.81	95.095	79.358	59.582	233.38	213.49
800	87.19	36.074	90.070	45.186	138.52	402.90
(d) With valve point effects and emission constraints						
700	36.354	22.915	80.785	192.94	172.38	194.61
800	14.400	37.321	97.829	147.49	172.39	330.55

Table 1 depicts the cost functions associated with a 6 generator system and their comparison with different approaches. As seen, FA fails to provide better convergence, because of the improvisation in fitness values for BFO with the increase in number of agents or dimensions (based on the nature of objective function used). Also, it can be seen that all generator ratings sum up to the demand, i.e. 700 MW and 800 MW.Under the six generator system, weightage given to emission constraint was around 35% while to valve point effect was about 65%. The multi objective function was taken into consideration for analysis and BFO was found to give better results for cost function unlike in single objective function. When considering valve point effects only, CBFO-h fails to converge well, and gets stuck at local optimum points while FA gives minimum cost incurred in case of 800 MW and BFO for 700 MW. Similarly for objective functions with emission constraints only, CBFO-h fails to converge. In this case, as seen in all algorithms mentioned are giving the same results for cost function with minor variations between Rs.36400 to Rs.36500.Based on the above stated observations, the firefly algorithm shows lesser values of cost functions with 3 generator systems(Table 1). The effect is more pronounced with 700 MW demand.So, on comparison with BFO, CBFO-S and CBFO-h; FA gives better fitness values in all these concerned cases.

It can be observed that the adjustment routines implemented in the Firefly algorithm as shown in Appendix A, effectively distributes the output power from every generator as per the ratings as seen in their unit commitment values from Table 2, without disturbing the optimum values.This prevents damage to the generator windings due to overload or overcurrent conditions. It is also able to commit all units into proper operation, without stress on the power system.Moreover, the initial adjustments do not disturb the global best value. However, It can be seen that the FA starts with an assumption on the power values at the start of simulation. This ensures that the reults obtained are cogent and plausible. Also, with the use of random values at the start FA fails to give any desired output. On the contrary, BFO and its variants can be initiated with random power values without compromising the solution to the cost function.

For six generator system BFO and its variants have shown robustness in comparison with FA. This is primarily associated with the gradation in effectiveness of fitness

values with increase in dimensions and agents. In the following section an attempt has been made to study the convergence behavior of BFO and FA, and explain the effect of random function on BFO.

4.2 Convergence Data

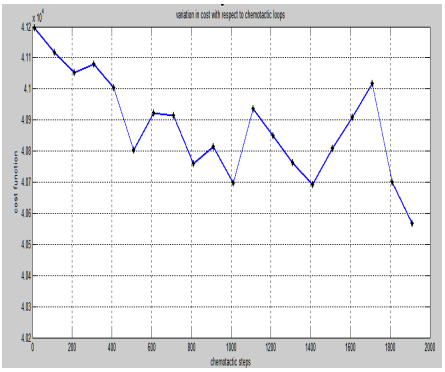


Fig. 1. Convergence data of BFO (6 generator system) without valve point effects and emission constraints

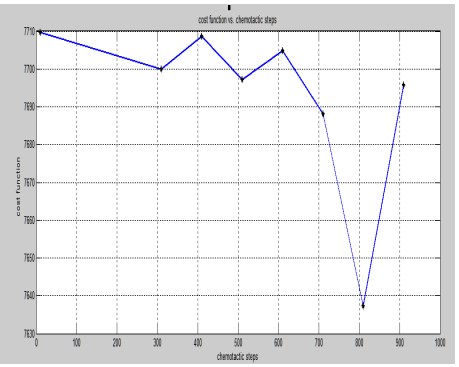


Fig. 2. Convergence values for 3 generator system with valve point effect and emission constraints (CBFO-S)

Compared to BFO and its variants, FA attains smooth convergence and a stable optimal solution after a few iterations. However, due to poor search space exploitation FA sometimes fails to give satisfactory results. Paradoxicaly, BFO enhances the search space to attain the best possible result.

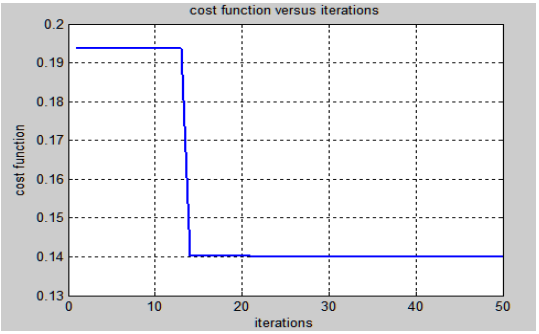


Fig. 3. Convergence data of FIREFLY (6 generator system) with valve point effects and emission constraints

5 Conclusion

The basic idea of BFO and firefly algorithm is the exploitation of the search space for global optimum values. Initially in both these algorithms the step length is very huge resulting in global scale search. However, after every iteration the fireflies or the

bacteria move towards this best value. Finally all the focus is driven on this particular optimum only. To start with, both these algorithm are based on stochastic methods, but when applied to BFO the algorithm gets stuck with local optimal points. So a deterministic approach is more advisable for BFO. No such problems occur with fireflies as the algorithm is self-correcting in nature. Moreover, FA gives the desired optimum after a few iterations and is relatively faster than BFO and its variants. To sum up, the firefly algorithm is potentially much better than BFO in terms of performance and results.

References

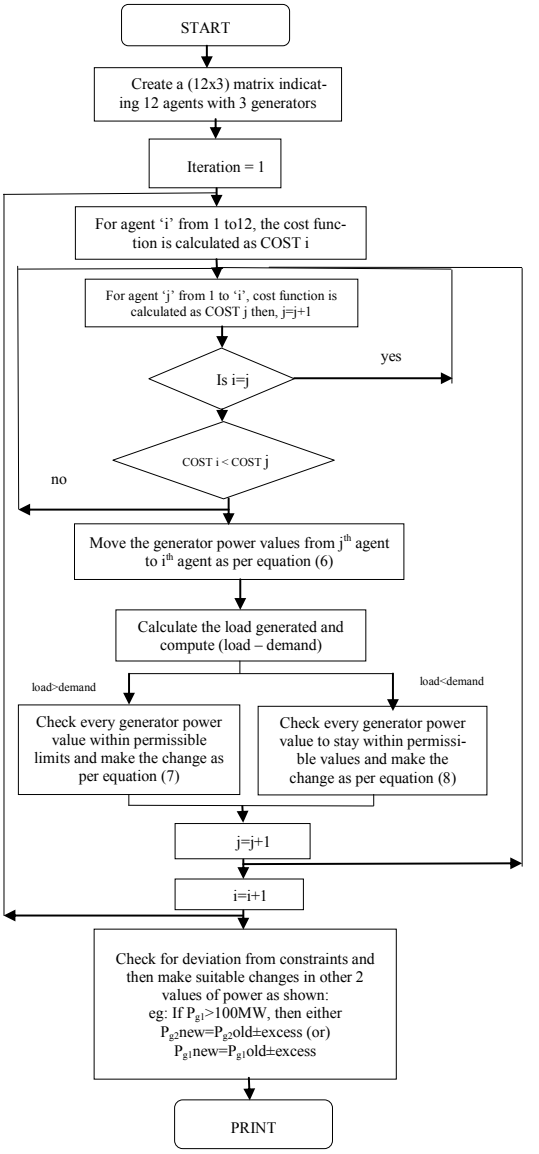
1. Parti, S.C., Kothari, D.P., Gupta, P.V.: Economic Thermal Power Dispatch. Institution for Engineers, (India) Journal 64, 123–132 (1983)
2. Yalanoz, T., Altun, H.: Environmentally constrained Economic Dispatch via. G.A. with arithmetic crossover. In: 6th Africon Conference in Africa, October 2-4, vol. 2, pp. 923–928 (2002)
3. Rahman, T.K.A., Asim, Z.M.: Artificial Immune-Based Foraging for Solving Economic Dispatch in Power system. In: National Power and Energy Conference, vol. 1, pp. 31–35 (2004)
4. Da Silva, I.N., Nepomuceno, L.: An Efficient Neural Approach to Economic Load Dispatch in Power systems. In: Power Engineering Society Summer Meeting, vol. 2, pp. 1269–1274 (2001)
5. Kumaran, G., Mouly, V.S.R.K.: Using Evolutionary Computation to solve the Economic Load Dispatch Problem. In: Congress on Evolutionary Computation, vol. 1, pp. 296–301 (2001)
6. Praveena, P., Vaisakh, K.: A Bacterial Foraging PSO-DE Algorithm for Solving Dynamic Economic Load Dispatch Problem with Security Constraints. In: Joint International Conference on Power Electronics, Drives and Energy Systems (PEDES), pp. 1–7 (2010)
7. Supriyono, H., Tokhi, M.O.: Bacterial Foraging Algorithm with Adaptable Chemotactic step Size. In: Second International Conference on Computational Intelligence, Communication Systems and Networks, pp. 72–77 (2010)
8. Munoz, M.A., Halgamuge, S.K.: Simplifying the Bacteria Foraging Optimization Algorithm. In: IEEE Conference on Evolutionary Computation, pp. 1–7 (2010)
9. Hui, C., Yang, L.: CBFO-The cooperative optimization of bacteria foraging. In: Computer Applications and System Modelling (ICCASM), vol. 2, pp. 106–109 (2010)
10. Shao, Y., Chen, H.: A Novel Bacteria Foraging Algorithm. In: Fourth International Conference on Bio Inspired Computing, pp. 1–4 (2009)
11. Yang, X.-S.: Firefly Algorithms for Multimodal Optimization. In: Watanabe, O., Zeugmann, T. (eds.) SAGA 2009. LNCS, vol. 5792, pp. 169–178. Springer, Heidelberg (2009)
12. Abedinia, O., Amjady, N.N.: Multi-Objective Environmental Economic dispatch using Firefly Technique. In: 11th International Conference on Environment and Electrical Engineering, pp. 461–466 (2012)
13. Wu, L.H., Wang, Y.N., Yuan, X.F.: Economic Power Dispatch Problem using multi obj. differential algorithm. Power Systems Research 80, 1171–1181 (2010)
14. Chen, H., Zhu, Y.: Research Article-Cooperative bacteria foraging optimization. Key Laboratory of Industrial Informatics, Chinese Academy of Sciences, Discrete Dynamics in Nature and Society 2009, Article ID 815247
15. Raj, P.A.D.V.: Performance Evaluation of Swarm Intelligence based Power system Optimization Techniques, ch. 2 (November 29, 2010)

Appendix: A

$$P_{gi}, new = P_{gi,old} - (load - demand)$$
$$P_{gi}, new = P_{gi,old} + (load - demand) \quad (9)$$

$$P_{gi}, new = P_{gi,old} - (demand - load)$$
$$P_{gi}, new = P_{gi,old} + (demand - load) \quad (10)$$

Appendix:B



MESFET DC Model Parameter Extraction Using Adaptive Accelerated Exploration Particle Swarm Optimizer

Layak Ali[1], Samrat L. Sabat[2], and Siba K. Udgata[3]

[1] Department of Information Technology, Deccan College of Engineering and Technology, Hyderabad-500001, India
informlayak@gmail.com
[2] School of Physics University of Hyderabad, Hyderabad-500046, India
slssp@uohyd.ernet.in
[3] Department of Computer & Information Sciences, Centre for Modeling Simulation and Design University of Hyderabad, Hyderabad 500046, India
udgatacs@uohyd.ernet.in

Abstract. This paper presents an application of Adaptive Accelerated Exploration Particle Swarm Optimization (AAEPSO) algorithm for extracting DC model parameters of a fabricated GaAs based Metal Extended Semiconductor Field Effect Transistor (MESFET). The AAEPSO algorithm is a variant of Particle swarm optimization algorithm that has proven to outperfrom basic PSO in solving benchmark problems. In this work we applied this algorithm to extract the MESFET model parameters by minimizing the error between the measured and modeled drain current. The performance of this approach is compared with popular algorithms like Simulated Annealing, Complex Method (CM), Artificial Bee Colony (ABC) and Particle Swarm Optimization (PSO) algorithms based on the (i) mean square error between the measured and modeled drain current, and (ii) convergence time. The comprehensive analysis of AAEPSO is carried out on four different MESFET DC models. Simulation results indicate that the AAEPSO algorithm gives good qaulity of solution in all the cases where as complex method takes less time for executing each iteration.

Keywords: Particle Swarm Optimization, Artficial Bee Colony, Simulated Annealing, Complex Method, Parameter Extraction, MESFET DC model.

1 Introduction

Most of the semiconductor devices operate satisfactorily in low frequencies and Gallium Arsenide Metal Semiconductor Field Effect Transistor (MESFET) device operates at microwave frequencies. Thus MESFET has its own niche at microwave frequencies. For every semicondctor device to be used in circuit simulation a model is necessary. The model describes the behavior of the device in

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 66–73, 2012.

different operating conditions. There are different techniques to develop models like physics based, empirical based etc. Physics based model requires in depth knowledge about the physics of the device where as empirical model avoids this, although each technique has their own pros and cons. These models will have set of parameters that can be either empirical or related to physics of the device. Extracting the values of these parameters is a crucial task, because many a times, there is no direct method to extract these parameters. So optimization based approach is the best proven way to find the values of model parameters. Many commercial TCAD tools such as [2], Silvaco UTMOST [1], TMA AURORA [3] etc are being used for extracting the model parameters. The commercial tool uses simulated annealing algorithm for model parameter extracion. Although optimization based approach are proven to better for parameter extraction but poor convergence rate and non optimal resulted solutions are the major drawbacks of the optimization approach. Non optimal results are due to trap of solutions in local optima. To overcome these drawbacks, particle swarm optimization (PSO) is being used for device model parameter-extraction [13]. The advantages of using PSO is that it is a heuristic approach, comutaionaly simple and leads to a global solution by avoiding local non-optimal solutions. Although the basic PSO is simple but sometimes it traps in local minima depending on the error surface. The authors have developed a new version of algorithm namely AAEPSO which has been proven to be a robust and efficient whilolving testbench functions [12]. The objective of this work is to find the suitability of AAEPSO [12] algorithm for MESFET DC model parameter extraction and compare its performance with algorithms like simulated annealing, complex method [11], Artificial bee colony (ABC) [9] and PSO algorithm [6] interms of convergence speed and quality of solution. The methodology proposed in this paper is to optimize the error between measured and simulated drain current using AAEPSO algorithm. The simulated data were obtained using different popular device models [4,10,14,7].

The paper is organized as follows: Section 2 presents a brief description of Device modeling and different well known MESFET device models. Section 3 presents an AAEPSO algorithm. In Section 4, the simulation and results are presented. Conclusions are drawn in Section 5.

2 Device Modeling and Parameter Extraction

Device model is the representation of a device through which the characteristics of the device can be studied prior to fabrication. Each semiconductor device have different models that reflects the behavior of the device under different operating conditions. Usually the model parameters are extracted using commercial software like HP IC-CAP [2], Silvaco UTMOST [1], TMA AURORA [3], etc. The accuracy of the commercial available software used for simulation depends on the accuracy of the models being used in it. A model is accurate if it fits the experimental data in each region of I_{ds} V_{ds} characteristics curve.

In order to closely resemble the model to experimental data, more numbers of parameter are required to represent the behavior of device accurately. This results to a complex model. The values of model parameters are extracted using parameter extraction algorithm. As the number of model parameter increases, the conventional parameter extraction algorithm fails to provide accurate values. An accurate model is always in demand for simulation before fabrication step. So an efficient parameter extraction algorithm is also in need for fabrication industry. The efficiency of algorithm depends on the computational complexity and quality of solution it finds.

2.1 Problem Formulation

MESFET device model parameter extraction can be easily solved by translating it to an optimization problem where the fitness/objective function is the mean square error between measured and modeled drain current at different bias points. This objective function is optimized to extract the model parameter of the device. For obtaining simulated data, we have considered four different popular MESFET DC model namely Curtice [4], Materka [10],Tajima [14] and TOM3 [7]. Mathematically the objective function is expressed as

$$E = \sum_{i=1}^{P} E_i \tag{1}$$

$$E_i = \sum_{j=1}^{K} \left(I_{modj} - I_{measj}\right)^2 \tag{2}$$

Where, P is the different bias points (V_{gs}) and I_{modj} is modeled drain current (using any of the model) (I_{ds}) at j bias point of (V_{ds}) and I_{measj} is the corresponding measured value. Proposed variant of PSO algorithm is used to minimize eqn (2) for extracting the model parameters which fits the device's behavior. In order to avoid the danger of blind optimization careful selection of the proper range of parameters to be optimized are chosen.

As we have considered four different MESFET model, due to number of page constraints, here we are describing only TOM3 model for a clear understanding of parameters in the model.

In this model the drain current is expressed as as [8]

$$I_{ds}(V_{ds}, V_{gs}, \theta) = \beta(V_G)^Q \frac{\alpha V_{ds}}{\sqrt{(1 + (\alpha V_{ds})^2)}}(1 + \lambda V_{ds}) \tag{3}$$

where

$$V_G = QV_{st}log[1 + exp(u)] \tag{4}$$

$$u = \frac{V_{gs} - V_{to} + \gamma V_{ds}}{QV_{st}} \tag{5}$$

$$V_{st} = V_{sto}(1 + M_{sto}V_{ds}) \tag{6}$$

where θ is the model parameter vector as listed in Table 1.

Table 1. TOM3 MESFET model

parameters	Description	Unit
β	Transconductance parameter	mA/V^2
V_{to}	Threshold voltage	V
Q	Power-law parameter	-
K	Knee-function parameter	-
α	Knee-voltage parameter	V^{-1}
λ	Channel length modulation parameter	mV^{-1}
V_{sto}	Sub-threshold slope	V
M_{sto}	Sub-threshold slope drain parameter	V^{-1}

Algorithm 1. Pseudo code for AAEPSO

Initialize
$Set \leftarrow X_{max}, X_{min}, D, NP$
$V_{max} \leftarrow 0.20 * (X_{max} - X_{min})$
$t \leftarrow 0, i \leftarrow 0$ ▷ t for iterations and i for particles
Randomly initialize particle's position $X_i^0 \in D$ in R;
Randomly initialize particle's velocity $V_i^0 \leq V_{max}$
Evaluate fitness function values f_i^0
$pbest_i^0 \leftarrow f_i^0, \quad gbest^0 \leftarrow f_{best}^0$
Optimize
while $t \leq MaximumGeneration$ **do**
  Decrease AE_f Acceleration and Exploration factor exponentially with iteration [12].
  while $i \leq NP$ **do**
Update velocity (equation (7)) and position (equation (8))
Evaluate fitness function values f_i^{t+1}
    Find $pbest_i^{t+1}$
    if $f_i^{t+1} < pbest_i^t$ **then** ▷ for minimization problem
      $pbest_i^{t+1} \leftarrow f_i^{t+1}$
    end if
    $i \leftarrow i + 1$, *go to step* 10.
  end while
  Find $gbest_d^{t+1}$
  if $gbest_d^{t+1} < gbest_d^t$ **then** ▷ for minimization problem
    $gbest_d^{t+1} \leftarrow gbest_d^t$
  end if
  if stop criteria not met **then**
    $t \leftarrow t + 1$, *go to step* 8.
  end if
  Identification & acceleration of diverged particles [12]
end while
Report results
Terminate

3 Adaptive Accelerated Exploration Particle Swarm Optimizers

Particle Swarm Optimization is a heuristic technique used for finding global solution proposed by Kennedy and Eberhart in 1995 [6,5]. It has proven to be simple, and efficient compared to other optimization techniques. As the dimension of problem increases basic PSO also suffers with poor quality of solution and premature convergence. In this paper we are using a novel variant of PSO called Adaptive Accelerated Exploration Particle Swarm Optimization (AAEPSO) [12], that gives better results for higher dimensional optimization problems.

The practical implementation of AAEPSO involves the steps as shown in Algorithm 1. It is almost similar to PSO with the major difference being adaptive acceleration and exploration. The selection factor S_f in AAEPSO is the random permutation in the range [10 - 90] % of the diverged population. The acceleration-exploration factor AE_f is decreased exponentially with iteration [12]. The velocity and position update equation for AAEPSO are

$$V_{i,d}^{t+1} = w^t V_{i,d}^t + c_1^t * rand_1 * (pbest_{i,d}^t - X_{i,d}^t) + c_2^t * rand_2 * (gbest_d^t - X_{i,d}^t) \quad (7)$$

$$X_{i,d}^{t+1} = X_{i,d}^t + V_{i,d}^{t+1} \quad (8)$$

Where w^t, c_1^t and c_2^t are linearly decreasing inertia & learning factors.

4 Results

4.1 Simulation Settings

The simulations were carried out using Pentium Core2Duo, 2GHz with 2GB RAM.Algorithms are coded in Matlab 7.2 in Windows-XP platform. Different Models of MESFET for same dimension gate width W= 50 μ m and gate length L = 2μm as are simulated through the proposed technique. Experiments are carried out with the population size (NP) of 20, the number of iterations (MaxGen) 1000. The dimension of the problem is model dependent. The algorithms are tested on a different models mentioned above. The extraction algorithms were executed for 20 times to observe the consistency of the extracted parameter value. So the consistent result appeals that proposed technique is a robust technique for parameter extraction.

4.2 Experimental Results and Discussion

The DC model parameters of all the four considered models are extracted using different stochastic algorithms including AAEPSO algorithm. The mean value of extracted parameters , the mean square error between the measured and simulated drain current using different models and the total convergence time are tabulated in Table 1. The tabulated result infers that Complex Method

takes very less convergence time but the quality of solution it achieves is poor compared to AAEPSO on all the models. Similarly AAEPSO gives least error as compared to all other algorihms in all the model except Tajima odel. Though AAEPSO takes more time for execution compared to Complex Method but the accuracy AAEPSO achieves is far better than any other algorithm. Again on TOM3 model AAEPSO retains the same trend and surpasses all other algorithm in achieving the quality of results. Simulated Annealing (SA) gives better results on Tajima model. Complex meethod had has faster convergence rate in all the model. So if quality of solution is considered then AAEPSO algorithm is the best choice where as if speed is considered with a trade off to quality of solution then complex method is better for device model parameter extraction.

Fig. 1 shows accuracy of simulated results compared with measured data from a fabricated MESFET using AAEPSO algorithm. The reported results are the mean result of 20 simulation runs. The solid lines are experimental drain current and ··· lines are modeled drain current of different models after parameter extraction. Materka model simulates reasonably well in linear region, whereas Tajima and TOM3 models has excellent agreement with saturation region.

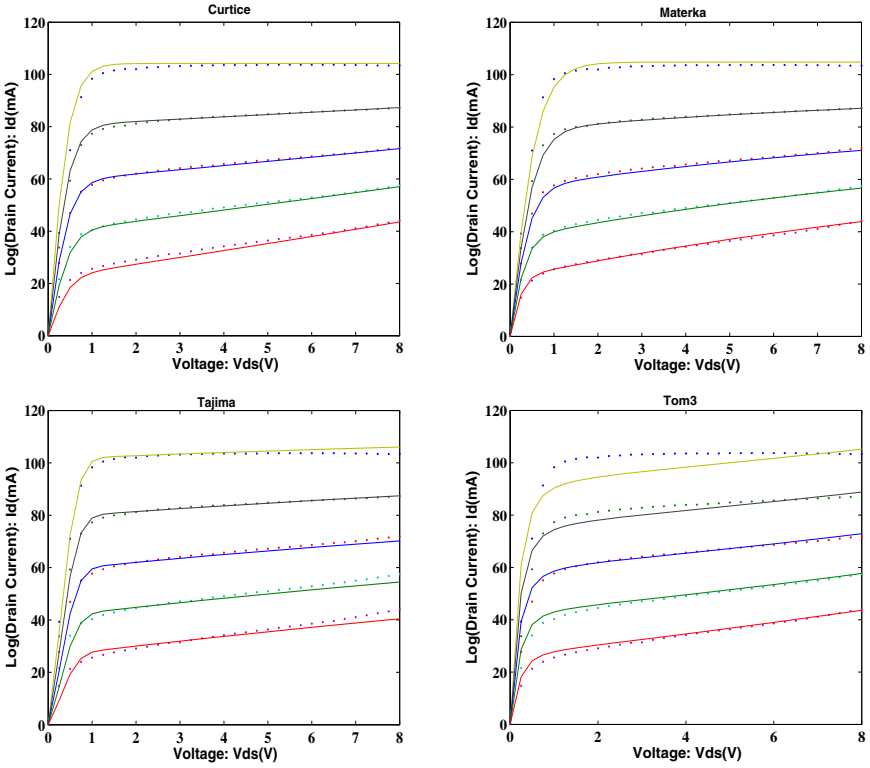


Fig. 1. Simulated (–) and measured (···) DC characteristics

Table 2. Optimum values of extracted parameters for MESFET DC Models

Matreka	I_{DSS}	V_{r0}	α	γ	Error	Con.Time (sec)
SA	0.104	-1.90	3.29	-0.121	1.76e-3	1.61
ABC	0.105	-1.90	3.24	-0.121	1.67e-3	5.75
CMP	0.104	-1.89	3.95	-0.124	1.04e-3	0.08
PSO	0.1	-1.89	4.02	-0.13	3.1e-4	7.1
AAEPSO	0.104	-1.89	4.15	-0.19	2.64e-4	0.57

Curtice	β	A_0	A_1	A_2	A_3	γ	V_{ds0}	Error	Con.Time (sec)
SA	0.038	0.103	0.089	0.014	-1.94e-4	1.85	1.97	7.22e-6	2.45
ABC	0.043	0.105	0.11	0.056	1.64e-2	1.93	0.79	1.98e-4	8.95
CMP	0.041	0.103	0.099	0.018	1.63e-3	1.96	1.78	1.30e-3	0.11
PSO	0.04	0.1	0.09	0.03	0.01	0.02	2.18	5.93e+1	2.94
AAEPSO	0.0425	0.104	0.093	0.024	5.56e-3	1.91	2.11	3.55e-8	1.26

Tom3	α	β	λ	γ	V_{t0}	M_{st0}	Q	V_{st0}	Error	Con.Time (sec)
SA	2.60	0.06	3.1e-3	0.017	-1.48	0.32	1.13	0.11	1.4e-2	3.00
ABC	2.74	0.068	4.30e-4	0.021	-1.37	0.25	1.03	0.137	4.40e-3	11
CMP	2.72	0.058	3.58e-3	0.019	-1.51	0.27	1.17	0.15	7.53e-3	0.1
PSO	2.57	0.03	1.0e-2	0.0101	-2.13	0.14	1.75	0.051	2.46e-1	5
AAEPSO	2.65	0.047	4.29e-3	0.019	-1.67	0.25	1.42	0.18	5.54e-4	1.60

Tajima	V_{BI}	V_{PO}	P	m	I_{dsp}	V_{DSS}	a	b	Error	Con.Time (sec)
SA	0.53	1.11	0.082	0.64	0.097	0.93	2.26	0.19	1.31e-4	3.64
ABC	0.17	1.79	0.094	1.10	0.11	0.89	2.80	-0.31	9.72e-5	14.8
CMP	0.302	1.87	0.08	1.06	-0.13	1.03	3.07	0.51	5.89e-3	0.23
PSO	0.41	1.6	0.09	0.46	0.12	1.08	2.89	0.41	1.8e-3	10
AAEPSO	0.93	0.64	0.089	-0.32	0.12	1.11	2.30	0.58	4.34e-3	3.10

5 Conclusion

In this paper, we have used a new variant of PSo algorithm namely AAEPSO for MESFET DC model parameter extraction. The parameters of different MESFET models are extracted from measured I_d - V_d data of MESFET device of channel length 2 μ m and width 50 μm. Out of these models Materka, TOM3 and Curtice models are the suitable model for the fabricated MESFET. The complex method also proved the superiority of faster convergence in this problem. The experimental result concludes that if quality of solution is the criteria then AAEPSO algorithm is the best choice where as if speed is the criteria with a trade off to quality of solution then complex method is better for device model parameter extraction. In future we will explore more about complex method and tune it for giving good quality of solution.

References

1. Silvaco international. Utmost iii modeling manual. Santa Clara, CA (1992)
2. Hewlett-packard co. IC-CAP users manual (1994)
3. Technolong modeling associates, inc. AURORA device characterization system (1994)

4. Curtice, W., Ettenberg, M.: A nonlinear GaAs FET model for use in the design of output circuits for power amplifiers. IEEE Transactions on Microwave Theory and Techniques (1985)
5. Eberhart, R., Kenedy, J.: A new optimizer using particle swarm theory. In: Proceedings of 6th International Symposium on Micro Machine and Human Science (MHS), Cape Cod, MA, pp. 39–43 (November 1995)
6. Eberhart, R., Kenedy, J.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, Piscataway, NJ, pp. 1114–1121 (November 1995)
7. Hallgen, R.B., Litzenberg, P.H.: TOM3 capacitance model: linking large-and small-signal MESFET models. IEEE Transactions on Microwave Theory and Techniques (1999)
8. Hallgren, R., Litzenberg, P.: TOM3 capacitance model: linking large- and small-signal MESFET models in SPICE. IEEE Trans. Microwave Theory Tech. 47(5), 556–561 (1999)
9. Karaboga, D., Basturk, B.: On the performance of artificial bee colony (ABC) algorithm. Appl. Soft Computing 8(3), 687–697 (2008)
10. Materka, A., Kacprzak, K.: Computer calculation of large-signal GaAs FET amplifier characteristics. IEEE Transactions on Microwave Theory and Techniques (1985)
11. BoX, M.J.: A new method of constrained optimization and a comparison with other methods. Computer Journal 8, 42–52 (1965)
12. Sabat, S.L., Ali, L., Udgata, S.K.: Adaptive accelerated exploration particle swarm optimizer for global multimodal functions. In: Proceedings of World Congress on Nature and Biologically Inspired Computing, Coimbatore, India, pp. 654–659 (December 2009)
13. Sabat, S.L., dos S. Coelho, L., Abraham, A.: MESFET DC model parameter extraction using Quantum Particle Swarm Optimization. Microelectronics Reliability 49(6), 660–666 (2009)
14. Tajima, Y., Wrona, B., Mishima, K.: GaAs FET large-signal model and its application to circuit designs. IEEE Transactions on Electron Dev. 28(2), 171–175 (1981)

Kernel Group Method of Data Handling: Application to Regression Problems

Kalam Narendar Reddy and Vadlamani Ravi*

Institute for Development and Research in Banking Technology (IDRBT), Castle Hills Road #1, Masab Tank, Hyderabad-500057 (AP), India
narendarcse07@gmail.com, rav_padma@yahoo.com

Abstract. In this paper, a novel Kernel based Soft Computing hybrid viz., Kernel Group Method of Data Handling (KGMDH) is proposed to solve regression problems. In the proposed KGMDH technique, Kernel trick is employed on the input data in order to get Kernel matrix, which in turn becomes input to GMDH. Several experiments are conducted on five benchmark regression datasets to assess the effectiveness of the proposed technique. The results and a statistical t-test conducted thereof indicate that the proposed KGMDH yields more accurate results than the standalone GMDH in most datasets. This is the significant outcome of the study.

1 Introduction

Many data mining algorithms are intrinsically so powerful that they can efficiently mine data which are nonlinearly separable. However, another approach of mining nonlinearly separable datasets is to project the original dataset with some dimensionality into a higher dimensional feature space. Support vector machine follows this approach to a prodigious effect. The mapping transforms the nonlinearly separable data into linearly separable data that can be easily mined thereafter. Kernel trick is employed to avoid explicit mapping from original input space S to higher dimensional inner product space V as the process of explicit mapping is cumbersome. Consequently, the observations will become linearly separable. According to Mercer's theorem there is no need to know to which higher dimension we have to project the data to make it linearly separable.

David Hilbert [5] used the German word kern in his first paper on integral equations. The mathematical result underlying the kernel trick, Mercer's theorem [13], is almost a century old. The use of Mercer's theorem for interpreting kernels as inner products in a feature space was introduced into machine learning by Aizerman et al [2] (1964). Boser et al. [3] suggested a way to create nonlinear classifiers by applying the Kernel trick to maximum-margin hyper planes, which is used to solve the soft margin problem of Support Vector Machine (SVM). Scholkopf et al [22] used kernel functions to perform Principle Component Analysis (PCA). The goal of PCA is to find the directions along which the variance is maximum. But,

* Corresponding author.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 74–81, 2012.

if the data is non linearly separable, then PCA tends to fail. Hence, in KPCA [17], they performed on reproducing kernel Hilbert space. Then, Mika et al. [15] used kernels to perform Fisher Discriminant Analysis. Fisher's linear discriminants are methods used in statistics used to find linear combination of features that separates two or more classes. Other kernel techniques. where the traditional data analysis techniques are supplied with kernel matrix as input include kernel logistic regression [27] and kernel quantile regression [26].

The present paper proposes a hybrid classification technique viz., Kernel Group Method of Data Handling (KGMDH). The KGMDH employs Kernel trick and Group Method of Data Handling (GMDH) in tandem, where Kernel trick is used to find the kernel matrix, which is fed as input to GMDH. The effectiveness of the proposed techniques is tested on several benchmark regression datasets.

The rest of the paper is organized as follows. Section 2 presents a review of Kernel techniques so far in the literature and review of applications of GMDH. The architecture of the proposed classification techniques are explained in Section 3. Experimental setup is explained in section 4. Results and discussions are presented in section 5 followed by conclusions in section 6.

2 Literature Review

Group Method of Data Handling (GMDH), proposed by Ivakhnenko [6] has a wide range of applications in Data Mining, Forecasting systems, Pattern recognition and Optimization. The GMDH is a self-organizing approach based on sorting-out of gradually complicated models and evaluating them by external criterion on a separate part of data sample. Not only polynomials but also non-linear, probabilistic functions are used as basic models. GMDH models the input-output relationship of a complex system using a multilayered Rosenblatt's perception-type network structure. Each element in the network implements a non-linear equation of two inputs and its coefficients are determined by regression analysis. Self selection thresholds are given at each layer in the network to delete those elements, which can not estimate the correct output. Only those elements whose performance indices exceed the threshold are allowed to pass to succeeding layers, where more complex combination is formed. These steps are repeated until the convergence criterion is satisfied or a predetermined number of layers is reached. GMDH has the following advantages:

- A small training set of data is required.
- It guarantees that the most accurate or unbiased models will be found i.e., it doesn't miss the best solution during sorting of all variants (in given class of functions).
- The procedure automatically filters out input properties that provide little information about the location and shape of hyper surface.
- A multilayer structure is a computationally feasible way to implement multinomials of high degree.

GMDH was applied in complex modeling system of a mandarin tree water usage [14]. Abdel-Aal [1] used GMDH to feature ranking and selection and used these reduced features to achieve higher classification rate. Marcin et al. [7] proposed new passive robust fault detection scheme using non-linear models with GMDH that

include parameter uncertainty. Then, Schetinin et al. [21] used a polynomial network technique using an evolutionary strategy implemented within GMDH for classifying clinical electroencephalograms (EEGs) presented by noisy features. Later, Srinivasan [24] employed GMDH for electric energy forecasting. Thereafter, Mohanty et al. [16] used GMDH for Software Reliability Prediction. Like all neural network architectures, GMDH also suffers from the black box stigma. Hence, Naveen et al. [17] made GMDH transparent by employing GMDH and a decision tree in tandem for rule extraction for predicting churners in bank credit cards.

3 Kernel GMDH

The GMDH attempts to provide a hierarchic solution, by trying many simple models, retaining the best, and building on them iteratively, to obtain a composition of functions as the model. GMDH algorithm can be represented as a set of neurons in which different pairs of them in each layer are connected through a quadratic polynomial and thus produce new neuron in the next layer. These polynomial terms are created by using linear and non-linear regressions.

The building blocks, or polynomial nodes, usually have the quadratic form as follows.

$$Z = W_0 + W_1X_1 + W_2X_2 + W_3{X_1}^2 + W_4{X_2}^2 + W_5X_1X_2$$

for inputs x_1 and x_2, coefficient (or weight) vector w, and node output, z. The coefficients are found by solving the Linear Regression (LR) equations with z = y, the response vector.

The GMDH neural network develops on a data set. The data set including independent variables (x_1, x_2. . . x_n) and one dependent variable y is split into training and testing set. During a learning process a forward multi-layer neural network is developed in the following steps:

1. In the input layer of the network n units with an elementary transfer function y = xi are constructed. These are used to provide values of independent variables from the learning set to following layers of the network.
2. When constructing a hidden layer an initial population of units is generated. Each unit corresponds to Ivakhnenko polynomial form:

$$y = +bx_1 + cx_2 + dx_1^2 + ex_1x_2 \quad (1)$$

   Where y is an output variable; x_1, x_2 are two input variables and a, b,. . . , e are parameters.
3. Parameters of all units in the layer are estimated using the learning set.
4. Then the mean square error between the dependent variable y and the response of each unit is computed for the testing set.
5. Units are sorted out by the mean square error and just a few units with minimal error survive. The rest of units are deleted. This step guaranties that only units with good approximation ability are chosen.
6. Next hidden layers are constructed while the mean square error of the best unit decreases.
7. Output of the network is considered as the response of the best unit in the layer with the minimal error.

GMDH network is a relatively unexplored neural network. In these networks, the most important input variables, number of layers, neurons in hidden layers and optimal model structure are determined automatically. Majority of GMDH network implementations use regression analysis for solving the problem. The first step is to decide the type of polynomial that regression should find. General connection between input and output variables can be expressed by Volterra functional series, discrete analog of which is Kolmogorov–Gabor polynomial. The next step is to construct a linear combination of all of the polynomial terms with variable coefficients. The algorithm determines values of these coefficients by minimizing the squared sum (over all samples) of differences between sample outputs and model predictions.

The initial layer is simply the input layer. The first layer created is made by computing regressions of the input variables and then choosing the best ones. The second layer is created by computing regressions of the values in the first layer along with the input variables. This means that the algorithm essentially builds polynomials of polynomials. Again, only the best are chosen by the algorithm. These are called survivors. This process continues until a pre-specified selection criterion is met. The architecture of GMDH is shown in Fig.1. In the proposed Kernel based hybrid Kernel GMDH we applied Kernel trick on the input data to get the Kernel matrix on which GMDH is employed.

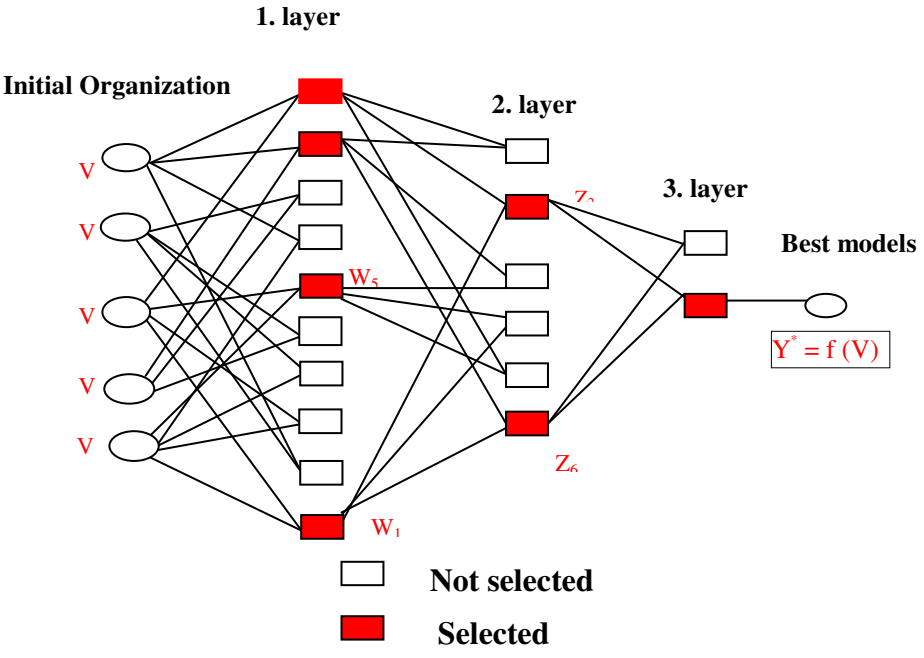


Fig. 1. Architecture of KGMDH

The steps involved in Kernel GMDH algorithms are given below:

Step 1: Applying Kernel technique

i. Compute the kernel matrix, for the training data, $K = [K_{ij}]_{npXnp}$, where $K_{ij}=K(x_i,x_j)$.

ii. Compute the kernel matrix, for the testing data, $K_{te} = [K_{ti}]_{ntXnp}$, where $n_{=}$ np + nt, $K_{te}=K(x_t,x_i)$. Here K_{te} projects the test data x_t onto training data x_i in the high dimensional feature space in terms of the inner product.

iii. Centralize K and K_{te} using

$$K = \left[I_{np} - \frac{1_{np}\,1'_{np}}{np}\right] K \left[I_{np} - \frac{1_{np}\,1'_{np}}{np}\right] \quad (2)$$

$$K_{te} = \left[K_{te} - \frac{1_{np}\,1'_{np}}{np} K\right] \left[I_{np} - \frac{1_{np}\,1'_{np}}{np}\right]$$

iv. Combine K and K_{te} matrices to form total centralized kernel matrix K_{tot} of order $n\times np$, where $n_{=}$ np + nt.

v. Center and scale all the features in the matrix K_{tot} by forming new matrix Z to make them dimension less.

Step 2: Applying GMDH on Kernel matrix

The procedure for employing GMDH on Kernel matrix is already explained in the above steps.

4 Dataset Description and Experimental Setup

The Boston housing dataset is taken from Statlib library which is maintained at Carnegie Mellon University. The dataset describes the housing values in the suburbs of Boston. The dataset contains 506 records and 13 attributes and is obtained from [7]. The Forest Fires dataset contains 517 records and 11 attributes and is obtained from [8]. The Auto MPG dataset is taken from Statlib library maintained at Carnegie Mellon University. The dataset is used in the 1993 American Statistical Association Exposition. The dataset concerns city-cycle fuel consumption in miles per gallon, to be produced in terms of 3 multivalued discrete and 5 continuous attributes. The dataset is obtained from [9]. The Body fat dataset lists the estimates of percentage of body fat determined by underwater weighing and various body circumference measurements for 252 men. The Pollution dataset lists the estimates relating air pollution to mortality determined by various characteristics of the environment and people. It contains 60 instances and 16 attributes. Body fat and Pollution datasets are obtained from [10] and [11] respectively.

All the experiments are conducted using 10 fold cross validation in all datasets. For these regression datasets, the error measure is mean squared error.

5 Results and Discussion

We implemented Kernel method using Java (JDK 1.5) on Windows 7 platform on machine with a RAM of 2GB. We employed GMDH using the tool NeuroShell 2. The parameters used for GMDH were maximum variables in connection, maximum product term in connection, maximum variable degree in connection, selection criterion, model optimization, maximum number of survivors in the first layer, schedule type and criterion coefficient. We observed that the following parameter values constitute the best parameter set in most of the cases. The maximum number of variables is taken as 3. The selection criterion is selected as regularity (calibration). The model optimization is selected as thorough. The maximum number of survivors in the first layer is taken as the number of inputs of the given dataset. The schedule type is taken as constant.

For the five regression datasets viz., Auto MPG, Forest fires, Boston housing, Body fat and Pollution, the results of the proposed KGMDH are compared with that of the standalone GMDH. The results are presented in Table 1.

Table 1. Average MSE values and t-statistic values

Dataset name	KGMDH	GMDH	t-statistic value
Forest Fires	0.0027	0.0032	0.474
Boston Housing	0.0044	0.0052	0.752
Auto MPG	0.0044	0.0046	2.88
Body Fat	0.0038	0.0040	8.078
Pollution	0.0074	0.0079	4.35

By employing KGMDH, in the case of Auto MPG dataset, average MSE value got reduced from 0.0046 to 0.0044; in the case of Forest fires dataset, MSE value got reduced from 0.0032 to 0.0027; in the case of Boston Housing dataset, its value got reduced from 0.0052 to 0.0044; in the case of Body fat dataset, its value got reduced from 0.0040 to 0.0038; and finally, in the case of Pollution dataset, its value got reduced from 0.0079 to 0.0074.

This reduction in MSE could be a statistical happening. Hence, to conclusively comment on the superiority of KGMDH, we performed t-test at 1% level of significance on the average MSE of all the datasets, to see if the difference between GMDH and KGMDH is statistically significant. The t-statistic values (see Table 1) are compared to 2.83, which is the t-test table value at 18 degrees of freedom (10+10-2=18) at 1% level of significance. In the case of Auto MPG, Body fat and Pollution datasets, the t-test suggests that there is a statistically significant difference between GMDH and KGMDH and KGMDH outperforms GMDH. However, in the case of Boston housing and Forest fires datasets, the t-test indicates that there is no statistically significant difference between the two techniques. Thus, in most of the datasets, KGMDH outperformed GMDH.

6 Conclusions

A novel hybrid using Kernels and GMDH in tandem for solving regression problems is proposed. We observed that the proposed architecture KGMDH outperformed the GMDH in terms of MSE for all the regression datasets. The t-test conducted between GMDH and KGMDH, also suggested that there is statistically significant difference between them, in majority of the cases. Hence, we can conclude that proposed kernel technique KGMDH is a viable option for building model for regression datasets.

References

1. Abdel-Aal, R.E.: GMDH-based feature ranking and selection for improved classification of medical data. Journal of Biomedical Informatics 38, 456–468 (2005)
2. Aizerman, M.A., Braverman, E.M., Rozonoer, L.I.: Theoritical Foundations of the Potential Function Method in Pattern Recognition Learning. Automation and Remote Control 25(6), 821–837 (1964)
3. Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classifiers. In: COLT 1992: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pp. 144–152. ACM Press, New York (1992)
4. Farlow, S.J.: Self-Organizing Methods in Modeling: GMDH type Algorithm. Marcel Dekker Inc., New York (1984)
5. Hilbert, D.: Gruendzugeeiner allgemeinen Theorieder linearen Integralgleichungen (Erste Mitteilung). Nachrichten von der Koenigl. Gesellschaft der Wissenschaften zu Goettingen, Mathematisch-physikalische Klasse (1), 49–91 (1904) (in German)
6. Ivakhnenko, A.G.: The group method of data handling - a rival of the method of stochastic approximation. Soviet Automatic Control 13(3), 43–55 (1966)
7. `http://archive.ics.uci.edu/ml/machine-learning-databases/housing/housing.data`
8. `http://archive.ics.uci.edu/ml/machine-learning-databases/forest-fires/forestfires.csv`
9. `http://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.data`
10. `http://lib.stat.cmu.edu/datasets/bodyfat`
11. `http://lib.stat.cmu.edu/datasets/pollution`
12. Marcin, W., Jozef, K., Marcin, M., Ron, J.P.: A GMDH neural network-based approach to robust fault diagnosis: applications to the DAMADICS benchmark problem. Control Engineering Practice 14, 671–683 (2006)
13. Mercer, J.: Functions of Positive and Negative Type, and their Connection with the Theory of Integral Equations. Philosophical Transactions of the Royal Society of London. Series A. Containing Papers of a Mathematical or Physical Character 209, 415–446 (1990)
14. Kordik, P., Naplava, P., Snorek, M., Genyk, M.: The modified GMDH Method Applied to Model Complex System (1982)
15. Mika, S., Ratsch, G., Weston, J., Scholkopf, B., Muller, K.R.: Fisher Dicriminant Analysis. In: Neural Networks for Signal Processing IX, Proceedings of the 1999 IEEE Signal Processing Society Workshop, pp. 41–48 (1999)

16. Mohanty, R., Ravi, V., Patra, M.R.: Software Reliability Prediction Using Group Method of Data Handling. In: Sakai, H., Chakraborty, M.K., Hassanien, A.E., Ślęzak, D., Zhu, W. (eds.) RSFDGrC 2009. LNCS, vol. 5908, pp. 344–351. Springer, Heidelberg (2009)
17. Naveen, N., Ravi, V., Raghavendra Rao, C.: Data Mining via Rules Extracted from GMDH: An Application to Predict Churn in Bank Credit Cards. In: Setchi, R., Jordanov, I., Howlett, R.J., Jain, L.C. (eds.) KES 2010, Part I. LNCS (LNAI), vol. 6276, pp. 80–89. Springer, Heidelberg (2010)
18. Li, Y., Liu, Y., Zhu, J.: Quantile Regression in Reproducing Kernel Hilbert Spaces. American Statistical Association 102(477) (2007)
19. Rosipal, R., Trejo, L.J., Cichicki, A.: Kernel principal component regression with EM approach to nonlinear principal components extraction, Technical Report, University of Paisley, Scotland, UK (2000)
20. Rosenblatt, F.: The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review 65(6), 386–408 (1958)
21. Schetinin, V., Schult, J.: Learning polynomial networks for classification of clinical electroencephalograms. Soft Computing 10, 397–403 (2006)
22. Scholkopf, B., Smola, A.J., Muller, K.R.: Nonlinear component analysis as a kernel eigen value problem. Neural Computation 10(5), 1299–1319 (1998)
23. Scholkopf, B., Smola, A.J.: Learning with Kernels. MIT Press (2002)
24. Srinivasan, D.: Energy demand prediction using GMDH networks. Neurocomputing 72(1-3), 625–629 (2008)
25. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, New York (1995)
26. Li, Y., Liu, Y., Zhu, J.: Quantile Regression in Reproducing Kernel Hilbert Spaces. American Statistical Association 102(477) (2007)
27. Zhu, J., Hastie, T.: Kernel logistic regression and the import vector machine. In: Advances in Neural Information Processing Systems, vol. 14 (2001)

An Efficient Algorithm for Gray Level Image Enhancement Using Cuckoo Search

Sanjay Agrawal and Rutuparna Panda

Department of Electronics & Telecommunication Engineering
VSS University of Technology, Burla, Odisha, India
agrawals_72@yahoo.com

Abstract. This paper presents an efficient algorithm for gray level image enhancement using Cuckoo search (CS). The results are compared with Particle Swarm Optimization (PSO). The basic idea is to treat image enhancement as an optimization problem and then solve it using CS. It is observed that the proposed method provides better results than existing techniques.

Keywords: Cuckoo Search, Image Enhancement, PSO.

1 Introduction

The process of improving the visual quality of an image such that the resultant image is more informative; and well suited for specific image processing applications or machine perception; is called image enhancement. It finds numerous applications in the field of bio-medical imaging and remote sensing. Many studies can be found in the literature on various image enhancement techniques. Based on these studies, four classes of image enhancement operations are identified: point operation, spatial operation, transformation and pseudo-coloring [1]. We have used the spatial operation of image enhancement in this paper. Spatial operation deals with images in spatial domain based on processing of individual pixels in the image. Many images usually suffer from poor contrast and that's why contrast improvement has been the focus of researchers. Histogram equalization and contrast manipulations are some of the known methods of contrast improvement. But recently, optimization techniques have also been introduced in image enhancement. Both hardware devices and software algorithms can be used for image enhancement. Usually a software algorithm is used to manipulate the histogram of the image for improvement in the contrast and information. Since it is pixel based, a transformation function is defined to alter the histogram of the image such that the contrast is improved. Many studies are available in the literature to support the use of optimization techniques in image enhancement [2-5]. We are using here Cuckoo search algorithm for gray level image contrast improvement. The results are then compared with PSO.

The rest of this paper is organized as follows. Section 2 gives the formulation of image enhancement problem. Section 3 introduces the framework of CS algorithm.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 82–89, 2012.

The proposed scheme is described in Section 4. Section 5 gives simulation results and Section 6 gives the conclusion.

2 Image Enhancement Problem

The basic purpose of image enhancement is to improve the visual quality of the image so that it can be better suitable for human perception or machine interpretation. As presented earlier, contrast improvement by modification of gray level is a very effective way of image enhancement. The main problem in this technique is to formulate a transformation function or a mapping operator for obtaining the desired result. The transformation function may be linear or non-linear. We are using a non-linear function to achieve our goal. Image enhancement using a transformation function in spatial domain can be represented as:

$$g(x, y) = T[f(x, y)] \tag{1}$$

Where $f(x,y)$ is the input image pixel, $g(x,y)$ is the enhanced image pixel and T[·] is the transformation operator on f. Because of improvement in image acquisition techniques, improved image enhancement techniques are needed. Some techniques do not need contrast measure for enhancement, whereas some techniques need local contrast measure as a criteria function for improving the image. The contrast of the image is improved by defining an appropriate objective function, which is optimized by Cuckoo search algorithm. The objective function is defined as in [6].

3 Cuckoo Search Algorithm

There are various types of algorithms for any parametric optimization purpose. Out of them many are based on the natural behavior of animals in their habitats. Some techniques are Genetic Algorithm, Artificial Bee Colony, Particle Swarm Optimization, Bacteria Foraging, etc. Here we are using the Cuckoo Search algorithm for optimizing the objective function in gray level image enhancement process and compare the results with that of PSO technique. Yang and Deb [7] have developed the cuckoo search algorithm based on the parasitic breeding behavior of the cuckoo bird. The cuckoo bird depends on other host bird's nest for laying its eggs. The host bird nurtures the egg assuming it as its own. There is also a probability of discarding the egg if the host bird recognizes the alien egg. Then the host birds either throw away the egg or dumps the old nest to build a new nest at a new location. The basic objective of cuckoo search algorithm is to find the best nest where the probability of hatching of an egg is maximum. The algorithm consists of the following steps.

1. A generation is represented by set of host nests. The best nest which carries the egg is the solution.
2. The best nests in each generation are carried over to next generation.

3. A nest with a fitness value is randomly chosen as best nest. The fitness of each nest is then compared with best nest. If the current nest is better than the best nest, it replaces the best nest.
4. A probability p_a is chosen which represents the possibility of recognizing the cuckoo's egg by the host bird. Such nests are declared worst nests and discarded from further calculations.

4 Proposed Scheme for Image Enhancement

The contrast of the image is improved by defining an appropriate objective function, which is optimized by Cuckoo search algorithm. The objective function is defined as [6]:

$$O(z) = \log(\log(E(I(z)))).\frac{n_e(I(z))}{M.N}H(I(z)) \tag{2}$$

and

$$g(i,j) = \left[d.\frac{M}{\sigma(i,j)+b} \right].[f(i,j) - c.m(i,j)] + m(i,j)^a \tag{3}$$

where $m(i,j)$ is the gray level local mean, $\sigma(i,j)$ is the gray level local standard deviation; $O(z)$ is the objective function; $I(z)$ is the enhanced image using equation (3); $E(I(z))$ is the intensity of edges detected with an appropriate edge detector applied to the enhanced image; we use here the Sobel edge operator; n_e is the number of edge pixels as detected by the edge detector; M and N is the size of the image; $H(I(z))$ is the entropy of the enhanced image; a, b, c, d are the parameters that needs to be optimized to get an enhanced image. The cuckoo search technique is used here to maximize $O(z)$.

The transformation function as in equation (3) contains four parameters a,b,c,d which aid in the enhancement process. The main objective is now to find the best set of values for these four parameters within their range which will give an enhanced image. This is here done by cuckoo search algorithm and the results are compared with PSO. The input image is taken and the number of nests, probability of detection of alien egg, number of iterations is initialized. Simple bounds of the search domain are initialized. For each nest, a,b,c,d parameters are randomly initialized. The best set of parameters is obtained using cuckoo search. For each set, an enhanced image is generated. The quality of the image is evaluated by the objective function as defined by equation (2). At the end of the iterations, we get the enhanced image corresponding to the set of parameters obtained from the best nest. The termination condition in the proposed method is the number of iterations. The flowchart is displayed in Fig.1.

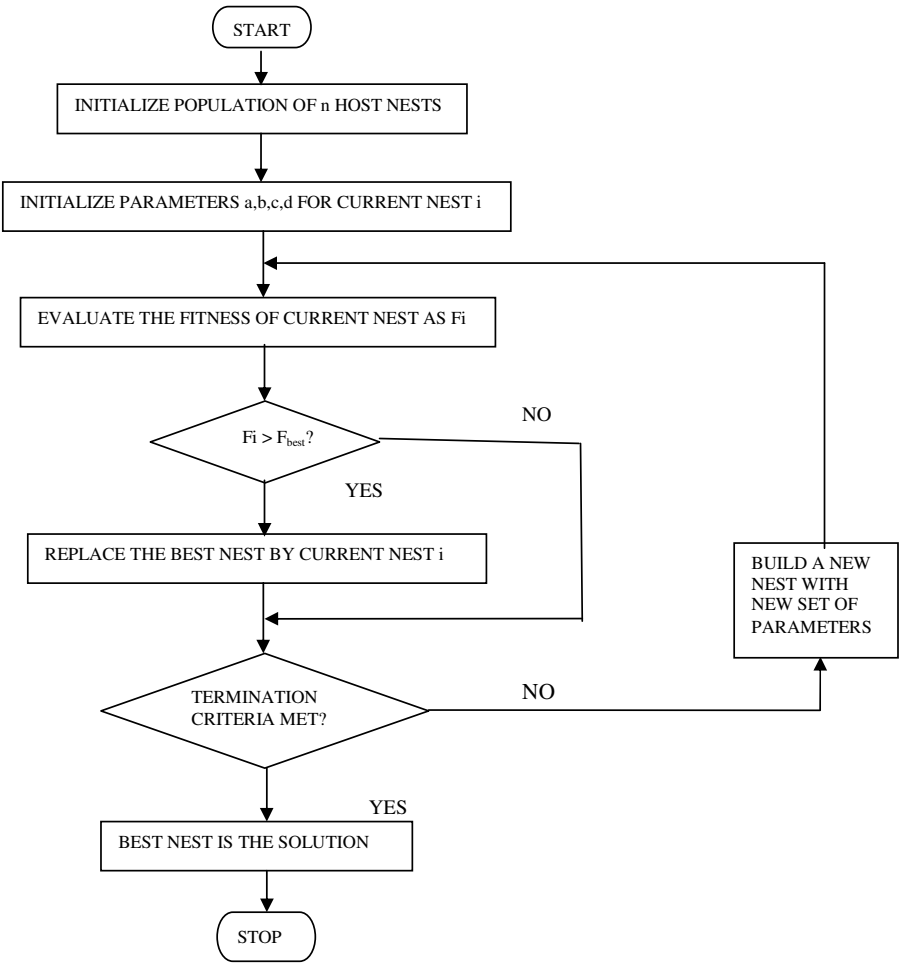


Fig. 1. Flow chart of cuckoo search algorithm

5 Simulation Results

In order to demonstrate the effectiveness of the proposed method, four gray level images are taken as shown in Fig. 2 and Fig. 3. It is observed that images enhanced with CS algorithm are visually better than PSO. Table 1 illustrates the comparative performance of image enhancement using both CS and PSO. The parameters used in the transformation function, as in (3), have their range as,

$$\langle a \in [0.1.5], b \in [0, 0.5], c \in [0,1], d \in [0.5,1.5] \rangle \text{ [8-9].}$$

TEST IMAGES

a. Original Lady Image

d. Original Duck Image

b. Image enhanced using PSO

e. Image enhanced using PSO

c. Image enhanced using CS

f. Image enhanced using CS

Fig. 2. Results of Lady Image and Duck Image enhanced using CS and PSO

a. Original Hut Image

b. Image enhanced using PSO

c. Image enhanced using CS

d. Original Building Image

e. Image enhanced using PSO

f. Image enhanced using CS

Fig. 3. Results of Hut Image and Building Image enhanced using CS and PSO

The number of nests for cuckoo search is taken 15, probability of recognizing an alien egg is taken 0.01 and number of iterations is taken 25. For calculating local mean and standard deviation the window size is chosen 5. The number of particles is 15 and the number of iterations for PSO is taken 25. Other parameters are taken standard values. As given in Table 1, we are maximizing the objective function and the proposed method gives a higher value. The entropy obtained by the proposed method is also better than PSO. Feature Similarity Index (FSIM) indicates the similarity between two images [10]. Higher is this value, better is the enhancement. It is higher in the proposed method. For maximizing the objective function as in equation (2); the relative number of pixels in the edges of the image is higher. It is observed that the number of edge pixels is higher in our proposed method.

Table 1. Performance comparison of CS and PSO method for image enhancement

Sl.No.	IMAGE	FITNESS VALUE		ENTROPY		FSIM		SOBEL EDGE	
		CS	PSO	CS	PSO	CS	PSO	CS	PSO
1	LADY	15.3317	14.0504	7.5964	7.0879	0.9964	0.9940	1413	1167
2	DUCK	15.6568	14.1908	7.6188	7.3856	0.9946	0.9626	1912	1229
3	HUT	15.6929	14.9690	7.7402	7.7154	0.9972	0.9746	1498	1422
4	BUILDING	15.6883	13.7073	7.6901	6.4197	0.9915	0.9816	1549	1435

6 Conclusion

This paper has demonstrated that the proposed method outperforms the PSO based enhancement technique both qualitatively and quantitatively. The claim is justified by the quantitative measures as given in Table 1. Future work includes some more examples to support the proposed method. Also some more comparisons with other swarming methods like GA may be included in the future. The technique proposed here may be useful for image enhancement. Image enhancement using cuckoo search seems to be a promising technique for remote sensing, object recognition and fusion applications.

References

1. Gonzalez, R.C., Woods, R.E.: Digital Image Processing. Addison-Wesley, New York (1987)
2. Munteanu, C., Rosa, A.: Evolutionary image enhancement with user behavior modeling. ACM SIGAPP Applied Computing Review 9(1), 8–14 (2001)
3. Saitoh, F.: Image contrast enhancement using genetic algorithm. In: Proc. IEEE SMC, Tokyo, Japan, pp. 899–904 (1993)
4. Pal, S.K., Bhandari, D., Kundu, M.K.: Genetic algorithms for optimal image enhancement. Pattern Recognition Letter 15, 261–271 (1994)
5. Jingquan, S., Mengyin, F., Chanjian, Z.: An image enhancement algorithm based on chaotic optimization. Computer Engineering and Applications 27, 4–6 (2003)

6. dos Santos Coelho, L., Sauer, J.G., Rudek, M.: Differential evolution optimization combined with chaotic sequences for image contrast enhancement. Chaos, Solitons and Fractals 42, 522–529 (2009)
7. Yang, X.S., Deb, S.: Engineering optimization by cuckoo search. Int. J. Mathematical Modelling and Numerical Optimization 1(4), 330–343 (2010)
8. Munteanu, C., Rosa, A.: Gray-scale enhancement as an automatic process driven by evolution. IEEE Transaction on Systems, Man and Cybernetics-Part B: Cybernetics 34(2), 1292–1298 (2004)
9. Gorai, A., Ghosh, A.: Gray level Image enhancement by Particle Swarm optimization. In: World Congress on Nature & Biologically Inspired Computing, pp. 72–77 (2009)
10. Zhang, L., Zhang, L., Mou, X., Zhang, D.: FSIM: a feature similarity index for image quality assessment. IEEE Transactions on Image Processing 20(8), 2378–2386 (2011)

A Multi-objective Workspace Optimization of 3R Manipulator Using Modified PSO

Sumanta Panda[1], Debadutta Mishra[1], and B.B. Biswal[2]

[1] Department of Manufacturing Science & Engineering
VSS University of Technology, Burla, Odisha, 768018
{sumanta.panda,dmvssut}@gmail.com
[2] Department of Industrial Design,
National Institute of Technology, Rourkela 769008
bbbiswal@nitrkl.ac.in

Abstract. The manipulator capability of a robot largely depends on the workspace (WS) of the manipulator apart from other parameters. With the constraints in mind, the optimization of the workspace is of prime importance in designing the manipulator. The workspace of manipulator is formulated as a constrained optimization problem with workspace volume as objective function and workspace volume and maximum manipulator size as a multi-objective function. It is observed that the previous literature is confined to use of conventional soft computing algorithms only, while a new search modified algorithm is conceptualized and proposed here to improve the computational efficiency. The proposed algorithm gives a good set of geometric parameters of manipulator within the applied constrained limits for both mono and multi-objective optimization. The efficiency of the proposed approach to optimize the workspace of 3R manipulators is exhibited through two cases.

Keywords: Industrial robot, Manipulators, Workspace, Constraints.

1 Introduction

The manipulator workspace is defined as the region of reachable points by a reference point H on the extremity of a manipulator chain, Gupta and Roth [1]. Generally speaking, a robot manipulator structure can be subdivided into a regional structure and orientation structure. The regional structure consists of the arm, which moves the end-effectors to a desired position in the workspace of the robot manipulator. The orientation structure comprised of the links that, rotates the end effectors to the desired orientation in the workspace. In this study, the regional structure of the robot manipulators is examined rather than the orientation structure. The workspace volume should be computed with high accuracy as it influences the manipulator's dimensional design, its positioning in the work environment, and its dexterity to execute tasks. In this approach the design of manipulators with three-revolute joints (3R) is reformulated as an optimization problem that takes into account the characteristics of the workspace. This research proposes an algebraic formulation to estimate the cross section area, and workspace volume.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 90–97, 2012.

Ceccarelli [3] presented an algebraic formulation to determine the workspace of revolution manipulators. The formulation is a function of the dimensional parameters in the manipulator chain and specifically of the last revolute joint angle, only through mathematical model of workspace developed by Ceccarelli [3] is of crucial importance, however the manipulators optimal design is not considered. Lanni et al. [4] investigated and solved the design of manipulators modeled as an optimization problem that takes into account the characteristics of the workspace. Wenger [5] demonstrated that it is possible to consider a manipulator's execution of non-singular changing posture motions in the design stage. Ceccarelli and Lanni [6] presented a suitable formulation for the workspace that can be used in the design of manipulators, which was formulated as a multi-objective optimization problem using the workspace volume and robot dimensions as objective functions. Bergamaschi et al.[2,7] presented the condition for the regularity and parameterization of the envelop, penalties are imposed to minimize the voids and control the total area.

An effort has been made to include the regularity, inequality and limiting constraints separately and simultaneously. Taking into account the algebraic, formulation the optimization problem has been solved as a mono-objective and multi-objective using PSO. Finally, the workspace volume for diverse cases obtained by using modified PSO is reported and the results are compared with the results given by Bergamaschi et al. [7] using modified SQP, DE, GA and PSO. The innovative parts of this work are: avoidance of constraints violation, determination of total void cross section area and use of modified PSO for optimization of workspace volume. Finally, the results obtained in diverse cases are compared with the available standard results.

2 Workspace Volume Formulation

The design parameters for the link size are represented as a_1, a_2 ,a_3, d_2, d_3, α_1, α_2 (d_1 is not meaningful, since it shifts the workspace up and down). Here, *1* and reference *0* have the same origin, the transformation matrices of a reference on the former. The workspace of a three-revolute open chain manipulator can be given in the form of the radial reach r and axial reach z with respect to the base frame, according to Ceccarelli and Lani [6]. The axial z and radial r coordinates of the boundary points are given as the solution for the system Eq.(1), assuming that $sin\alpha_1 \neq 0$, $C \neq 0$, and $E \neq 0$. After some algebraic manipulation we can see that

$$\left.\begin{aligned} z &= [-FG \pm (-E^2Q)^{1/2}]/C(E^2+G^2) - D/C \\ r &= \{[(Cz+D)G+F]/(E+A-z^2)\}^{1/2} \end{aligned}\right\} \quad (1)$$

where, $Q=B(E^2+G^2)+F^2$

Eq.(1) that represents an envelope is a only function of the parameter θ_3.This research proposes numerical formulation to approximate the cross-sectional area, through its discretization within a rectangular mesh. Initially, the extreme values of vectors r and z should be obtained as

$$r_{min} = \min(r);\ r_{max} = \max(r);\ z_{min} = \min(z);\ z_{max} = \max(z); \quad (2)$$

The points that are belong to the workspace is identified by $P_{ij} = 1$, otherwise $P_{ij} = 0$, which means:

$$P_{ij} = \begin{cases} P_{ij} = 1, if, P_{ij} \in W(H) \\ P_{ij} = 0, if, P_{ij} \notin W(H) \end{cases} \tag{3}$$

where W(H) indicates workspace region. In this way, the total area is obtained by the sum of every elementary area of the mesh that are totally or partially contained in the cross section. In Eq. (3), it is observed that only the points that belong to the workspace contribute to the calculation of the area:

$$\mathrm{A_{r,z}} = \sum_{i=1}^{i_{\max}} \sum_{j=1}^{j_{\max}} (P_{ij} \Delta r \Delta z) \tag{4}$$

The coordinates of the centre of the mass is calculated considering the sum of the centre of the mass of each elementary area, divided by the total area, using the following equation:

$$r_g = \frac{\sum_{i=1}^{i_{\max}} \sum_{j=1}^{j_{\max}} (p_{ij} \Delta r \Delta z)\left((i-1)\Delta r + \frac{\Delta r}{2} + r_{\min}\right)}{A_T} \tag{5}$$

Finally, after the calculation of the cross sectional area and the coordinates of the centre of the mass, given by Eqs. (4) and (5), the workspace volume of the manipulator can be evaluated as presented by Beer and Johnston [8] is formulated as

$$V = 2\pi r_g A_T \tag{6}$$

The optimum design can be formulated as a multi-objective optimization problem and expressed in the form

$$\text{Max } F_c = w_1 * V(x)/(V_0) + w_2 * (-L_{max}) /(L_0) \tag{7}$$

Where V(x) is the workspace volume and L_{max} is the maximum dimension of manipulator. Weighting coefficients w_1 and w_2 where V_0 and L_0 are determined by obtaining attainable maxima and minima for both the objective functions separately.

3 Synthesis for Void Cross Section Area

The main purpose of design of manipulator is to come up with a dimensional synthesis of 3R manipulators. The objective of the optimization problem is to maximize the workspace volume, thereby obtaining the best dimensions. As the workspace volume depends on the radial cross sectional area so it becomes essential to estimate the void cross section area.

$$A_{max} = (r_{max} - r_{min}) * (z_{max} - z_{min}) \tag{8}$$

In this work Amax is obtained by considering

$$\left.\begin{aligned} \mathrm{r}_{\min} &= \mathrm{z}_{\min} = 0 \\ r_{\max} &= z_{\max} = L_{\max} \end{aligned}\right\} \tag{9}$$

$$A_{reachable} = (A_T - A_{void}) \tag{10}$$

$$A_{void} = 0 \text{ when ring void does not exist. } A_{TV} = (A_{max} - A_{reachable}) \tag{11}$$

Where A_{TV} is total void cross section area.The different limiting constraints used are as follows

$$\left.\begin{aligned} &0.01 < a_i < a_i^u \; for \; i = 1,2,3,..... \\ &0.01 < d_i < d_i^u \; for \; j = 2,3. \\ &0.05^0 < \alpha_k < 90^0 \; for \; k = 1,2. \end{aligned}\right\} \tag{12}$$

The regularity constraint is given as follows:

$$g_1 = \frac{\partial^2 f}{\partial \theta^2} \neq 0 \tag{13}$$

In this technique, the problems with constraints are transformed into unconstrained problems by adding a penalty function *p(x)* to the original objective function to limit constraint violations. This new objective function is called as pseudo-objective function. The pseudo-objective function is given by the form:

$$\phi(x) = F_c(x) + r_p * p(x) \quad r_p \text{ is the penalty factor} \tag{14}$$

$$p(x) = [\sum_{i=1}^{m} \{\max[0, g_i(x)]\}^2 + \sum_{i=1}^{p} [h_i(x)]^2] \tag{15}$$

where *Fc(x)* is the original objective function given in Eq. (14), *P(x)* is an imposed penalty function given by Eq. (15), g_i are the inequality constraints, h_i are the equality constraints. The scalar r_p is a multiplier that quantifies the magnitude of the penalty.

4 Proposed Algorithm

Particle swarm optimization (PSO) is a population-based evolutionary algorithm originally presented by Kennedy and Eberhart [9]. The particle swarm concept originated as a simulation of a simplified social system. The original intent was to graphically simulate the graceful but unpredictable choreography of a bird flock. However, in PSO each potential solution is also assigned a randomized velocity, and the potential solutions, called particles, are then "flown "through the problem space. Each particle keeps track of its coordinates in the problem space which are associated with the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called pbest. Another "best "value that is tracked by the global version of the particle swarm optimizer is the overall best value, and its location,

obtained so far by any particle in the population. This location is called g_{best}. The particle swarm optimization concept consists of, at each time step, changing the velocity (accelerating) each particle toward its p_{best} and g_{best} locations (global version of PSO). Acceleration is weighted by a random term, with separate random numbers being generated for acceleration toward p_{best} and g_{best} locations. There is also a local version of PSO in which, in addition to p_{best} , each particle keeps track of the best solution, called best, attained within a local topological neighbourhood of particles. The process for implementing the modified version of PSO so as to expedite the optimization process is as follows:

During the velocity and position update stages if the limiting constraints on link size are not satisfied at the beginning of the iteration or at the end of the iteration then that particular value of the fitness function is skipped and control goes to next iteration. If the limiting constraints on link size are not satisfied at any iteration other than beginning and end of the iteration then the average value of DH parameter of two previous iterations near local best value is taken and the fitness function is evaluated. The Pseudo-code for modified PSO is as shown in Fig.1.

```
                                        % 1. Define problem hyperspace
% Define maximum and minimum range for parameters
                    % 2. initialize the swarm
D and  NP = D * 4                       w, c1 and  c2 and iteration
                                        %3. evaluate the initial particles
                                          to find local and global best
for i = 1 : NP    for j = 1:D  x_old(i, j)  = range_min(D) +((range_max(D)-range_min(D))*(rand));  end
                                        % 4. start iteration
for iter = 1 : iterations       for i = 1 : NP          value(i)= conf_testworkspace_test11pso(x_old(i,:));
   end
for i = 1 : NP   if value(i) > lbest(i)   lbest(i)  = value(i);   xbest(i,:) = x_old(i,:);
      end    end             gbest1 = max(lbest); %Global best value
   if gbest1 > globalbest      globalbest = gbest1; index = find(lbest == max(lbest));
     global_xbest = x_old(index(1),:);    global_xbest_mtx = (repmat(global_xbest,[NP,1]));   end

                    % 5. update velocity and position
  for i = 1 : NP        for j = 1
 velocity(i,j) = w*velocity(i,j) + c1*rand*(xbest(i,j) - x_old(i,j))+    c2*rand*(global_xbest_mtx(i,j) - x_old(i,j));
            x(i,j) = x_old(i,j) + velocity(i,j);   end     end
   w = 0.9-0.4*(iter/iterations);    [iter globalbest];    globalbest1 = [globalbest1, globalbest];
                    for i = 1 : NP
. if  x(i,j) satisfied limiting conditions (Eq. 25)   then    value(i)= conf_testworkspace_test11pso(x(i,:));
        Else       if   i==1 or i= NP-1;  then   iter = iter+1    (skip)
                    else
x(i,j) = [{xbestt(i-1,D) + xbest (i +1,D)}/2]; x_old (i,j)=  x(i,j)       end                 end
       for i = 1 : NP              if value(i) > lbest(i)     lbest(i) =  value(i);     xbest(i,:) = x(i,:);
                                    end                        end
     gbest1 = max(lbest);           if gbest1 > globalbest
    globalbest = gbest1;   index = find(lbest == max(lbest));
   global_xbest = x(index(1),:);  global_xbest_mtx = (repmat(global_xbest,[NP,1]));
                                    end
         ;  end
% end of iteration
```

Fig. 1. Pseudo-code for modified PSO

5 Simulation Results

The optimization problem is investigated using the proposed PSO algorithm. The algorithm code was developed in MATLAB by the authors. The results of the final

parameters, final volume and time of execution which are the best value obtained are represented. The optimal radial cross section for each case is as illustrated.

Case-1: Optimization with Only Regularity and Limiting Constraint

The mono and multi objective function of the optimization problem are given by Eq. (6) and (7). The limiting constraints is given by Eq. (14) i.e. considering $a_i'' = d_i'' < 1$ and the regularity constraint as given by Eq. (15). The optimization problem is investigated using the Particle Swarm Optimization through application of Matlab code. The result for this case is as shown in Table 2. The radial cross section of the workspace volume after optimization is as shown in Fig. 2 & 3. The convergences are shown in Fig. 4 &5. The workspace volume obtained for mono and multi-objective optimization are 107.35.and 120.39 unit of volume (u.v.) and computational time are 91.56 &110.1 min.

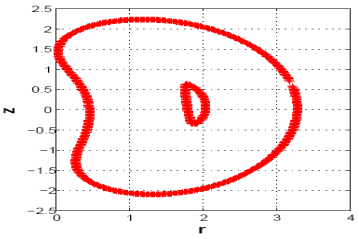


Fig. 2. Optimal radial section for Mono-objective Optimization for Case-1

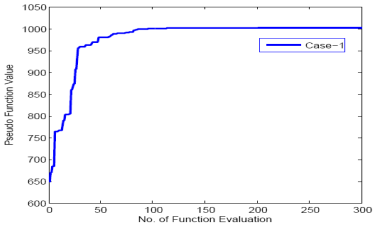


Fig. 4. Performance Characteristics of Mono-objective Optimization algorithm for Case-1

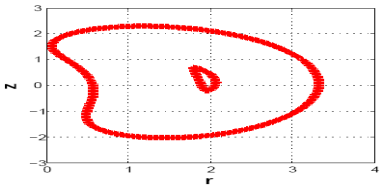


Fig. 3. Optimal radial section for Multi-objective Optimization for Case-1

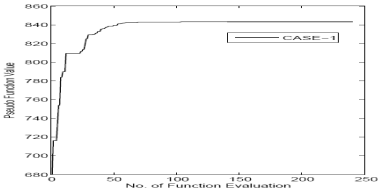


Fig. 3. Performance Characteristics of Multi-objective Optimization algorithm for Case-1

Case-2: Optimization without Imposing Any Constraints

The mono and multi-objective function of the optimization problem are given by Eq. (6) and (7). The limiting constraints is given by Eq. (14) i.e. considering $a_i'' = d_i'' < 1$. The optimization problem is investigated using the Particle Swarm Optimization through application of Matlab code. The result for this case is as shown in Table 1. The radial cross section of the workspace volume after optimization is as shown in Fig. 6 & 7. The convergences are shown in Fig. 8 &9. The workspace volume obtained for mono and multi-objective optimization are 125.42.and 133.89 unit of volume (u.v.) and computational time are 164.20 &183.5 min respectively.

Table 1. Comparisons of Results

Cases	Optimum DH Parameters							Volume [u.v.]	Void Cross section area	Time (Min)
	a_1 [u.m]	a_2 [u.m]	$_3$ [u.m]	d_2 [u.m]	d_3 [u.m]	α_1 (deg.)	α_2 (deg.)			
Mono-Objective Optimization										
Case-1	0.99	0.99	0.99	0.71	0.99	80.03	50.01	107.35	0.30	91.56
Case-2	0.99	0.99	0.97	0.98	0.99	82.0	75.2	125.42	0.71	164.2
Multi-Objective Optimization										
Case-1	0.99	0.99	0.99	0.27	0.99	81.37	52.28	120.39	0.21	110.1
Case-2	0.99	0.99	0.99	0.99	0.99	76.88	68.43	133.89	0.64	183.5

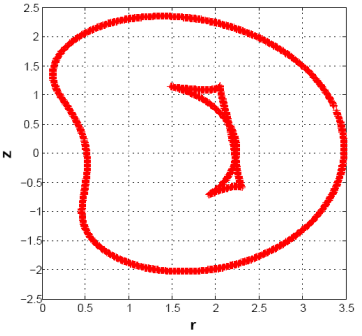


Fig. 6. Optimal radial section for Mono-objective Optimization Case-1

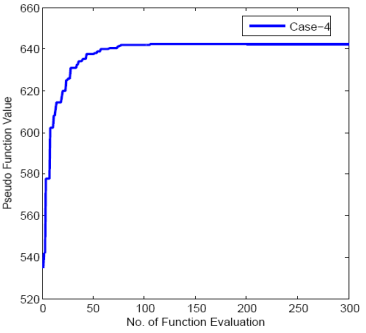


Fig. 8. Performance Characteristics of Mono-objective Optimization algorithm for Case-1

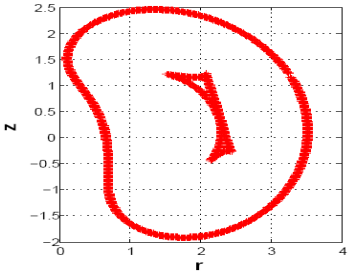


Fig. 7. Optimal radial section for Multi-objective Optimization Case-1

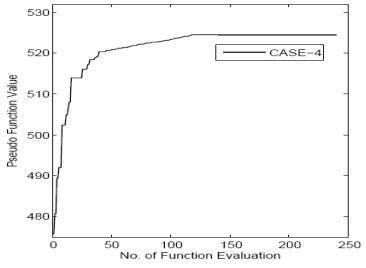


Fig. 9. Performance Characteristics of Multi-objective Optimization algorithm for Case-1

6 Conclusions

The proposed optimum design of 3R manipulators has been formulated as a multi-objective optimization problem by using workspace characteristics and size of manipulators as objective functions. The algebraic formulation has been easily adjusted to the software package requirements so that an interesting procedure has been obtained in term of an easy and efficient design procedure. Diverse cases are presented imposing different constraints to demonstrate the efficiency of the proposed algorithm. It may be noted that the optimum workspace volume depends mainly on angles α_1 and α_2. Our empirical studies with the diverse cases show that multi-objective optimization formulation is more effective as compared to mono-objective optimization formulation. Table-2 shows that the workspace volume in case-1 and that in case-2 is increased by 12.14%, and 6.75% in multi-objective optimization, where as the computational time are increased by 16.83%, and 10.51% respectively. The total void cross section area is decreased by 30% and 10.93% in multi-objective optimization formulation for case-1 and case-2 respectively. It is further realized that even in case of unconstrained problem as in case-2, the computational time is very high and the workspace volume is also high as compared to other case. These results as shown in Table- 2 encourage researchers in this particular area to develop hybrid techniques which will further reduce the computational time for this problem.

References

[1] Gupta, K.C., Roth, B.: Design considerations for manipulator workspace. Journal of Mechanical Design 104, 704–711 (1982)
[2] Bergamaschi, P.R., Nogueira, A.C., Saramago, S.F.P.: Design and optimization of 3R manipulators using the workspace features. Applied Mathematics and Computation 172(1), 439–463 (2006)
[3] Ceccarelli, M.: A formulation for the workspace boundary of general Nrevolute manipulators. Mechanism and Machine Theory 31(5), 637–646 (1996)
[4] Lanni, C., Saramago, S.F.P., Ceccarelli, M.: Optimal design of 3R manipulators using classical techniques and simulated annealing. Revista Brasileira de Ciências Mecânicas 24(4), 293–301 (2002)
[5] Wenger: Some guidelines for the kinematic design of new manipulators. Mechanism and Machine Theory 35, 437–449 (2000)
[6] Ceccarelli, M., Lanni, C.: A multi-objective optimum design of general 3R manipulators for prescribed workspace limits. Mechanism and Machine Theory 39, 119–132 (2004)
[7] Bergamaschi, P.R., Saramago, S.F.P., Coelho, L.S.: Comparative study of SQP and metaheuristics for robotic manipulator design. J. Applied Numerical Mathematics 58, 1396–1412 (2008)
[8] Beer, F.P., Johnston Jr., E.R.: Vector Mechanics for Engineers: Statics and Dynamics, 3rd edn. Mc Graw Hill, New York (1977)
[9] Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proc. IEEE International Conf. on Neural Networks, Perth, Australia. IEEE Service Center, Piscataway (1995)

An Analysis of Genetic Algorithm Based Anycast Routing in Delay and Disruption Tolerant Networks

Éderson R. Silva and Paulo R. Guardieiro

Faculty of Electrical Engineering, Federal University of Uberlandia, Uberlandia (MG), Brazil
ersilva@eletrica.ufu.br, prguardieiro@ufu.br

Abstract. Populations in developing countries, especially in regions that lack telecommunications infrastructure, usually do not have access to the information technology. Instead, Delay-/Disruption-Tolerant Networks (DTNs) have the capacity to interconnect areas that are underserved by traditional networks. Anycast routing can be used for many applications in DTNs, and it is useful when nodes wish to send messages to at least one, and preferably only one, of the members in a destination group. In this paper, aiming an efficient routing, it is analyzed a Genetic Algorithm (GA) based anycast routing algorithm. Simulation experiments show that the proposed algorithm can produce good results in typical scenarios including delays and disconnections in message delivery.

Keywords: Anycast routing, DTNs, genetic algorithms, subpopulation.

1 Introduction

It has been awhile since the Internet has become very popular due to its flexibility, capacity, and robustness offered by the success of protocols used, such as TCP/IP. However, as the Internet is used more and more for communication, new challenges arise. In this sense, there is a growing effort to allow communications in networks whose scenarios involve frequent disruption and/or long and uncertain delays, thus requiring a Delay-/Disruption-Tolerant Network (DTN) architecture. Many of the principles and challenges of DTN architecture are reviewed in [1].

One of the main challenges that arises in the design of DTNs is routing. Zhang reviewed some of the routing protocols and categorized them as deterministic and stochastic [2]. In this paper, the network is modeled for the problem of providing data communication to remote and rural areas. These regions, possibly disconnected, can use a car, bus, motorbike, and/or truck, equipped with a storage device, to act as a carrier (mobile nodes that exploit device mobility are used to enable the communication) to deliver messages (Fig. 1). Moreover, it is assumed that the network topology may be known ahead of time, and it is used an evolving graph to represent the DTN.

When nodes wish to send messages to at least one, and preferably only one, of the members in a destination group the semantic is anycast. Due to resource constraints in DTNs, anycast delivery is appropriate and shows promise in many applications [3], such as disaster rescue field, content distribution, long distance education, etc.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 98–105, 2012.

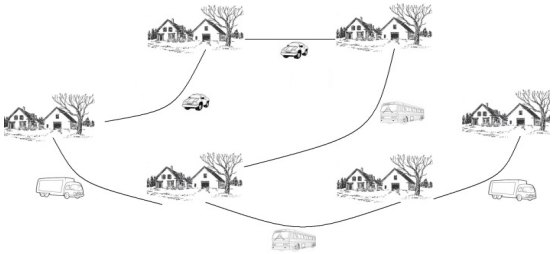


Fig. 1. Typical scenario

A critical issue in DTN routing is to find the appropriate combination of routes taking into account the node storage constraints and the network traffic dynamics (this problem is NP-complete). Thus, an anycast routing algorithm making use of genetic algorithms (GAs) is analyzed, because GAs have the ability to solve complex optimization problems. Moreover, the DTNs can tolerate longer delays than the elaboration time required for GAs to converge toward the optimal solution. Hence, DTNs can be a good application field for GAs. The efficiency of the proposed algorithm is evaluated using modeling and simulation. To improve the performance of the algorithm we limit the number of solutions to be evaluated. Among the main contributions, we have a detailed description and analysis of the proposed GA-based anycast routing algorithm, including the concept of subpopulation (the next generation population is produced using four subpopulations), distribution of network traffic and scalability.

The remainder of this paper is organized as follows: Section 2 shows related works and Section 3 the proposed GA-based anycast routing algorithm. Section 4 presents the simulation and discusses the results. At last, Section 5 presents conclusions.

2 Related Works

According to [2] most DTN routing algorithms deal with unicast service. However, in this case, the destination is fixed and is determined when the message is generated. On the other hand, the anycast service is appropriate to take the opportunity to send messages to only one destination, possibly the one that provides the best communication opportunities among the nodes in a destination group, allowing communication in scenarios where the unicast service would be impracticable. Therefore, a few unicast routing algorithms developed [4] can be adapted to perform anycast service.

Gong et al [3] analyzed the anycast semantic for DTNs. The authors assumed that nodes in the network were stationary. The connectivity among the nodes was the mobile devices that act as carrier to deliver messages for the nodes. Thus, we use the same DTN model showed in [3], but to incorporate the network traffic and the storage constraint (not considered in [3]), information about network topology are used.

Many researchers have applied GA to various types of network routing problems [5]. However, those approaches are beyond the scope of this paper. Some of concepts addressed in these previous works are used to find the combination of routes of each anycast session with optimized rate of delivery and delay (see Section 3).

To our knowledge, GA applied to search combination of anycast routes for performance optimization in DTNs is a promising approach and has not been very well explored yet. The use of GAs in DTNs was introduced in [6]. In order to make the proposal more attractive in terms of time required to converge, the concept of subpopulation and a limited number of potential solutions in isolation are used [7]. However, it is necessary a more detailed analysis and description of this GA-based solution, including the scalability and the concept of subpopulation.

3 Proposed Genetic Algorithm Based Anycast Routing

GAs are defined as search algorithms based on the mechanics of natural selection and natural genetics [8]. In this paper, the main objective of GA is to assist in the anycast routing for route and destination decisions. Our proposed GA-based anycast routing algorithm uses two steps: a) a set of possible solutions for each session in isolation are first found; b) the combinations of these solutions are evaluated. These steps can be also seen in [9]. However, we use a different and optimized to do this [7].

3.1 Genetic Representation and Population Initialization

The chromosome is essentially a list of nodes along the path for all anycast sessions (z). Potential solutions are found by applying Dijkstra's algorithm to compute paths with the least number of hops. The method used to compute and limit the potential solutions in isolation is detailed in [7]. The adopted method for encoding potential solutions into chromosomes is based on associating each gene to each node forming the route between the source-destination couple (Fig. 2), where s_i (i=1,2,…,z) is a subset denoting the anycast source nodes, n_x are the intermediate nodes (x=m,n,…,q) and $k_{i,l}$ (i=1,2,…,z and l = number of intended destinations) is the destination group.

Each anycast session has its source node position represented by sp_i (constant for all individuals). To create the initial population a random initialization is adopted.

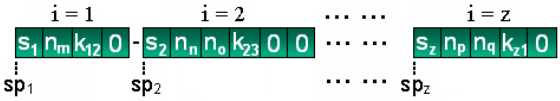


Fig. 2. Genotype coding

3.2 Fitness Function and Reproduction

To evaluate the fitness of chromosome we consider the performance metric delivery probability (*DP*) and delay *D*. *DP* is the rate of total number of unique anycast messages received by an anycast group member from each anycast session per total number of messages transmitted by each anycast source node. Delay *D* represents the delay for all traffic. A description of the proposed fitness function can be found in [7].

A characteristic of our proposed GA-based algorithm is that it searches routes with *DP* above a threshold (DP_{min}) and having the least delay *D*. Since different DTN

applications can be found, it is clear that a large number of values can be used for DP_{min}. We use the following strategy to sort the fitness of chromosomes:

```
for every individual at current generation do
  sort the individuals in descending order of DP
  for the individuals with DP above DP_min do
      sort the individuals in ascending order of delay
  end for; end for
```

We use the pairwise tournament selection, crossover and mutation to reproduction. To produce only regular individuals, the genetic operators use the positions sp_i as a cross point. Mutation is performed changing one route of the chromosomes.

3.3 Next Generation Population and GA-Based Approaches

We use the concept of subpopulation. This method is efficient and effective [10]. This way, four subpopulations are generated and produce the next generation population:

- Elitism reservation strategy: the chromosomes with the best fitness survive and are carried into the next generation;
- Changing one element of the chromosomes: replaces one route in the chromosomes with best fitness to form a new individual;
- The stochastic universal sampling: spin a roulette wheel;
- Complete random method: population is generated randomly.

We have two competing factors in the selection procedure: selection pressure and population diversity. When we use these four subpopulations we have different selection pressure, e.g. the stochastic universal sampling method increases selection pressure, and on the other hand, the complete random method decreases it.

At last, we set up control parameters, e.g., population size (50), crossover probability (0.8) and mutation probability (0.03). However, as we use the concept of subpopulation, the influence of these parameters is short. We propose a GA-based approach (GA1) that uses the concept of subpopulation. Then we consider a GA-based approach (GA2) that uses only the elitism reservation strategy to verify the effectiveness and efficiency of the concept of subpopulation. The GA-based algorithms are controlled by the number of generations (limited to 300) and we set DP_{min} to 92%.

4 Computational Experiments

In this paper, the DTN is represented by an evolving graph [11] and the link capacities are time-dependent. An edge between node *1* and *2* means that there exist some mobile devices moving from the initial node *1* to the terminal node *2*. *b*(*1*)-b(*2*) and *c*(*1*,*2*) are the storage capacity of nodes and mobile devices, respectively. Every mobile device has a moving delay *md*(*1*,*2*), and a leaving time *w*(*1*,*2*). Besides, the nodes in the network are stationary (remote/rural areas) and generate messages.

In our simulation, we employ the Waxman Network Topology Generator [12] to generate a random graph of nodes over a square coordinate grid. Initially, we set α to 0.4 (a density of short edges relative to longer one middle), and β to 0.25 (graphs with lower edge densities) for 40 nodes distributed in the area 1300 m x 1300 m.

We assume the communication between nodes is carried out by mobile devices (to simulate the behaviors of DTNs). $w(i,j)$ of mobile devices on each edge are random numbers from the Poisson distribution with mean interval time selected randomly from 600 to 6000 seconds. $md(i,j)$ on each edge is a number selected randomly between 60 and 600 seconds, which is multiplied by the distance $dist(i,j)$ between the nodes. The storage capacity $c(i,j)$ and $b(i)$ may vary from 500 to 800 and from 600 to 1000 messages, respectively. These values are similar to those found in [3].

Each anycast session can have between 2 and 5 possible destinations l. Each source node can send to the destination group between 300 and 500 messages. A message is split only at source node. For all simulated scenarios we consider 16 anycast sessions.

We compare the performance of GA1 and GA2 with the shortest path (SP) algorithm (the Dijkstra's algorithm computes the path with least number of hops between source and any destination) and the earliest delivery (ED) algorithm that computes the path in which each edge has a cost proportional to delay ($md(i,j)$ and $w(i,j)$). Both algorithms are described in [4] and we adapted the algorithms to the anycast service.

To analyze the distribution of the messages in the network, we define a performance metric that computes the average number of messages carried (AMC) per edge:

$$AMC = \sum_{i=1}^{z} m_k(i) \Big/ \sum_{i=1}^{z} h(i) \tag{1}$$

where for each anycast session, $m_k(i)$ is the total number of unique anycast messages received by an anycast group member and $h(i)$ is the total number of edges (hops).

4.1 Simulation Results

The results are the average over 10 runs with different random seeds and network topologies. In the first scenario, the area in which the 40 nodes are distributed is varied: 1300 m x 1300 m, 2600 m x 2600 m, and 5200 m x 5200 m.

Fig. 3 presents the DP obtained by the GA1, GA2 and SP algorithms. We plot the 95% confidence interval using a scale of 1:5. When we increase the area in which the nodes are distributed the moving delay increases too. SP algorithm presents the worst performance and GA1 gets results more robust than GA2 (lesser confidence interval).

Since SP algorithm computes the lesser number of hops the AMC per edge is large (Fig. 4 (a)). The decreasing in AMC is because when the moving delay of the mobile device increases, the time waiting for an opportunity to transmit is large. Thus, the GA-based algorithms need to avoid that a lot number of messages pass through the same edge, i.e. it searches alternative routes. This way, a larger number of edges are used and the AMC per edge decreases. The delay D increases for all algorithms due to the raising of moving delay, as showed in Fig. 4 (b), and GA1 and SP algorithms get the best results. However, this delay obtained by SP algorithm is to a low DP.

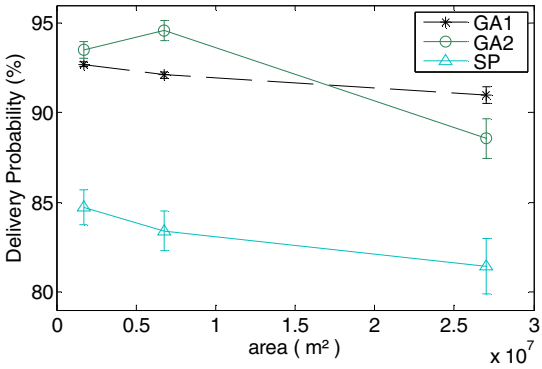
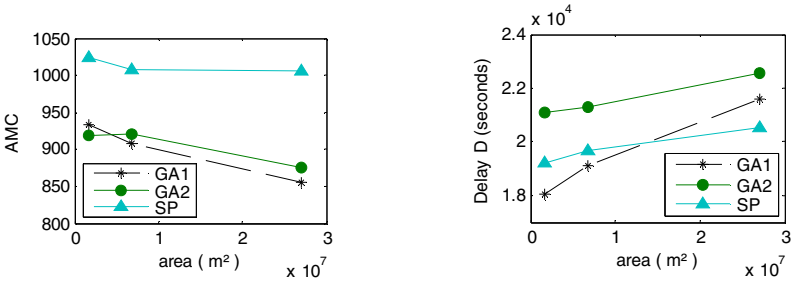


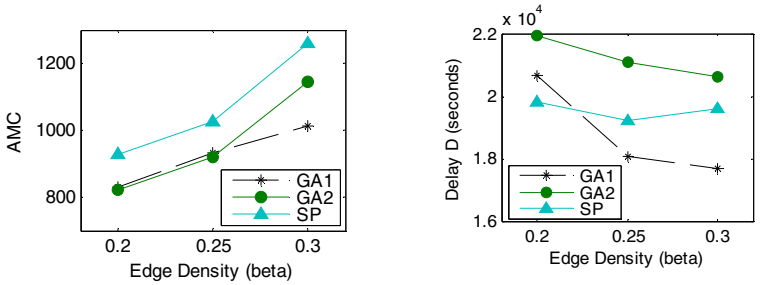
Fig. 3. Delivery probability for different areas



(a) Average number of messages carried per edge (b) Delay D

Fig. 4. Results for different areas

In the second scenario, we vary the edge density β using a constant area (1300m x 1300m). Again, *DP* results show that GA1 is less sensitivity to network variations than the GA2 and SP algorithms (these results were omitted). Besides, when the edge density is increased, the *DP* increases too because more mobile devices are available.



(a) Average messages carried per edge (b) Delay D

Fig. 5. Results for different edge densities

When we increase the edge density, the *AMC* per edge increases too (Fig. 5(a)). This is because with more edge, the messages can be routed through paths having a lesser number of hops. As result, the *AMC* per edge increases. GA1 algorithm uses

short numbers of *AMC* per edge and this is a good feature to nodes with low power. Moreover, Fig. 5 (b) shows that the delay *D* obtained by GA1 decreases more than other algorithms, because GA1 makes an efficient use of the increasing in the edges densities, i.e. GA1 is more effective in search the combination of routes than GA2.

In the third scenario, SP, ED and GA1 algorithms are analyzed using different numbers of nodes. Fig. 6 shows that GA1 obtains a better performance and results more robust than the other algorithms (lesser confidence interval). The 95% confidence interval is plotted using a scale of 1:2. The results are near to the DP_{min}.

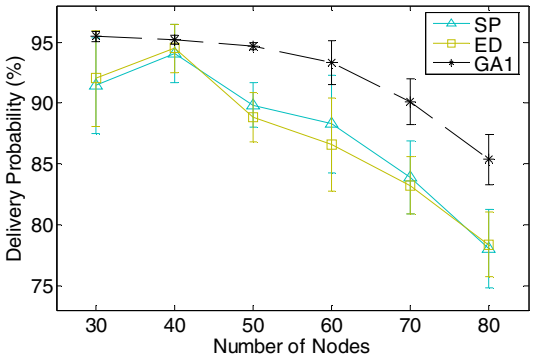
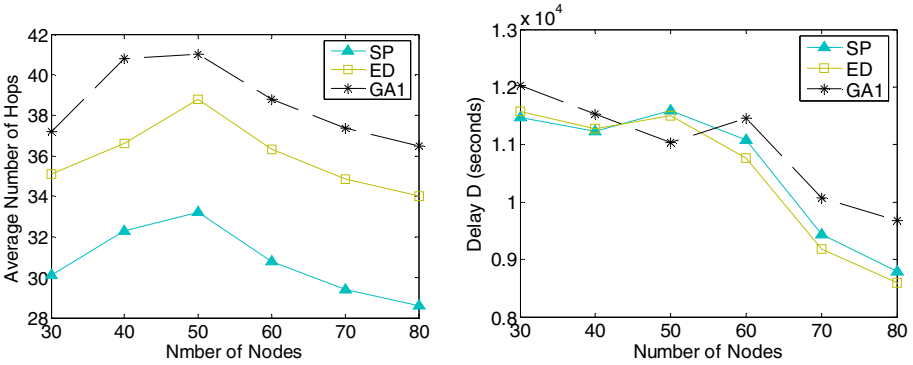


Fig. 6. Delivery probability for different numbers of nodes

SP algorithm computes paths with the lesser number of hops, as showed in Fig. 7 (a). GA1 uses more hops than the SP and ED algorithms. This is because GA1 searches alternative routes to distribute the traffic in the network better. Fig. 7 (b) shows that the delay *D* doesn't present large variations. Thus, GA1 obtains the *DP* presented in Fig. 6 without impair the delay *D*.



(a) Average number of hops (b) Delay *D*

Fig. 7. Delay *D* and average number of hops for different number of nodes

This way, in all simulated scenarios GA1 outperforms the remainder algorithms. This is because GA1 searches for the best combination of routes taking into account the storage constraints of mobile devices and uses the concept of subpopulation.

Furthermore, the number of generations has been limited for the algorithm to take an acceptable time to find the routes (less than the leaving time of the mobile devices).

5 Conclusion and Future Works

Future DTN nodes will likely have to support a number of different routing strategies and protocols. In this paper, we analyzed a GA-based solution to the anycast routing in deterministic DTNs. The proposed algorithm searches the combination of routes above a minimum delivery probability and having the least delay. Simulation results showed that the proposed GA-based algorithm outperforms the remainder studied algorithms and the concept of subpopulation is efficient. Thus, we can conclude based on the simulated scenarios, that the proposed GA-based algorithm has good scalability when we vary the area in which the nodes are distributed, edge densities and number of the nodes in the network. As future works, an extension of this research is to set up a testbed using real mobile devices and new and/or different strategies can emerge and be investigated in order to improve the GA-based anycast routing algorithm.

References

1. Khabbaz, M.J., Assi, C.M., Fawaz, W.F.: Disruption-Tolerant Networking: A Comprehensive Survey on Recent Developments and Persisting Challenges. IEEE Communications Surveys & Tutorials 14, 607–640 (2012)
2. Zhang, Z.: Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges. Communications Surveys & Tutorials 8, 24–37 (2006)
3. Gong, Y., et al.: Anycast routing in delay tolerant networks. In: Global Telecommunications Conference (GLOBECOM), pp. 1–5 (2006)
4. Jain, S., Fall, D., Patra, R.: Routing in a delay tolerant network. In: Special Interest Group on Data Communications – SIGCOMM, pp. 145–158 (2004)
5. Yussof, S.: Performance analysis of genetic algorithm (GA)-based multi-constrained path routing algorithm. International Journal of the Physical Sciences 33, 7524–7539 (2011)
6. Da Silva, E.R., Guardieiro, P.R.: Anycast routing in delay tolerant networks using genetic algorithms for route decision. In: 11th International Conference on Computer and Information Technology, pp. 65–71 (2008)
7. Da Silva, E.R., Guardieiro, P.R.: An efficient genetic algorithm for anycast routing in delay/disruption tolerant networks. IEEE Communications Letters 14, 315–317 (2010)
8. Goldberg, D.E.: Genetic algorithms in search, optimization and machine learning. Addison-Wesley, Massachusets (1989)
9. Randaccio, L.S., Atzori, L.: Group multicast routing problem: a genetic algorithms based approach. Computer Networks 51, 3989–4004 (2007)
10. Lo, C.C., Chang, W.H.: A multiobjective hybrid genetic algorithm for the capacitated multipoint network design problem. IEEE Trans. on System, Man, and Cybernetics. 30, 461–470 (2000)
11. Ferreira, A.: Building a reference combinatorial model for MANETs. IEEE Network 18, 24–29 (2004)
12. Waxman, B.M.: Routing of multipoint connection. IEEE Journal on Selected Areas in Communications 6, 1617–1622 (1988)

Reactive Power Optimization Using Hybrid Cultural Algorithm

Bidishna Bhattacharya[1,*], Kamal Krishna Mandal[2], and Niladri Chakraborty[2]

[1] Techno India, Saltlake, Electrical Engineering Dept., India
bidishna_inf@yahoo.co.in
[2] Jadavpur University, Department of Power Engineering, India

Abstract. Optimal reactive power dispatch is an important task to achieve secure and economic operation of power systems. A well-organized allocation of reactive power in an electric network can minimize the system losses. This paper presents a Cultural Algorithm (CA) with a single point crossover to minimize the real power loss subjected to limits on generator real and reactive power outputs. In this hybrid approach, CA is used to give a good direction to the optimal global region, and a domain knowledge is used as a fine tuning to determine the optimal solution at the final for better convergence. The solution can be achieved by varying the bus voltages, the on-load tap changer positions of transformers and by switching of shunt capacitors. The performance of the proposed method is demonstrated on IEEE 14-bus system to find the optimal reactive power control variables subjected to various equality and inequality constraints. It is found that the results obtained by the proposed method are comparable in terms real power losses.

1 Introduction

Reactive power optimization is one of the important optimization problems in optimal operation of power system. To decrease the active power losses along the transmission lines under various operating conditions a number of control variables such as switching reactive power sources, charging generator voltages and adjusting transformer tap settings has been selected[1]. Generally the typical optimal power flow (OPF) [2] solution is used for adjusting the appropriate control variables, so that the real power loss of the network is optimized with respect to the power system constraints which must be maintained within the allowable limits.

Several conventional optimization techniques [2-7] including the Gradient method, Non-linear Programming (NLP), Quadratic Programming (QP), Linear programming (LP) and Interior point method have been proposed earlier to solve the reactive power optimization problems. But due to the non-linearity and non-convex nature of the reactive power optimization problem, these techniques have complications in handling problems with the discrete variables. Recently, to avoid the deficiencies of the conventional methods, several heuristic optimization algorithms have been proposed like Expert System (ES),Genetic Algorithm (GA), Tabu Search (TS), Simulated

* Corresponding author.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 106–115, 2012.

Annealing (SA), Evolution Strategy (ES), Particle Swarm Optimization (PSO), differential evolution (DE) , evolutionary programming (EP) bacterial foraging optimization (BFO) etc in reactive power optimization [8 -18].

In this paper, an alternative approach to cultural algorithm has been proposed to solve the reactive power optimization problem. Cultural algorithm (CA) was proposed by Reynolds to model the social evolution [19], [20]. Cultural algorithm is basically a global optimization technique which consists of two evolutionary spaces; an evolutionary population space whose experiences are integrated into a Belief space which influences the search process to converge the problem in a direct way. Cultural algorithms have been successfully applied to global optimization of constrained functions, scheduling and real problems. Cultural Algorithm has been implemented for a few problem of Power System field such as substation planning [21], hydrothermal scheduling [22]. Economic load dispatch [23].

Here we integrate the evolutionary programming with single point crossover to cultural algorithm to solve the desired optimization problem involving several constraints. The framework of embedding EP into CA was developed by Chung and Reynolds [24] to investigate the influence of global knowledge on the justification of optimization problem. In this paper we worked on cultural algorithm having EP based population space which is dynamically controlled by the feasible region based records to direct the solutions towards the most promising region to solve constrained optimization problem efficiently.

2 Problem Formulation

System losses are most important issues for controlling and smooth operation of power system. Active or real power loss is a serious economic loss among these losses, so it is needed to minimize the active power loss. So the objective function of the reactive power optimization is as,

$$Minimize P_L = \sum_{k=nl} P_{loss} = \sum_{k=nl} g_k \left(V_i^2 + V_j^2 - 2V_i V_j Cos\theta_{ij} \right) \tag{1}$$

Where nl is the number of transmission lines, g_k is the conductance of the line connecting i and j bus, V_i is the voltage value of i th bus, V_j is the voltage value of jth bus and Θ_{ij} is phase angle of the voltage value between i and j bus.

The power loss is non-linear function of bus voltages, which are functions of control variables like generator bus voltage, tap settings of transformer and shunt capacitors. This minimization problem is subject to the limits on various control variables as inequality constraints and power flow constraints as the equality constraints).

Equality constraints:
There must be equilibrium between the produced active and reactive power and demanded power.

$$P_i - V_i \sum_{j=1}^{nb} V_j (G_{ij} Cos\theta_{ij} + B_{ij} Sin\theta_{ij}) = 0, i = 1,2,.....nb-1 \tag{2}$$

$$Q_i - V_i \sum_{j=1}^{nb} V_j (G_{ij} Cos\theta_{ij} - B_{ij} Sin\theta_{ij}) = 0, i = 1,2,.....nb-1 \quad (3)$$

Inequality constraints:
(i) Generator voltage constraints:

$$V_{G_i}^{\min} \le V_{G_i} \le V_{G_i}^{\max}, i = 1,2,.....ng \quad (4)$$

(ii) Generator reactive power capability limit:

$$Q_{G_i}^{\min} \le Q_{G_i} \le Q_{G_i}^{\max}, i = 1,2.....ng \quad (5)$$

(iii) Load voltage constraints:

$$V_{l_i}^{\min} \le V_{l_i} \le V_{l_i}^{\max}, i = 1,2,.....npq \quad (6)$$

(iv) Reactive power generation limit of capacitor banks:

$$Q_{C_i}^{\min} \le Q_{C_i} \le Q_{C_i}^{\max}, i = 1,2.....nc \quad (7)$$

(v) Transformer tap setting limit:

$$T_{K_i}^{\min} \le T_{K_i} \le T_{K_i}^{\max}, i = 1,2.....nt \quad (8)$$

(vi) Transmission line flow limit:

$$S_l \le S_l^{\max}, l = 1,2.....nl \quad (9)$$

Where nb, ng, npq, nc and nt are the number of total system bus, number of generator bus, number of load bus , number of capacitor bank and number of transformer, respectively. The equilibrium condition of equality constraints of equation (2) and (3) can be achieved by running the load flow program. And the inequality constraints of control variables are self-restricted by the optimization algorithm.

3 Cultural Algorithm (CA)

Reynolds first proposed Cultural Algorithm (CA) as a vehicle for modeling social evolution and learning the behavioral traits [20]. It is a high level searching technique. This evolutionary technique follows the cultural evolution of the society. Every society consisting of several classes of people has some rules and regulations which are obeyed by them and their offspring. A particular class, whom we called the elite class, is selected on the basis of their knowledge & wealth. This elite class people define and regulate the norms. The knowledge and concept of those people becomes the governing factor of the society. In this way culture or knowledge progresses from generations to generations making the new generation more up-to-date and fit for the survival. CA is nothing but the mathematical implementation of this learning procedure. The knowledge acquired by individuals through generations is stored to guide the behavior of the individuals. This acquired knowledge is stored in the search space called belief space in CA during the evolution of the population. Interaction between the two basic components i.e., population space and belief space make cultural

algorithm as a dual inheritance system. Population space is that where the information about individuals is stored and the belief space is where the culture knowledge is formed and maintained during the evolution of the population.

Belief space is basically a set of promising variable ranges that provide standards and guidelines within which individual adjustments can be made leading the individuals to go to the good range of solution. An acceptance function *accept ()* and updating function *update ()* play very vital role in belief space. After evolution of population space with a performance function *obj (), accept ()* will determine which individuals are kept aside for Belief space. Experiences of those elite individuals will update the knowledge of the Belief space via *update ().*These updated knowledge are used to influence the evolution of the population.

4 Overview of the Proposed Approach

The basic idea of using CA with evolutionary programming with crossover is to influence the mutation operator so that the current knowledge stored in the search space can be properly exploited. Crossover is a mixing operator that combines genetic material from selected parents to improve the search process. Cultural Algorithm belongs to the class of evolutionary algorithms which offers a unique strategy for optimization. The strategy used here is described by the following steps.

4.1 Initialization

The algorithm starts by creating a population vector P of size N_p composed of individuals that evolve over G generations. Each individual X_i is a vector that contains as many elements as the problem decision variable. The population size N_p is an algorithm control parameter selected by the user. Thus,

$$P^{(G)} = \left[X_i^{(G)}, \ldots\ldots\ldots\ldots X_{N_P}^{(G)}\right]$$

$$X_i^{(G)} = \left[X_{1,i}^{(G)}, \ldots\ldots\ldots\ldots X_{D,i}^{(G)}\right] \quad i = 1, \ldots\ldots\ldots N_P$$

The initial population is chosen randomly in order to cover the entire searching region uniformly. A uniform probability distribution for all random variables is assumed in the following as

$$X_{j,i}^{(0)} = X_j^{\min} + \sigma_j (X_j^{\max} - X_j^{\min})$$

Where $i = 1, \ldots\ldots\ldots N_p$ and $j = 1, \ldots\ldots\ldots\ldots D$

Here D is the number of decision or control variables, X_j^{min} and X_j^{max} are the lower and upper limits of the j th decision variable and $\sigma_j \in [0,1]$ is a uniformly distributed random number generated anew for each value of j. $X_{j,i}^{(0)}$ is the j th parameter of the i th individual of the initial population.

4.2 Initialization of Belief Space

In this paper two types of knowledge are used, one is situational knowledge and another is normative knowledge. Situational knowledge*(S)* is the set of best individuals and normative belief *(N)* is a set of interval information for each domain variable. For domain variable *j, N[j]* is represented as *<I,L,U>*. *I* denote the closed interval, that is a continuous set of real numbers, *x,* and is represented as:

$$I = [l, u] = \{x \mid l \le x \le u$$

Usually, *l* (lower bound) and *u* (upper bound) are initialized as the given domain values. *L* represents the performance score of the individual for the lower bound and *U* represents the performance score of the individual for the upper bound.

4.3 Evaluation of Objective Function

The objective function for each individual in the initial population is evaluated using load flow calculation. The situational knowledge *S* is updated by the current best solution as per the following rule:

$$S = \begin{cases} x_{best}; f(x_{best}) \le f(s) \\ s; otherwise \end{cases}$$

4.4 Modification of Belief Space

The parameter values for the current selected individuals by the acceptance function are used to determine the current satisfactory range of the normative knowledge. To update the normative knowledge minimum (X_i) and maximum (X_k) values for parameter j between the accepted individuals in the current generation are selected. Then the updated interval of normative knowledge is as follows,

$$l_j^{t+1} = \begin{cases} X_{i,j}^t, \text{ if } X_{i,j} \le l_j^t \text{ or } f(X_{i,j}^t) < L_j^t \\ l_j^t \quad otherwise \end{cases}$$

$$L_j^{t+1} = \begin{cases} f\left(X_{i,j}^t\right) \text{ if } X_{i,j} \le l_j^t \text{ or } f(X_{i,j}^t) < L_j^t \\ L_j^t \quad otherwise \end{cases}$$

and,

$$u_j^{t+1} = \begin{cases} X_{k,j}^t, \text{ if } X_{k,j} \le u_j^t \text{ or } f(X_{k,j}^t) < U_j^t \\ u_j^t \quad otherwise \end{cases}$$

$$U_j^{t+1} = \begin{cases} f\left(X_{k,j}^t\right) \text{ if } X_{k,j} \le u_j^t \text{ or } f(X_{k,j}^t) < U_j^t \\ U_j^t \quad otherwise \end{cases}$$

Where, l_j^t represents lower bound for parameter j at generation t and L_j^t denotes the performance score for it and u_j^t represents upper bound for parameter j at generation t and U_j^t denotes the performance score for it.

4.5 Generation of New Offspring

The influence function is liable for choosing the individuals of population space within the updated interval stored in belief space. The current individual of *n* numbers of candidate for parameter j can be selected by the formula given,

$$X_{i+n,j} = \begin{cases} X_{n,j} + |(u_j - l_j) * N_{n,j}(0,1)| & \text{if } X_{n,j} < l_j \\ X_{n,j} - |(u_j - l_j) * N_{n,j}(0,1)| & \text{if } X_{n,j} < u_j \end{cases}$$

u_j and l_j represent the upper value and lower value of parameter j of current elite in the belief space.

4.6 Mutation Operation

Gaussian mutation is used for the parameter j an offspring vector $x_{i,j}^{'}$ is created from each parent by adding to each component of $x_{i,j}$, a Gaussian random variable with a zero mean and a standard deviation proportional to the scaled cost values of the parent trial solution, i.e.,

$$x_{i,j}^{'} = x_{i,j} + N(0, \sigma_i^2) \text{ for } i = 1,2.....n$$

Where $N(0, \sigma_i^2)$ represents a Gaussian random variable with mean 0 and standard deviation σ_i^2 .

4.7 Crossover

The crossover operator is mainly responsible for the global search. The operator basically combines substructures of two parents to produce new structures. Crossover can occur at a single position (single crossover), or at number of different positions (multiple crossover). In this work single point crossovers is employed in which one crossover site is chosen and offspring are created by swapping the information between the chosen crossover sites.

4.8 Selection Operation

Selection is the operation through which better offspring are generated. In this algorithm, we perform tournament selection. After performing mutation, we will have a

population of size 2*p* (*p* parents generate *p* children). Tournament is performed considering the entire population .Tournaments consists of *c* confrontations per individual, with the *c* opponents randomly chosen from the entire population. When the tournaments finish, the *p* individuals with the largest number of victories are selected to form the following generation.

The optimization process is repeated for several generations. The iterative process of updating of belief space, mutation, crossover and selection on the population will continue until a user-specified stopping criterion, normally, the maximum number of generations allowed, is met.

5 Simulation Results

In order to validate the proposed hybrid approach, it is tested with the standard IEEE 14-Bus test system having non-linear characteristics. The algorithm has been written in MATLAB and run a 3.0 MHZ, 1GB RAM PC. The test system consists of five generator buses in which bus 1 is the slack bus, 2, 3, 6 and 8 are PV buses and the rest are PQ buses. The system also has 20 branches in which 3 branches are tap changing transformers branches. In addition, buses 9 and 14 have been selected as shunt VAR compensation buses. In IEEE 14 bus system, totally 9 control variables are taken for reactive power dispatch. The initial transmission line loss is 0.1349 p.u. Table 1 gives the parameter values for simulation which are selected by trial and error method.

Table 1. Simulation parameters used for proposed algorithm

Parameters	IEEE 14-bus
Population size	100
Maximum number of generation	200
Mutation factor	0.6

Table 2. Variable settings of the test system

Variables	Min [p.u]	Max [p.u]
V_G	0.95	1.10
V_{PQ}	0.95	1.05
T_C	0.90	1.10
C	0.00	0.18

The maximum and minimum limits of different variables are given in table 2. The optimal values of the control variables and power loss obtained are presented in Table 3. It is seen from Table 3 that, from the base case value of 0.1349p.u , the active power loss is reduced to 0.1328 p.u. which is also smaller than the result obtained by sequential quadratic program (SQP) [16] for the same IEEE 14-bus system. To illustrate the efficacy of the algorithm, the optimal power loss is plotted against the number of generations in Figure 1. The convergence characteristic proves the effectiveness of the proposed method for the reactive power optimization as the search process of the proposed method converges rapidly towards the optimal solution.

Table 3. Simulation Result for IEEE 14-bus System

Variable		Base Case	SQP	Hybrid CA
Generator Voltages	V_2	1.0450	1.0450	1.0433
	V_3	1.0100	1.0149	1.0164
	V_6	1.0700	1.0828	1.0825
	V_8	1.0900	1.1000	1.0973
Transformer Taps	T_{C4-7}	0.9467	1.0300	0.9257
	T_{C4-9}	0.9524	1.1000	0.9244
	T_{C5-6}	0.9091	1.0600	1.0049
Shunt Compensation	Q_{C9}	0.1800	0.1800	0.0600
	Q_{C14}	0.1800	0.0600	0.0600
Power Loss	P_{loss}	0.1349	0.1330	0.1328

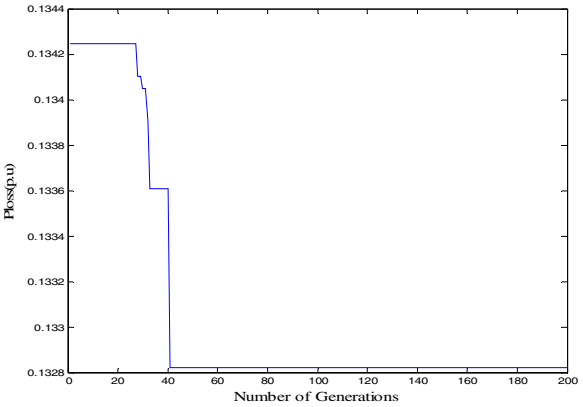


Fig. 1. Convergence Characteristics for IEEE 14-bus System

6 Conclusions

CA is an efficient heuristic algorithm in dealing with complex optimization problems. Besides its capability, the algorithm is also simple to be implemented. This study presents a hybrid cultural algorithm based method to solve the optimal reactive power dispatch problem. The problem is formulated as an optimization problem subject to several operational and electric constraints. The performance of proposed algorithm was examined and compared with other heuristic algorithms. The IEEE 14-bus system is used as the test case. The simulation results revealed that the proposed approach is able to remove the voltage limit violations and give higher power loss reduction as compared with SQP. The convergence characteristics show that the proposed method obviously has the better convergence speed.

References

1. Bansal, R., Bhatti, T., Kothari, D.: Artificial intelligence techniques for reactive power/voltage control in power systems: A review. International Journal of Power and Energy Systems 23, 81–89 (2003)
2. Dommel, H.W., Tinney, W.F.: Optimal Power Flow Solutions. IEEE Trans. Power App. Syst. PAS-87(10), 1866–1876 (1968)
3. Lee, K.Y., Park, Y.M., Ortiz, J.L.: A United Approach to Optimal Real and Reactive Power Dispatch. IEEE Trans. on Power Apparatus and Systems PAS-104(5), 1147–1153 (1985)
4. Granville, S.: Optimal Reactive Power Dispatch Through Interior Point Methods. IEEE Trans. on Power Systems 9(1), 98–105 (1994)
5. Momoh, J.A., El-Hawary, M.E., Adapa, R.: A review of selected optimal power flow literature to 1993 part I & II. IEEE Trans. Power Syst. 14(1), 96–111 (1999)
6. Ramos, J.L.M., Exposito, A.G., Quintana, V.H.: Transmission power loss reduction by interior-point methods: implementation issues and practical experience. IEE Proc. Gener. Trans. Distrib. 152(1), 90–98 (2005)
7. Quintana, V.H., Santos-Nieto, M.: Reactive power-dispatch by successive quadratic programming. IEEE Trans. Energy Convers. 4(3), 425–435 (1989)
8. Bakare, G.A., Venayagamoorthy, G.K., Aliyu, U.O.: Reactive Power and Voltage Control of the Nigerian Grid System Using Micro-Genetic Algorithm. In: Proceedings of IEEE Power Engineering Society General Meeting (2005)
9. El-Sayeed, M.A.H.: Ruled Based Approach for Real Time Reactive Power Control in Interconnected Power System. Expert System with Applications 14, 335–360 (1998)
10. Miranda, V., Srinivasan, D., Proenca, L.: Evolutionary Computation in Power Systems. Electrical Power and Energy Systems 20(2), 89–98 (1998)
11. Yoshida, H., Kawata, K., Fukuyama, Y., Nakanishi, Y.: A Particle Swarm Optimization for Reactive Power and Voltage Control Considering Voltage Stability Assessment. IEEE Transactions on Power Systems 15(4), 1232–1241 (2000)
12. Krost, G., Bakare, G.A.: A Genetic Algorithm Based Approach for Improvement in Voltage Profile and Real Power Loss Minimization. In: Proceedings of the IEEE Power Tech 1999 Conference, Budapest, Hungary, August 29-September 2, Section 24, paper BP99-293-23 (1999)
13. Bakare, G.: Removal of Overload and Voltage Problems using Genetic Algorithm and Expert Systems, Dissertation Gerhard Mercator University, Duisburg, Germany (2001)
14. del Valle, Y., Venayagamoorthy, G.K., Mohagheghi, S., Hernandez, J.C., Harley, R.G.: Particle Swarm Optimization: Basic Concepts, Variants and Applications in Power Systems. IEEE Transactions on Evolutionary Computation (in press)
15. Liang, C.H., Chung, C.Y., Wong, K.P., Duan, X.Z., Tse, C.T.: Study of differential evolution for optimal reactive power flow. IEE Proc. Gener. Trans. Distrib. 1(2), 253–260 (2007)
16. Varadarajan, M., Swarup, K.S.: Network loss minimization with voltage security using differential evolution. Elec. Power Syst. Research 78, 815–823 (2008)
17. Wu, Q.H., Ma, J.T.: Power system optimal reactive power dispatch using evolutionary programming. IEEE Trans. Power Syst. 10(3), 1243–1249 (1995)
18. Tripathy, M., Mishra, S.: Bacteria Foraging-Based Solution to Optimize Both Real Power Loss and Voltage Stability Limit. IEEE Trans. Power Syst. 22(1), 240–248 (2007)

19. Reynolds, R.G.: An Adaptive Computer Model of the Evolution of Agriculture for Hunter-Gatherers in the Valley of Oaxaca Mexico. Ph.D. Dissertation, Department of Computer Science, University of Michigan (1979)
20. Reynolds, R.G.: An Introduction to Cultural Algorithms. In: Proceedings of the 3rd Annual Conference on Evolutionary Programming, pp. 131–139. World Scientific Publishing (1994)
21. Liu, J., Gao, H., Zhang, J., Dai, B.: Urban power network substation optimal planning based on Geographic Cultural Algorithm. In: The 8th Int. Conf. Power Engineering, pp. 500–504 (2007)
22. Yuan, X., Yuan, Y.: Application of Cultural Algorithm to generation scheduling of hydrothermal systems. Energy Conversion and Management 47(15-28), 2192–2201 (2006)
23. Seifi, A.R.: A new Hybrid Optimization method for optimum distribution capacitor planning. Modern Applied Science, CCSE 3(4), 196–202 (2009)
24. Chung, C.-J., Reynolds, R.G.: CAEP: An Evolution-based Tool for real valued Function Optimization using Cultural algorithms. International Journal on Artificial Intelligence Tools 7(3) (1998)

Techno-Economic Feasibility Analysis of Hybrid Renewable Energy System Using Improved Version of Particle Swarm Optimization

Bhimsen Tudu, Preetam Roy, Sajjan Kumar, Diptendu Pal, Kamal Krishna Mandal, and Niladri Chakraborty

Power Engineering Department, Jadavpur University, Kolkata 700098, India
{bhimsen_ju,kkm567}@yahoo.co.in,
{preetamroy33,sajjan.pradhan48,dptndpal0}@gmail.com,
chakraborty_niladri@hotmail.com

Abstract. The present paper presents an improved version of particle swarm optimization method for obtaining unit sizing and techno-economic feasibility analysis of off-grid hybrid energy system for Sundarban region, world's largest mangrove forest located partly in West Bengal, India considering the real load data and other meteorological parameters. Initially the hybrid energy system is designed keeping in mind the load pattern and the availability of the renewable sources of that specific location. The hybrid system is designed with the combination of different renewable energy sources like wind turbines, solar panels along with battery and diesel generator to meet the localized load demand in different hours. Net present cost (NPC) and cost of energy (COE) for power generation have been considered to obtain the optimal unit sizing of the system. Emission from the hybrid system is also considered and compared with a conventional energy system in terms of emission per unit of generation and it is seen that with the implementation of this hybrid system, 0.688Kg of CO_2 emission per unit generation of electricity can be reduced while meeting the local demand. It is also seen that the improved version of particle swarm optimization technique is quite capable of solving this complex non-linear optimization problem quite efficiently.

Keywords: Particle Swarm Optimization, Stochastic inertia weight (Sto-IW) PSO, Off-grid renewable energy system, Solar panels, Wind turbine, Diesel generator, Battery, Emission.

1 Introduction

It is found that almost thirty three percent of the world's populations do not have access to electricity [1-2]. These regions can be electrified either by grid extension or by constructing isolated standalone new energy systems. But sometimes, grid extension is not possible due to high cost associated and many other social and economic reasons. Rapid depletion of conventional energy resources like fossil fuel and adverse environmental effects of conventional power generation systems have created massive interest in renewable energy sources and leads to a new era towards building a

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 116–123, 2012.

sustainable energy economy. The use of wind-solar hybrid energy system is growing fast all over the world and expanding globally at a rate of 25–35% annually over the last decade [3, 4]. But the hybrid system of wind & solar is not so reliable because of the intermittent nature of the resources. So battery or diesel generator can be combined with the system for better reliability and flexibility of planning. A hybrid power system has the ability to provide 24hours grid quality electricity to the load. For better understanding of performance and proper optimization of this type of hybrid system, performance analysis with different approaches and different meta-heuristic techniques have been used extensively such as Genetic Algorithm [5-6], Particle Swarm Optimization [7-8], Differential Evolutionary Algorithm, Ant Colony, Tabu Search etc. [9-11]. Different types of hybrid systems were explored by many researchers. Gavanidou and Bakirtzis explored a trade-off between SPV array sizing and number of batteries within a system to maintain the system cost low on the one hand, and for the minimum Loss of Load Probability (LOLP) requirement on the other [12]. Kaushika *et al.* [13] employed interconnected SPV arrays instead of the traditional single aperture model to increase the amount of electricity produced. A study by Sinha and Kandpal revealed that stand-alone Renewable Energy Technologies (RETs) can be acted as a proper and essential alternative to grid extension [14].A case study on Dahran which had energy demand of 620,000 KWh/year was conducted by Shaadid and Elhadidy. They built a hybrid SPV-DG-battery system, with 175 KW DG and 80KW SPV, which led to a COE of 0.149 $/KWh [15].

2 Hybrid Test System Model

The hybrid test system is modeled with the solar photovoltaic, wind turbine, battery, diesel generator and inverter and simulated with the help of advanced meta-heuristic optimization method called particle swarm optimization technique (PSO) and feasibility analysis of the model system is being done on a test location in West Bengal, India. Initially, test location is investigated and social-economical condition and different renewable resources data has been studied. It is seen that the area is endowed with plenty of renewable sources. It is assumed that the project lifetime is 25 years and 10% increase in average load profile over the lifetime of the project has been considered for obtaining the sizing of the proposed hybrid energy system. The model system is shown in Fig. 1.

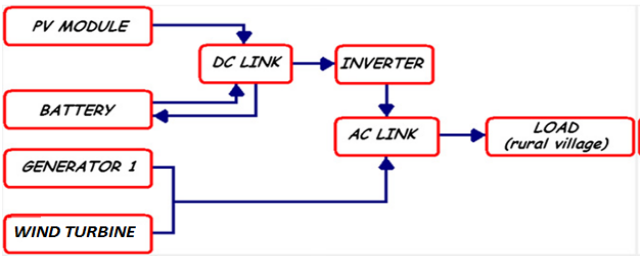


Fig. 1. Hybrid test system model

The solar insolation, wind speed and load calculated for the test location of Laxmijanardanpur village of South 24 Parganas District in West Bengal, India is shown in Fig. 2. The village is located in latitude of 21°46′N, longitude of 88°20′E and has total number of household of about 753.

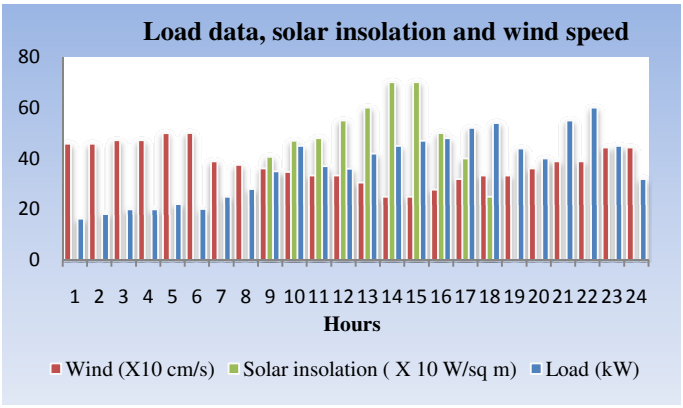


Fig. 2. Load data, solar insolation and wind speed of the site

The different parameters like capacity, lifetime and other associated parameters of the respective components considered for the system are given in Table 1. The different costs considered for the components are also given in Table 1. The all parameters data are taken from the standard manufacture make components. In this study, net present cost (NPC) is taken as the objective function for the hybrid system where NPC can be calculated as follows:

$$C_{NPC} = \frac{C_{ann,tot}}{CRF_{proj}} \tag{1}$$

Where CRF_{proj} is capital recovery factor of the project and $C_{ann,tot}$ for this hybrid system is expressed as [6]:

$$C_{ann,tot} = \sum_{s=1}^{N_s} C_{ann,solar,s} + \sum_{w=1}^{N_w} C_{ann,wind,w} + \sum_{c=1}^{N_c} C_{ann,inverter,c} + \sum_{b=1}^{N_b} C_{ann,battery,b} + \sum_{dg=1}^{N_{dg}} C_{ann,diesel\ generator,dg} \tag{2}$$

Where, N_s = number of installed solar modules, N_w = number of installed wind turbines, N_c = number of installed inverters, N_b = number of batter banks, N_{dg} = number of diesel generators.

Table 1. Different parameter values & costs (Rs./Yr) considered for the hybrid energy system

Component	Parameter	Value	Capital Cost(Rs.)	Replacement Cost(Rs.)	Operation Cost(Rs.)
Wind Turbine	Capacity	15kW	1125K	1012K	80K
	Rated speed	10.5m/s			
	Cut–in speed	3m/s			
	Furlingspeed	23m/s			
	Lifetime	20years			
	Anemometer height	4m			
	Alpha Co-efficient	0.14			
Solar PV Module	Rated power	0.15kW	25K	-	1K
	Slope	25degrees			
	Derating Factor	0.9			
	Nominal operating cell temperature	47^0C			
	Temperature co-efficient of panel	0.0043			
	PV cell temperature	25^0C			
	Lifetime	25 years			
Battery Bank	Rating	800Ah	50K	45K	5K
	Voltage	12V			
	Depth of discharge	0.5			
	Ambient temperature multiplier	1.0			
	Lifetime	10Years			
Diesel Generator	Capacity	20kW	190K	171K	19K
	Efficiency	90%			
	Lifetime	10Years			
Inverter	Capacity	9kVA	654K	-	6K
	Efficiency	90%			
	Lifetime	25Years			

The net present cost of the system is expressed as:

$$C_{NPC} = \frac{C_{ann,tot}}{CRF_{proj}} \tag{3}$$

The cost of energy of the system is given by [16] as:

$$COE = \frac{C_{ann,tot}}{Total\ \ load\ \ catered\ \ by\ the\ system\ \ over\ \ the\ \ year\ \ in\ kWh} \tag{4}$$

3 Particle Swarm Optimization technique (PSO)

The concept of Particle swarm optimization was originally proposed by Kenedy and Eberhart [17] based on natural behaviour of the particles or agents. Each position of the particle in search space represents the possible solution of the problem. Particles' positions are varied to give the better solutions and are governed by their own experiences and their neighbour particles' experiences. A particle's position is updated depending upon its current velocity, its own distance from its previous best position and distance from the best position occupied by the particle in the group. For each particle *i,* its position is updated in the following manner [Kennedy *et al*]:

$$X^{i}_{k+1} = X^{i}_{k} + V^{i}_{k+1} \tag{5}$$

Where V^{i}_{k+1} is pseudo velocity and can be calculated as follows:

$$V^{i}_{k+1} = w_k V^{i}_{k} + c_1 r_1 (P^{i}_{k} - X^{i}_{k}) + c_2 r_2 (P^{g}_{k} - X^{i}_{k}) \tag{6}$$

Here k represents a pseudo time increment, P^{i}_{k} represents the best ever position of particle i at time k and P^{g}_{k} represents the global best position in the group of swarm. r_1 & r_2 represent the uniform random values in the range [0 1], w represents the weight parameters and c_1 & c_2 are the cognitive and social parameters respectively.

There are different and modified versions of PSO like Canonical Particle Swarm Optimization, Self-Organizing Hierarchical Particle Swarm Optimization, Stochastic Inertia Weight Particle Swarm Optimization, Time-Varying Acceleration Coefficients Particle Swarm Optimization etc**.** In this study Stochastic Inertia Weight Particle Swarm Optimization (St-IW PSO) is tested and used on this hybrid system for optimization.

3.1 Stochastic Inertia Weight Particle Swarm Optimization (St-IW PSO)

This modified version of PSO was introduced by Russell Eberhart and Yuhui Shi [18]. Here inertia weight is randomly selected according to a uniform distribution in the range [0.5, 1.0] which is reflection of constriction factor proposed by *Maurice Clerc and James Kennedy*. In this version, both acceleration coefficients c_1 and c_2 are set to 1.494 [19].

4 Simulated Results

The system is optimized with the above mentioned technique and sizing is determined and is given in Table 2. The system is initially optimized with the solar PV, wind turbine and battery. The total output from PV and wind is shown in Fig. 3 and if surplus power is created in any hour, then it is used to charge battery and during power deficit, battery is discharged to meet the demand. The use of battery and diesel

generator is the trade-off between excess/deficit power and diesel fuel cost. The use of inverter is very much required for converting the DC output from battery and solar PV into AC which is fed to load.

Table 2. Optimal sizing obtained for the integrated hydro-wind-solar-fuel cell system

Component	Optimal Sizing (No. of Units)
Wind Turbine	5
Solar PV array	2
Inverter	1
Diesel Generator	2
Battery	10
Net Present cost (Rs.)	8879.7K
Cost of Energy (COE)(Rs./kWh)	8.927
Emission of carbon dioxide per year (Tons)	37.6607
Emission of carbon dioxide per unit generation (Kg/kWh)	0.116

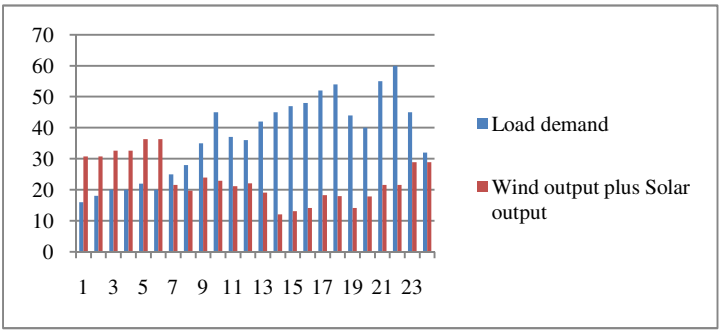


Fig. 3. Load versus total power output from PV and wind turbine

It is also being studied the emission from the system and it is seen that for per unit electricity generation, the emission is quite low compared to a 60MW Indian power plant [20] and comparison is shown in Table 3. Also the convergence for modified version of PSO is shown in Fig. 4.

Table 3. Comparison of emission with 60MW thermal plant

Objective Function	Results
Emission from 60MW thermal power plant (Kg/kWh)	0.804
Emission of carbon dioxide per unit generation (Kg/kWh)	0.116

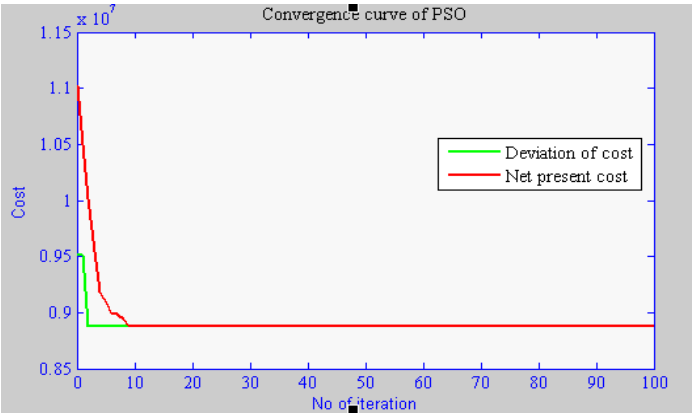


Fig. 4. The optimal Net Present Cost (NPC) in terms of the iterations for St-IW PSO

5 Conclusion

It is seen from this study that solar-wind-battery-DG hybrid system can serve the local load demand and with this, the COE for per unit generation is quite comparable with the conventional system and also the emission from the system is less. So, this hybrid system can be used for this type environment with less environmental effect and this version of PSO is quite efficient in solving this type of problem

References

1. Osama, O., Egon, O., Alaa, M., Danny, M.: An online control strategy for DC coupled hybrid power systems. In: IEEE Power Engineering Society General Meeting, Tampa, FL, pp. 1–8 (July 23, 2007)
2. Phuangpornpitak, N., Kumar, S.: PV hybrid systems for rural electrification in Thiland. Renewable and Sustainable Energy Reviews 11, 1530–1543 (2007)
3. Onar, O.C., Uzunoglu, M., Alam, M.S.: Dynamic modeling, design and simulation of a wind/fuel cell/ultra-capacitor-based hybrid power generation system. Journal of Power Sources 161, 707–722 (2006)
4. Setiawan, A.A., Zhao, Y., Susanto-Lee, R., Nayar, C.V.: Design, economic analysis and environmental considerations of mini-grid hybrid power system with reverse Osmosis desalination plant for remote areas. Renewable Energy 34, 374–383 (2009)
5. Koutroulis, E., Kolokotsa, D., Potirakis, A., Kalaitzakis, K.: Metho-dology for optimal sizing of stand-alone photovoltaic/wind-generator systems using genetic algorithms. Solar Energy 80, 1072–1088 (2006)
6. Dufo-Lopez, R., Bernal-Agustin, J.L.: Design and control strategies of PV-Diesel systems using genetic algorithms. Solar Energy 79, 33–46 (2005)
7. Hakimi, S.M., Moghaddas-Tafreshi, S.M.: Optimal sizing of a stand-alone hybrid power system via particle swarm optimization for Kahnouj area in south-east of Iran. Renewable Energy 34, 1855–1862 (2009)

8. Hakimi, S.M., Tafreshi, S.M., Kashefi, A.: Unit sizing of a stand-alone hybrid power system using particle swarm optimization (PSO). In: Proceedings of the IEEE International Conference on Automation and Logistics, Jinan, China, August 18-21 (2007)
9. Qi, Y., Jianhua, Z., Zifa, L., Shu, X., Weiguo, L.: A new methodology for optimizing the size of hybrid PV/wind system. In: ICSET 2008 (2008)
10. Xu, D., Kang, L., Cao, B.: Graph-Based Ant System for Optimal Sizing of Standalone Hybrid Wind/PV Power Systems. In: Huang, D.-S., Li, K., Irwin, G.W. (eds.) ICIC 2006. LNCS (LNAI), vol. 4114, pp. 1136–1146. Springer, Heidelberg (2006)
11. Katsigiannis, Y.A., Georgilakis, P.S.: Optimal sizing of small isolated hybrid power systems using tabu search. Journal of Optoelectronics and Advanced Materials 10(5), 1241–1245 (2008)
12. Gavanidou, E.S., Bakirtzis, A.G.: Design of a standalone system with renewable energy sources using trade off methods. IEEE Trans. on Energy Conversions, 42–48
13. Kaushika, N., Gautam, N.K., Kaushik, K.: Simulation model for sizing of standalone solar PV system with interconnected array. Solar Energy Materials and Solar Cells 85(4), 499–519 (2005)
14. Sinha, C.S., Kandpal, T.C.: Decentralized versus grid electricity for rural India – the economic factors. Energy Policy 19(5), 441–448 (1991)
15. Shaahid, S.M., Elhadidy, M.A.: Technical and economic assessment of grid independent hybrid photovoltaic-diesel-battery power systems for commercial loads in desert environments. Renewable and Sustainable Energy Reviews 11, 1794–1810 (2007)
16. Katsigiannis, Y.A., Georgilakis, P.S., Karapidakis, E.S.: Genetic Algorithm Solution to Optimal Sizing Problem of Small Autonomous Hybrid Power Systems. In: Konstantopoulos, S., Perantonis, S., Karkaletsis, V., Spyropoulos, C.D., Vouros, G. (eds.) SETN 2010. LNCS, vol. 6040, pp. 327–332. Springer, Heidelberg (2010)
17. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks IV, pp. 1942–1948 (1995)
18. Eberhart, R., Shi, Y.: Tracking and optimizing dynamic systems with particle swarms. In: Proceedings of the 2001 IEEE Congress on Evolutionary Computation, pp. 94–100. IEEE Press, Piscataway (2001)
19. Montes de Oca, M.A., Stützle, T., Birattari, M., Dorigo, M.: A Comparison of Particle Swarm Optimization Algorithms Based on Run-Length Distributions. In: Dorigo, M., Gambardella, L.M., Birattari, M., Martinoli, A., Poli, R., Stützle, T. (eds.) ANTS 2006. LNCS, vol. 4150, pp. 1–12. Springer, Heidelberg (2006)
20. Chakraborty, N., Mukherjee, I., Santra, A.K., Chowdhury, S., Chakraborty, S., Bhattacharya, S., Mitra, A.P., Sharma, C.: Measurement of CO_2, CO, SO_2 and NO emissions from coal-based thermal power plants in India. Atmospheric Environment 42, 1073–1082 (2008)

Software Effort Prediction Using Fuzzy Clustering and Functional Link Artificial Neural Networks

Tirimula Rao Benala[1], Rajib Mall[2], Satchidananda Dehuri[3], and V.L. Prasanthi[1]

[1] Department of Computer Science and Engineering
Anil Neerukonda Institute of Technology and Sciences
Sangivalasa-531162, Visakhapatnam, Andhra Pradesh, India
{b.tirimula,prasanthi.anitscse}@gmail.com
[2] Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
rajib@cse.iitkgp.ernet.in
[3]Department of System Engineering,
Ajou University, San 5, Woncheon-dong,
Yeongtong-gu, Suwon 443-749, South Korea
satchi@ajou.ac.kr

Abstract. We use the combined fuzzy C-Means (FCM) clustering algorithm and functional link artificial neural networks (FLANN) to achieve accurate software effort prediction. FLANN is a computationally efficient nonlinear network and is capable for complex nonlinear mapping between its input and output pattern space. The nonlinearity is introduced into the FLANN by passing the input pattern through a functional expansion unit. The proposed method uses three real time datasets. The Chebyshev polynomial has been used as choice of expansion to exhaustively study the performance. The simulation results show that it not only deals efficiently with noisy data but also proves to be a champion in producing promising results.

Keywords: Software cost estimation, Fuzzy C-Means, K-Means, and FLANN.

1 Introduction

Software cost estimation refers to prediction of effort, time, and staffing levels required to build a software system. Accurate software cost estimation is critical for the effective software project management. It significantly affects management activities such as resource allocation, project bidding, and planning. The importance of accurate estimation has led researchers to conduct extensive research on software cost estimation methods. In this paper, we investigate use of fuzzy C-means (FCM) algorithm and FLANN for software cost estimation. Clustering is a technique for discovering the hidden structure of data. It is a process of grouping the objects into clusters such that objects from the same cluster are similar and objects from different clusters is dissimilar. In practice, many datasets cannot be decomposed into disjoint partitions. In these situations, the use of clustering algorithms that are capable of

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 124–132, 2012.

dealing with such overlapping data clusters is recommended. Fuzzy clustering techniques addresses this issue effectively as they aim to find fuzzy clusters to which all the data points belong to some degree. Fuzzy C-means technique proposed by Bezdek [2] is one of the unsupervised ways of clustering the dataset.

The FLANN architecture for predicting software development effort is a single-layer feed forward neural network consisting of one input layer and an output layer. FLANN generates output (effort) by expanding the initial inputs (cost drivers) and then processing the final output layer. Each input neuron corresponds to a component of an input vector. The output layer consists of one output neuron that computes the software development effort as a linear weighted sum of the outputs of the input layer [15,16]. The large and non-normal data sets always lead FLANN methods to low prediction accuracy and high computational complexity. To alleviate these drawbacks our proposed technique has been formulated as follows.

i) Fix the number of clusters and initialize center locations of each cluster. Use iterative process described in the clustering algorithms to get quality clusters. The clustering algorithms used in this work are K-means and fuzzy C-means.
ii) Select a cluster based on the nearest-neighbor of an input sample to train the FLANN.
iii) Use trained FLANN for effort prediction.

Functional link neural networks and cost estimation fundamentals are briefly reviewed in Section 2. The proposed approach is described in Section 3. In Section 4, numerical examples from Cocomo81 (Coc81), Nasa93, Maxwell dataset is used to evaluate the performance. Section 5 concludes this paper.

2 Background

2.1 Software Cost Estimation

Software effort estimation (SEE) can be defined as the process of estimating the total effort necessary to complete a software project [7] . A typical example of SEE is to use attributes which characterize the software project to predict (or estimate) the cost, in terms of person-months, to conclude the software development, and consequently it is possible to predict the required time to deliver the product [1,11]. In order to effectively manage development, maintenance or customization of software, the project team has to estimate beforehand the software project development time and the project cost for the company. According to Oliveira [11], the main risk factors for software projects are the schedule and effort (cost) to finish it; the particularities of software project requisites, project team, and the employed technology make the process of cost estimation hard. Due to these and other characteristics of each project, it has been accepted that the accurate measurement of the cost and development time of software is only possible after the project is completed [3,11]. However, it is necessary to perform estimations before the project begins. There are wide number of techniques and methods which can be employed to estimate such variables. This work introduces a novel technique aimed to predict (estimate) the software development cost; our proposed technique is based on FCM algorithm combined with FLANN.

2.2 Architecture of FLANN

A FLANN can be used not only for functional approximation but also for decreasing the computational complexity. This method is mainly focused on functional approximation. In the aspect of learning, the FLANN network is much faster than other network. The primary reason for this is that the learning process in FLANN network has two stages and both stages can be made efficient by appropriate learning algorithms. The use of on FLANN to estimate software development effort requires the determination of its architecture parameters according to the characteristics of datasets [15,16]. The architecture of FLANN is shown in figure 1.

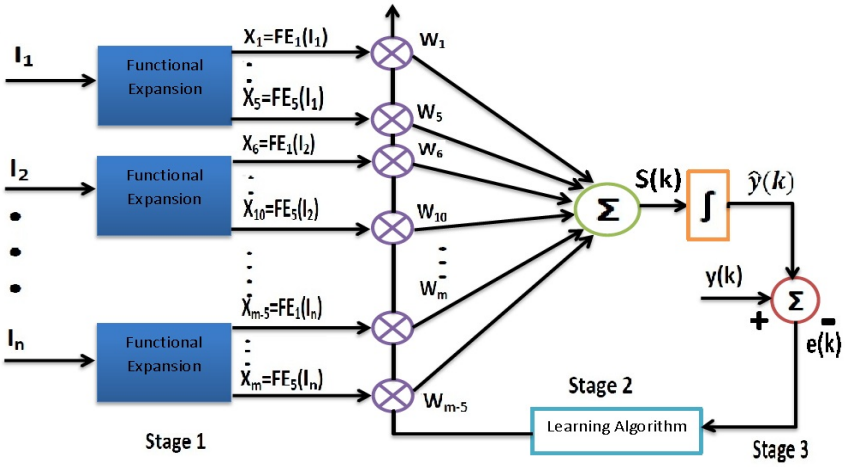


Fig. 1. FLANN Architecture

2.3 Clustering Techniques

The essence of all clustering techniques is: given n patterns, or data points, in a d-dimensional space, the clustering problem refers to partitioning the n patterns into c groups or clusters based on some similarity metrics. An optimal clustering is a partitioning that minimizes the intra-cluster distance and maximizes the inter-cluster distance. In our proposed work we have considered two most popular clustering techniques, namely, K-means and fuzzy C-means techniques.

2.3.1 K-means

K-means [10] clustering is a centre-based clustering method that is by far the most widely used partitional algorithm for data clustering as it is simple and very easy to implement. The algorithm starts with k initial seeds of clustering, one for each cluster. All the n objects are then compared with each seed by means of the Euclidean distance and assigned to the closest cluster seed. The procedure is then repeated over and over again. In each stage, the seed of each cluster is recalculated by using the average vector of the objects assigned to the cluster. The algorithm stops when the

changes in the cluster seeds from one stage to the next are close to zero or smaller than a pre-specified value. Every object is assigned to only one cluster.

2.3.2 Fuzzy C-Means

The fuzzy C-means (FCM) algorithm [2,4] is one of the most popular techniques used for clustering. The effectiveness of the clustering method relies on the distance measure. The conventional FCM method uses the Euclidean distance as the similarity criterion that measures the distance between each data point x_i and a cluster centroid v_c , i.e.$\|x_i - v_c\|^2$ with a weight w_{ic} which is inversely proportional to the distance. FCM partitions the data set X into C clusters by minimizing the errors in terms of the weighted distance of each data point x_i to all centroids of the C clusters. That is,

$$\text{Min } J_{FCM}=\sum_{c=1}^{C}\sum_{i=1}^{N} w_{ic}^{p} \quad \|x_i - v_c\|^2,$$

where $\sum_{c=1}^{C} w_{ic} = 1$, i=1, 2,…, N and p is the exponent.

By using Lagrange multipliers, we can solve for the weight w_{ic}. The weight w_{ic} and the centroid v_c can be updated by the expectation–maximization (E–M) algorithm:

E-Step:

$W_{ic}=1/\sum_{j=1}^{c}\left(\frac{d_{ic}^2}{d_{ij}^2}\right)^{1/(p-1)}$, i = 1,2,…,N and c=1,2,…,C ,

where

$$d_{ic}^2 = \|x_i - v_c\|^2$$

M-Step:

$v_c = \frac{\sum_{j=1}^{N} w_{jc}^{p} \cdot x_j}{\sum_{j=1}^{N} w_{jc}^{p}}$, c=1,2,…,C

The E–M algorithm recursively computes until a convergence condition is satisfied.

3 FCM-FLANN Algorithm

In this section, we first propose the methodology for software cost estimation, next algorithm and finally performance evaluation metrics.

3.1 Methodology

The two unsupervised clustering algorithms, namely K-Means and FCM, were applied on the patterns of the training set to partition the input space. The number of clusters is fixed beforehand and the center locations of each cluster are initialized. Each Cluster center is m-dimensional vector. Partitioning algorithms starts with an initial k partitions and then uses an iterative process to optimize the cluster quality. A test set pattern is assigned to the cluster to which the nearest (in terms of Euclidean distance) centroid belongs for the K-Means, and FCM algorithms, respectively. The framework is shown in Figure 2.

3.2 Algorithm

Step 1: Before applying any method, all training datasets are normalized in order to remove the scaling effects on different dimensions. By using min-max normalization, project attributes values are converted into the [0, 1] interval.
Step 2: Unsupervised clustering algorithms (K-means & FCM) are applied; The number of clusters is fixed at 3 for experimentation by trial and error method.
Step 3: Each test data (also known as input data point) is assigned to appropriate cluster using Euclidean distance as criterion ;thus selecting promising cluster for the given input data point
Step 4: The promising cluster is fed to the trained FLANN to obtain effort in Person-Months (PM).

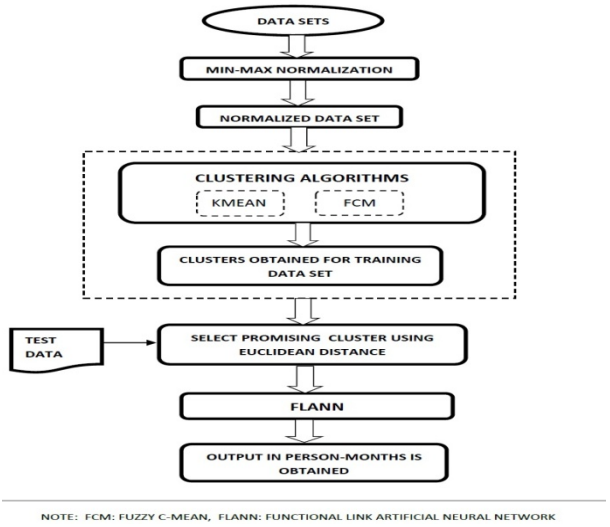


Fig. 2. Framework

3.3 Performance Evaluation Metrics

To measure the accuracies of the proposed methods, three performance metrics are considered: Mean Magnitude of Relative Error (MMRE), Median Magnitude of relative error (MdMRE), and PRED (0.25), because these measures are widely accepted in literature [5, 14].

4 Datasets, Experiments, and Results

In this section, three real world software engineering datasets, namely, Cocomo81 (Coc81), Nasa93, Maxwell [8] are utilized for empirical evaluation of our methods.

4.1 Dataset Preparation

Before the experiments, all types of features are normalized into the interval [0, 1] in order to eliminate the possibility of unequal influences. The three real datasets shown in Table 1 are randomly split into three nearly equal sized sub-sets for training and testing. The training set is treated as the targets for the optimization of feature weights and project subsets. The testing set is exclusively used to evaluate the optimized FLANN models. Table 1 provides an overview of the data sets, including number of features, size. The skewness, minimum, mean and maximum of effort and size in Klocs.

Table 1. Descriptive statistics for public datasets

Dataset	*Coc81*	*Nasa93*	*Maxwell*
Features	17	17	27
Size	63	93	62
Units	Months	Months	Hours
Minimum Effort	6	8	583
Median	98	252	5189.5
Mean	683	624	8223.2
Maximum Effort	11400	8211	63694
Skew	4.4	4.2	3.26

4.2 Cost Estimation Models

Out of the three functional expansions, namely, C-FLANN, P-FLANN and L-FLANN, C-FLANN based model is included in our experiments. Hereafter, C-FLANN will be annotated as FLANN. The proposed models using Fuzzy C-Means and K-means clustering will be hereafter known as FCM-FLANN and KM-FLANN. For a comprehensive evaluation of the proposed models, for comparison, other popular estimation models including Step wise regression (SWR) [12], Functional Link Artificial Neural Networks (FLANN) [15.16], and classification and regression trees (CART) [13], are also included in the experiments.

4.3 Experitmental Procedure

For the purpose of validation, we adopt three-fold cross validation [6,9] to evaluate accuracy of the methods. In this scheme all the three datasets are randomly divided into three nearly equal sized subsets. At each time only one of three subsets is used as the test sets which are exclusively used to evaluate the estimation performance, and other two subsets treated as Validation data set and training data set exclusively used to optimize the cost drivers. This process is repeated three times. Then the average

training error and testing error across all three trials are computed. The advantage of this scheme is that it does not matter how the data is split since each data point is assigned into a test set, a training set and a validation set respectively once. First, the performances of KMeans-FLANN (KM-FLANN) and FCM-FLANN are investigated. The best variants on training set are selected as the candidate for comparisons. Next, the optimizations of machines learning methods are conducted on the training dataset by searching through their parameter spaces. Thirdly, the training and testing results of the best variants of all estimation methods are summarized and compared. The experimental results and the analysis are presented in the next section.

4.4 Experimental Results

Tables 2 to 4 present a summary of all the methods applied on three real time dataset given in Table 1. The second annotated column in each table shows performance of various methods with respect to performance metrics MMRE. Similarly, the third column and fourth column of each table summarizes the results with respect to performance metrics MdMRE and PRED(0.25) respectively. With these values it can be interpreted that the testing results in the proposed methods outperform the testing results in traditional methods.

Our experiments suggest that hybrid combination of CM/FCM and FLANN improves the accuracy very efficiently when compared to SWR, FLANN and CART.

Table 2. Results on Coc81 Dataset

Methods	*MMRE*		*MdMRE*		*PRED(0.25)*	
	Training	**Testing**	**Training**	**Testing**	**Training**	**Testing**
FCM-FLANN	**0.35**	**0.30**	**0.43**	**0.42**	**0.51**	**0.54**
KM-FLANN	**0.41**	**0.41**	**0.52**	**0.46**	**0.39**	**0.38**
FLANN	**0.45**	**0.38**	**0.49**	**0.47**	**0.35**	**0.49**
SWR	**0.34**	**0.35**	**0.42**	**0.44**	**0.52**	**0.50**
CART	**1.28**	**1.12**	**0.62**	**0.58**	**0.17**	**0.19**

Table 3. Results on Nasa93 Dataset

Methods	*MMRE*		*MdMRE*		*PRED(0.25)*	
	Training	**Testing**	**Training**	**Testing**	**Training**	**Testing**
FCM-FLANN	**0.40**	**0.31**	**0.49**	**0.42**	**0.52**	**0.52**
KM-FLANN	**0.45**	**0.36**	**0.47**	**0.44**	**0.37**	**0.48**
FLANN	**0.42**	**0.49**	**0.46**	**0.48**	**0.38**	**0.48**
SWR	**0.39**	**0.34**	**0.47**	**0.49**	**0.44**	**0.44**
CART	**1.34**	**1.28**	**0.85**	**0.66**	**0.23**	**0.30**

Table 4. Results on Maxwell Dataset

Methods	*MMRE*		*MdMRE*		*PRED(0.25)*	
	Training	**Testing**	**Training**	**Testing**	**Training**	**Testing**
FCM-FLANN	**0.35**	**0.30**	**0.47**	**0.41**	**0.41**	**0.59**
KM-FLANN	**0.48**	**0.39**	**0.44**	**0.47**	**0.39**	**0.49**
FLANN	**0.48**	**0.42**	**0.39**	**0.40**	**0.45**	**0.28**
SWR	**0.42**	**0.42**	**0.47**	**0.39**	**0.39**	**0.45**
CART	**0.92**	**0.91**	**0.46**	**0.39**	**0.29**	**0.31**

5 Conclusion and Future Work

We have done our research in the direction of software cost estimation by hybrid system using FCM and FLANN. We extend connotations to our work with Artificial Bee Colony (ABC), Differential Evolution (DE), Artificial Immune System (AIS), Bacterial Foraging Optimization Algorithm (BFOA), Neuro Fuzzy, Neuro Genetic, Simulated Annealing, and Fuzzy Logic. We have evaluated the performance of FCM-FLANN. The experimental results show that our method gives improved performance as compared to conventional FLANN and outperforms the competitive techniques such as KM-FLANN, SWR and CART.

References

1. de Araújo, R.A., Oliveira, A.L.I., Soares, S.: A shift-invariant morphological system for software development cost estimation. Expert Systems with Applications 38, 4162–4168 (2011)
2. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithm. Plenum Press, New York (1981)
3. Braga, P.L., Oliveira, A.L.I., Ribeiro, G.H.T., Meira, S.R.L.: Software effort estimation using machine learning techniques with robust confidence intervals. In: IEEE International Conference on Tools with Artificial Intelligence (ICTAI) (2007)
4. Tsai, D.-M., Lin, C.-C.: Fuzzy C-means based clustering for linearly and nonlinearly separable data. Pattern Recognition 44, 1750–1760 (2011)
5. Foss, T., Stensrud, E., Kitchenham, B., Myrtveit, I.: A simulation study of the model evaluation criterion MMRE. IEEE Transactions on Software Engineering 29(11) (2003)
6. Huang, S.J., Chiu, N.H.: Optimization of analogy weights by genetic algorithm for software effort estimation. Information and Software Technology 48, 1034–1045 (2006)
7. Keung, J.W.: Theoretical Maximum Prediction Accuracy for Analogy-Based Software Cost Estimation. In: 15th Asia-Pacific Software Engineering Conference, pp. 495–502 (2008), http://ieeexplore.ieee.org/lpdocsepic03/wrapper.htm?arnumber=4724583
8. Menzies, T.: The PROMISE Repository of Software Engineering Databases. School of Information Technology and Engineering, University of Ottawa, Canada (2006), http://promise.site.uottawa.ca/SERepository

9. Mendes, E., Watson, I., Triggs, C., Mosley, N., Counsell, S.: A Comparative Study of Cost Estimation Models for Web Hypermedia Applications. Empirical Software Engineering 8, 163–196 (2003)
10. McQueen, J.B.: Some methods of classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
11. Oliveira, A.L.I.: Estimation of software project effort with support vector regression. Neurocomputing 69(13-15), 1749–1753 (2006)
12. Shepperd, M., Kadoda, G.: Comparing Software Prediction Techniques using Simulation. IEEE Transaction on Software Engineering 27(11), 1014–1022 (2001)
13. Stensrud, E.: Alternative Approaches to Software Prediction of ERP Projects. Information and Software Technology 43(7), 413–423 (2001)
14. Stensrud, E., Foss, T., Kitchenham, B.A., Myrtveit, I.: An empirical validation of the relationship between the magnitude of relative error and project size. In: Proceedings of the IEEE 8th Metrics Symposium, pp. 3–12 (2002)
15. Tirimula Rao, B., Sameet, B., Kiran Swathi, G., Vikram Gupta, K., Raviteja, C., Sumana, S.: A Novel Neural Network approach for Software Cost Estimation Using Functional Link Artificial Neural Networks. International Journal of Computer Science and Network Security (IJCSNS) 9(6), 126–131 (2009)
16. Tirimula Rao, B., Dehuri, S., Mall, R.: Functional Link Artificial Neural Networks for Software Cost Estimation. International Journal of Applied Evolutionary Computation (IJAEC) 3(2), 62–82 (2012)

Optimal Placement and Sizing of Distributed Generation in Radial Distribution System Using Differential Evolution Algorithm

Manas R. Nayak[1], Subrat K. Dash[2], and Pravat Kumar Rout[2]

[1] Deptt. of Electrical Engg , ITER, S'O'A University, Bhubaneswar, 751030, Odisha, India
[2] Deptt. of EEE, ITER , S'O'A University, Bhubaneswar, 751030, Odisha, India
{manasnk72,dashsubratkumar}@gmail.com,
pkrout_india@yahoo.com

Abstract. Integration of renewable energy based distributed generation (DG) units provides potential benefits to conventional distribution systems. The power injections from renewable DG units located close to the load centers provide an opportunity for system voltage support, reduction in energy losses and emissions, and reliability improvement. Therefore, the allocation of DG units should be carefully determined with the consideration of different planning incentives. Optimal placement and sizing of DG in distribution network is an optimization problem with continuous and discrete variables. This paper proposes a Differential Evolution Algorithm (DEA) for optimal placement and sizing of distributed generation (DG) in radial distribution system to minimize the total real power loss and improve the voltage profile within the frame work of system operation and security constraints. The proposed DE algorithm is also used to determine optimal sizes and locations of multi-DGs. The proposed method is tested on standard IEEE 69-bus test system and the results are presented and compared with different approaches available in the literature. The proposed method has outperformed than the other methods in terms of the quality of solution and computational efficiency.

1 Introduction

Electric utilities are now seeking upcoming new technologies to provide acceptable power quality and higher reliability to their customers in restructured environment. Non-conventional generation is growing more rapidly around the world, for its low size, low cost and less environmental impact with high potentiality. Technically, they are suitable for installation at low voltage distribution system, near loads centres . Due to limitation on fossil fuel resources, alternative solutions to traditional large power stations are under high priority in recent years to meet growing energy demand of the future.

Electric power systems have been originally designed based on the unidirectional power flow, but the concept of distributed generation (DG) has led to new considerations concerning the distribution networks [1]. The penetration of DG may impact the operation of a distribution network in both beneficial and detrimental

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 133–142, 2012.

ways. Some of the positive impacts of DG are: voltage support, power loss reduction, support of ancillary services and improved reliability, whereas negative ones are protection coordination, dynamic stability and islanding. In order to maximize benefits and minimize problems, technical constraints concerning the interconnection of DG units and their penetration levels are being adopted worldwide. Furthermore, the presence of DG in the deregulated market has raised new regulatory issues, concerning financial incentives, cost allocation methods, generation management techniques, etc.

There are a number of approaches proposed for placement and sizing of DG units. Khan and Choudhry [2] developed an algorithm based on analytical approach to improve the voltage profile and to reduce the power loss under randomly distributed load conditions with low power factor for single DG as well as multi DG systems. Hung et al. [3] used an improved analytical method for identification of the best location and optimal power factor for placing multiple DGs to achieve loss reduction in large-scale primary distribution networks. Ziari et al. [4] proposed a discrete particle swarm optimization and genetic algorithm (GA) based approach for optimal planning of DG in distribution network to minimize loss and improve reliability. Kamel and Karmanshahi [5] proposed an algorithm for optimal sizing and siting of DGs at any bus in the distribution system to minimize losses and found that the total losses in the distribution network would reduce by nearly 85%, if DGs were located at the optimal locations with optimal sizes. Singh et al. [6] discussed a multi-objective performance index based technique using GA for optimal location and sizing of DG resources in distribution systems.

Differential Evolutionary (DE) Algorithm [7] is a novel evolution algorithm as it employs real-coded variables and typically relies on mutation as the search operator. There are also a number of significant advantages when using DE which were ability to find the true global minimum regardless of the initial parameter values, Parallel processing nature and fast convergence; Capable of providing multiple solutions in a single run. DEA method is proposed to determine the optimal location and sizes of multi-DGs to minimize the total real power loss and improve the voltage profile within the frame work of system operation and security constraints in radial distribution system.

The organization of this paper is as follows. Section 2 addresses the Load flow Study in radial distribution network. The problem formulation is in Section 3. The DE algorithm is presented in Section 4. A DEA computation procedure for the optimal placement and sizing of distributed generation problem is given in Section 5. Simulation results on the test systems are illustrated in Section 6 and the conclusion in Section 7.

2 Load Flow Study

Conventional NR and Gauss Seidel (GS) methods may become inefficient in the analysis of distribution systems, due to the special features of distribution networks, i.e. radial structure, high R/X ratio and unbalanced loads, etc. These features make the distribution systems power flow computation different and somewhat difficult to analyze as compared to the transmission systems. Various methods are available to

carry out the analysis of balanced and unbalanced radial distribution systems and can be divided into two categories. The first type of methods is utilized by proper modification of existing methods such as NR and GS methods. On the other hand, the second group of methods is based on backward and forward sweep processes using Kirchhoff's laws. Due to its low memory requirements, computational efficiency and robust convergence characteristic, backward and forward sweep based algorithms have gained the most popularity for distribution systems load flow analysis. In the present study, network topology based backward and forward sweep method [8] is used to find out the load flow solution for balanced radial distribution system. Detailed description of load flow solution is omitted due to the lack of space.

3 Problem Formulation

The problem is to determine placement and size of the DGs which minimizes the distribution power losses, improve the voltage profile and maximize the voltage stability in a given radial distribution system while satisfying all constraints for a fixed number of DGs and specific total capacity of the DGs.

Fig. 1 shows a branch of radial system. In radial distribution system [9] each receiving node is fed by only one sending node, From Fig.1

$$I_{ni} = \frac{V_{mi} - V_{ni}}{R_{ni} + j X_{ni}} \tag{1}$$

$$P_{ni} - jQ_{ni} = V_{ni} * I_{ni} \tag{2}$$

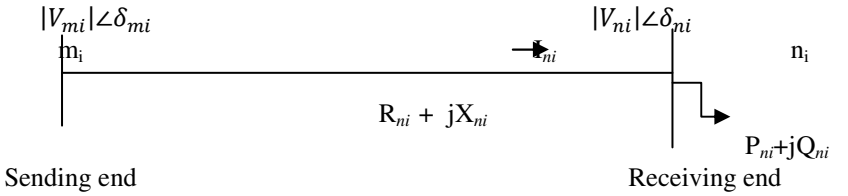


Fig. 1. Electrical equivalent diagram of a radial distribution system

3.1 Objective Function

In this paper, two objective functions are considered separately as single objective.

3.1.1 Minimization of the Real Power Loss

The real power losses in the system is given by (3)

$$f_1 = P_{RPL} \tag{3}$$

P_{RPL} is the real power losses of n_n – bus distribution system, and is expressed in components as :

$$P_{RPL} = \frac{R_{(ni)} * [P_{ni}^2 + Q_{ni}^2]}{|V_{ni}|^2} \tag{4}$$

3.1.2 Improvement of the Voltage Deviation (Voltage Profile)

The objective function to improve voltage profile is given by (5):

$$f_2 = \sum_{ni=1}^{n_n} (V_{ni} - V_{rated})^2 \tag{5}$$

3.2 System Constraints

The constraints are listed as follows:

a) Distribution line absolute power limits:

$$\left|P_{ni,Line}\right| \leq \left|P_{ni,Line}^{max}\right| \tag{6}$$

$P_{ni,Line}$ and $P_{ni,Line}^{max}$ are the absolute power and its corresponding maximum allowable value flowing over the distribution line between the nodes.

b) Bus voltage limit:

Bus voltage amplitudes are limited as

$$V_{ni}^{min} < V_{ni} < V_{ni}^{max} \tag{7}$$

Where V_{ni}^{min} and V_{ni}^{max} are the minimum and maximum values of bus voltage amplitudes, respectively.

c) Radial structure of the network:

$$M = N_{bus} - N_f \tag{8}$$

Where M is the number of branches, N_{bus} is the number of nodes and N_f is the number of sources.

d) Power limits of DG:

$$P_{gni}^{min} \leq P_{gni} \leq P_{gni}^{max}, \qquad Q_{gni}^{min} \leq Q_{gni} \leq Q_{gni}^{max} \tag{9}$$

Where P_{gni} and Q_{gni} are the injected active and reactive power of DG components at the nith bus.

e) Subject to power balance constraints:

$$\sum_{ni}^{nn} P_{gni} = \sum_{ni}^{nn} P_{Dni} + P_L \tag{10}$$

Where nn is total number of sections, P_L is the real power loss in the system, P_{gni} is the real power generation at bus ni, P_{Dni} is the power demand at bus ni.

4 Differential Evolution Algorithm (DEA)

DEA is an evolutionary computational algorithm that was originally introduced by Storn Price(1995). The DEA optimisation process is carried out by applying the following three basic genetic operation; mutation, recombination (as known as crossover) and selection. After the population is initialised, the operators of mutation,

crossover and selection create the population of the next generation $pop^{(G+1).}$ by using the current $pop^{(G)}$.

DEA Optimization Process

Initialisation: In the first step of the DEA optimization process, the population of candidate solutions must be initialised. The initial population of candidate is generated within its corresponding feasible limits as follows:

$$X_j^{(G=0)} = X_{min} + (X_{max} - X_{min}).* \text{rand}(j, D), \quad J=1, 2,....N_p \tag{11}$$

$$X = [X_1\ X_2\ X_3\ \ldots\ldots X_j\ \ldots\ \ldots X_{N_p}], \quad j = 1,2,3\ \ldots\ldots\ N_p \tag{12}$$

Where $N_p = \text{number of population}$ and D= number of decision variables.

In this study, the active power injected by DG is considered as decision variable.

Mutation: The mutation operator generates mutant vectors (v_i^G) by perturbing a randomly selected vector (X_{r1}) with the difference of two other randomly selected vectors (X_{r2} and X_{r3}). DEA has several strategies to generate mutant vectors but in the study, the simplest and most popular DE method is used.

$$V_i^G = X_{r1}^G + F\left(X_{r2}^G - X_{r3}^G\right), i = 1,2, \ldots . N_p \tag{13}$$

Vector indices r_1,r_2 and r_3 are randomly chosen where r_1 r_2 and r_3 belong to $\{1....N_p\}$ and $r_1 \neq r_2 \neq r_3 \neq i$. F is a user defined constant known as the scaling mutation factor which is typically chosen from within the range [0,1].

Crossover: Crossover operation helps to increase the diversity among the mutant parameter vectors and aids the algorithm to escape from local optima. At the generation G, the crossover operation creates trial vectors (U_i) by mixing the parameters of the mutant vectors (V_i) with the target vectors (X_i) according to a selected probability distribution:

$$U_i^G = U_{j,1}^G = \begin{cases} v_{j,i}^{(G)}, & \text{if } \text{rand}_j \leq CR \\ X_{j,i}^{(G)}, & \text{otherwise} \end{cases} \tag{14}$$

The crossover constant CR is a user-defined value (known as the crossover probability) which is usually selected from within the range [0,1].

Selection: The selection operator chooses the vectors that are going to compose the population in the next generation. This operator compares the fitness of the trial vector and corresponding target vector and selects the one that provides the best so0lution and advance it into the next generation according to following equation:

$$X_i^{(G+1)} = \begin{cases} U_i^{(G)}, & \text{if } f\left(U_i^{(G)}\right) \leq f\left(X_i^{(G)}\right) \\ X_i^{(G)}, & \text{otherwise} \end{cases} \tag{15}$$

The overall optimisation process is stopped whenever maximum number of generations is reached or other predetermined convergence criterion is satisfied.

5 Implementation of DEA to the Problem

In the problem the values of DG set are considered as decision variables. The step wise implementation of DEA to the problem is as follows:

1. Read the feeder data: maximum number of DG sets N_{DG}^{max}.
2. Set the control parameters of DEA optimization process those are population size (N_p=40) number of decision variables (D), scaling mutation factor (F=0.7), Crossover probability (CR=0.8) and maximum number of generations (G^{max} =30).
3. Generate initial population for the chosen decision variable(s).
4. Loop
5. To place number of DG sets and their positions at which they have to be placed can be obtained as follows: let N_{bus} = number of busses ; $N_{DG}{}^{G}$= number of DG sets to be placed at Gth iteration; K= probability of getting 1's in a binary number where: K= $\frac{N_{DG}^{G}}{N_{bus}}$
   - Generate a binary number with size N_{bus} and probability K
   - The positions of non zeros in the number themselves represent the positions of the DG sets
6. Run the load flow
7. Calculate total active power loss, Voltage Deviation and Voltage magnitude using load flow equations
8. Apply convergence criterion as if:
   $0.95 \leq V_i \leq 1.05$
   Print the result and STOP else, GO to step 10
9. Increment generation counter by one i.e G=G+1
10. Increase number of DG sets as follows if:
    $N_{DG}{}^{G} < N_{DG}{}^{max}$
    Then: $N_{DG}{}^{G} = N_{DG}{}^{G} + 1$
11. Apply mutation, crossover, selection operations using eq 13-15, respectively
12. Modify the population
13. End loop.

6 Simulation Results

The test system for the case study is radial distribution system with IEEE 69 buses [11].The total loads for these test systems are 3.80 MW and 2.69 MVAR. The initial total real power loss and reactive power loss in the system are 224.96 KW and 114.15 KVAR respectively. The substation voltage is 12.66 KV and the base of power is 100MVA. The minimum and maximum voltages are set at 0.95 and 1.05 p.u. respectively. The load data and branch data for this system are given in [10] which

Table 1. Performance analysis of the 69-bus system after DG installation

Method	Objective Function	
	f_1	f_2
DE	0.0121	0.0021
GA/PSO[16]	0.0811	0.0031
GA[16]	0.089	0.0012
PSO[16]	0.0832	0.0049
Without DG	0.2217	0.2197

have seven lateral lines..The maximum number of DG is 3. The rating real power of DG and the power factor are 1200KW and 1.

To validate the effectiveness of the proposed algorithm; two different cases have been taken into consideration as follows:

(1) Minimization of real power losses
(2) Improvement of the voltage profile

Table -1 gives the comparison of different algorithm [11] performance analysis for the 69-bus system after DG installation and before DG installation.

Case-1: Minimization of the real power loss

In this case the convergence characteristic of objective function (Real Power loss) is shown in Fig. 2. Real Power loss & Bus Voltage before and after DG installation are shown in the Fig 3- 4. Table -2 (case-1) gives the optimal size, location and total reduction of Real Power loss after DG installation.

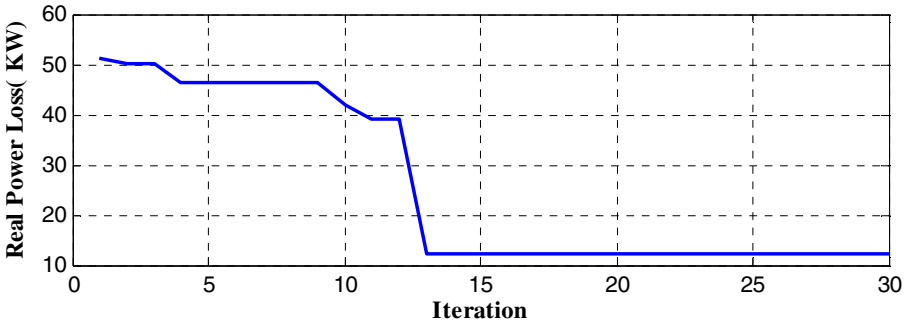


Fig. 2. Real power Loss Variation of Case 1

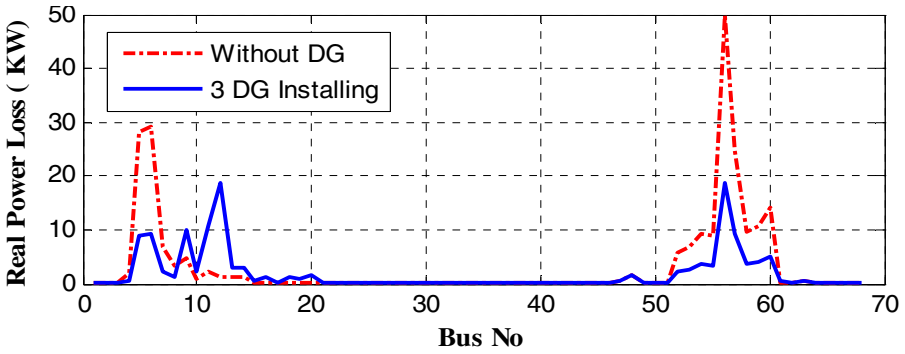


Fig. 3. Real Power loss before & after DG Installation (case 1)

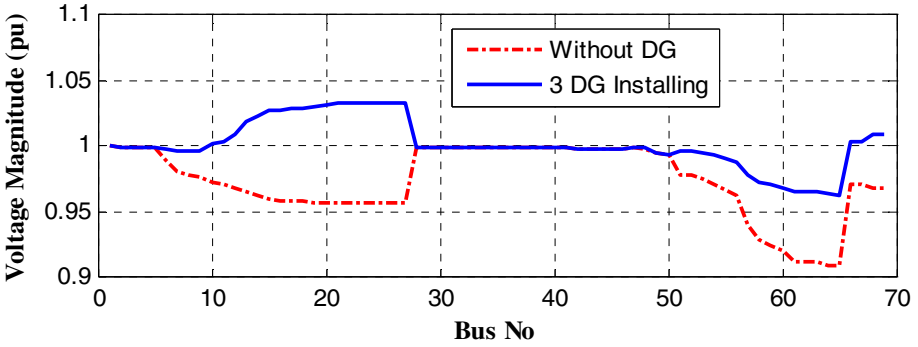


Fig. 4. Bus Voltage before & after DG Installation (case 1)

Table 2. Case - 1 & Case - 2

Case-1		Case-2	
System Type	69-Bus	System Type	69-Bus
Optimal Location of DG	22, 13, 62	Optimal Location of DG	58, 62, 24
Optimal DG size in MW	1.0112, 1.0865, 1.0623	Optimal DG size in MW	1.1665, 0.9819, 0.6645
Total Real power Loss in MW (Without DG)	0.2217	Voltage Deviation in pu (Without DG)	0.2197
Total Real power Loss in MW (With DG)	0.0121	Voltage Deviation in pu (With DG)	0.0021
Real Power Loss reduction in %	94.5	Improvement of Voltage deviation in pu	0.9904

Case-2: Improvement of the Voltage Profile

In this case the convergence characteristic of objective function (Voltage Deviation) is shown in Fig.5. Voltage Deviation & Bus Voltage before and after DG installation are shown in the Fig 6-7. Table -2 (case-2) gives the optimal size, location and Improvement of the voltage after DG installation.

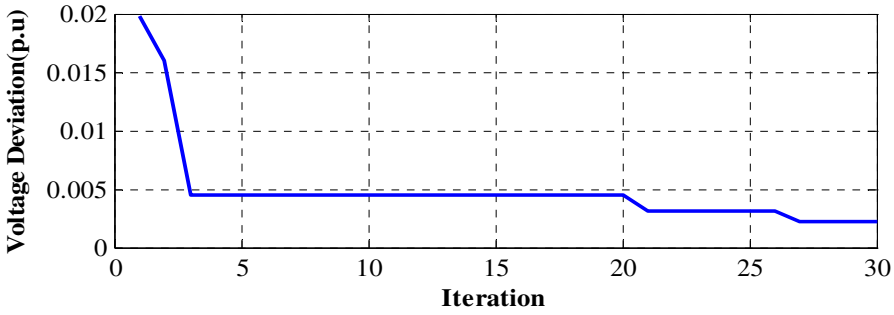


Fig. 5. Voltage deviation (p.u) Variation of case 2

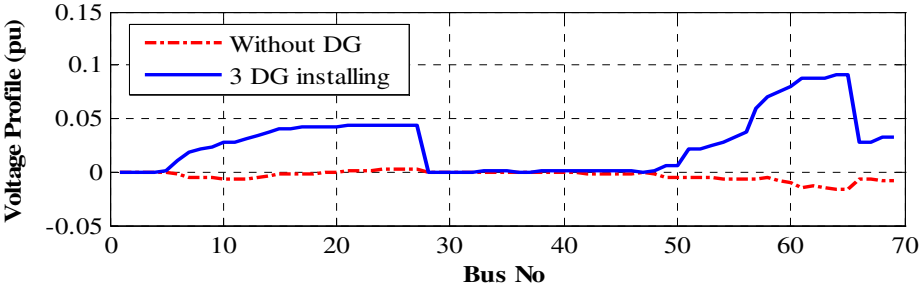


Fig. 6. Voltage deviation (p.u) before & after DG installation (case 2)

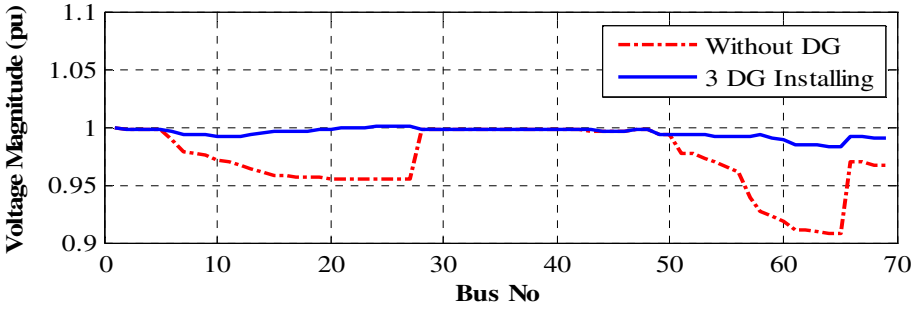


Fig. 7. Bus Voltage (p.u) before & after DG Installation (case 2)

7 Conclusions

The validity of the proposed method is proved from the comparison of the results of the proposed method with other existing methods. The results proved that the DEA is simple in nature than GA and PSO so it takes less computation time and accurate in determining the sizes and locations of DGs. By installing DGs at all the potential locations, the total power loss of the system has been reduced drastically. The voltage profile of the system is also improved. Inclusion of the real time constrains such as time varying loads and different types of DG units and discrete DG unit sizes into the proposed algorithm is the future scope of this work.

References

1. Ackermann, T., Andersson, G., Soder, L.: Distributed generation: a definition. Electrical Power System Research 57(3), 195–204 (2001)
2. Khan, H., Choudhry, M.A.: Implementation of distributed generation algorithm for performance enhancement of distribution feeder under extreme load growth. International Journal of Electrical Power and Energy Systems 32(9), 985–997 (2010)
3. Hung, D.Q., Mithulanathan, N., Bansal, R.C.: Multiple distributed generators placement in primary distribution networks for loss reduction. IEEE Transactions on Industrial Electronics (in press)
4. Ziari, I., Ledwich, G., Ghosh, A., Cornforth, D., Wishart, M.: Optimal allocation and sizing of DGs in distribution networks. In: IEEE Power and Energy Society General Meeting, pp.1–8 (2010)
5. Kamel, R.M., Karmanshahi, B.: Optimal size and location of DGs for minimizing power losses in a primary distribution network. Transaction on Computer Science and Electrical and Electronics Engineering 16(2), 137–144 (2009)
6. Singh, D., Singh, D., Verma, K.S.: Multi-objective optimization for DG planning with load models. IEEE Transactions on Power Systems 24(1), 427–436 (2009)
7. Storn, R., Price, K.: Differential Evolution – "A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces", Technical Report TR-95-012, ICSI (1995)
8. Haque, M.H.: Efficient load flow method for distribution systems with radial or mesh configuration. IEE Proc. on Generation, Transmission and Distribution 143(1), 33–38 (1996)
9. Charkravorty, M., Das, D.: Voltage stability analysis of radial distribution networks. Int. J. Electrical Power Energy Syst. 23(2), 129–135 (2001)
10. Vovos, P.N., Bialek, J.W.: Direct incorporation of fault level constraints in optimal power flow as a tool for network capacity analysis. IEEE Trans. Power Syst. 20(4), 2125–2134 (2005)
11. Moradi, M.H., Abedini, M.: A combination of genetic algorithm and particle swarm optimization for optimal DG location and sizing in distribution systems. Electrical Power and Energy Systems 34, 66–74 (2012)

Clustering Algorithm Recommendation: A Meta-learning Approach

Daniel G. Ferrari and Leandro Nunes de Castro

Natural Computing Laboratory (LCoN)
Mackenzie Presbyterian University,
São Paulo, Brazil
ferrari.dg@gmail.com,
lnunes@mackenzie.br

Abstract. Meta-learning is a technique that aims at understanding what types of algorithms solve what kinds of problems. Clustering, by contrast, divides a dataset into groups based on the objects' similarities without the need of previous knowledge about the objects' labels. The present paper proposes the use of meta-learning to recommend clustering algorithms based on the feature extraction of unlabelled objects. The features of the clustering problems will be evaluated along with the ranking of different algorithms so that the meta-learning system can recommend accurately the best algorithms for a new problem.

Keywords: clustering, algorithm recommendation, ranking, meta-learning.

1 Introduction

There is currently a huge amount of information represented and stored as data to posterior analysis [1]. Researchers began to dedicate themselves to the development of methods to extract knowledge from data; the process of applying these methods is known as *data mining* [2]. Nowadays, data mining tools are characterized by a variety of algorithms able to solve each one of the many data mining tasks. However, this process suffers from the lack of guidelines to select the best algorithm to solve a given data mining problem [3].

The meta-learning field has as objective to find which problem features contribute to a better or worse performance of an algorithm [4], and, from this, recommend the most appropriate algorithm for solving a given problem [3]. To reach this objective, meta-learning builds two key sets: (1) *Meta-attributes*: the set of features that is common to several instances of a class of problems, such as the number of objects and the number of binary attributes, among others; (2) *Ranking*: the set with rank positions, based on a performance evaluation, of several algorithms applied to the same problems. From these two sets it is created a model to recommend the ranking of the algorithms when applied to other problems, not used for training, based on the meta-attributes proposed.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 143–150, 2012.

The connection between data mining and meta-learning has been widely investigated for classification tasks [3,5]. However, few studies are available in the literature for clustering tasks [6,7]. There is no study, for instance, about which will be the best feature set for unsupervised learning problems, like clustering [3].

The experiments performed in this work aims at investigating the recommendation of algorithms for clustering problems based on meta-attributes described in the literature for classification problems. Despite that, the features to be selected here will not require the data labels, thus making our methodology generic for clustering tasks.

This work is organized as follows. Section 2 presents a brief theoretical background about meta-learning. Section 3 explains the methodology used in the experiments and presents the results. The paper is concluded in Section 4 with a discussion about the results and feasibility of our proposal.

2 Meta-learning

Meta-learning is learning about learning, i.e., one must learn about the behavior of machine learning algorithms in order to find out the best algorithm for each application [8]. In 1994, the EU ESPRIT project *StatLog* [5] extended this concept with the objective of relating the performance of the algorithm with the features of objects for classification problems.

Meta-learning is tightly connected with the process of extracting and exploiting the *meta-knowledge*, which can assume different forms and be defined as any kind of knowledge that can be extracted from the learning process of an algorithm while being applied to a problem [4].

The meta-knowledge, also known as *meta-data*, can be composed of the *meta-attributes* and the *ranking* [3]. The meta-attributes are the features extracted from the problems. For example, to make the characterization of classification problems, the *StatLog* project proposed a set of sixteen meta-attributes based on simple measures, statistics and information theory. The rankings are positions occupied by the algorithms when their performance is evaluated, through a measure, on the same problems [9], i.e., with the performance values, the best algorithm occupies the first position; the second best occupies the second position, and so on.

The *meta-algorithm* is responsible for learning the relationship between the meta-attributes and the ranking and making this knowledge available with the objective of estimating the algorithms' rank. Classical machine learning algorithms are often used as meta-algorithms, such as decision trees, neural networks, instance-based learning, among others.

A conceptual model for a meta-learning system (Fig. 1) can be constructed using three main components [3,4]: (1) *Feature Extraction Module*: extract the features (meta-attributes) responsible for characterizing the problems and distinguishing them from one another; (2) *Algorithm Evaluation Module*: generate the ranking of the algorithms on the problems using a predefined evaluation measure; and (3) *Application Module*: has a meta-algorithm that is responsible for estimating the algorithms' rank.

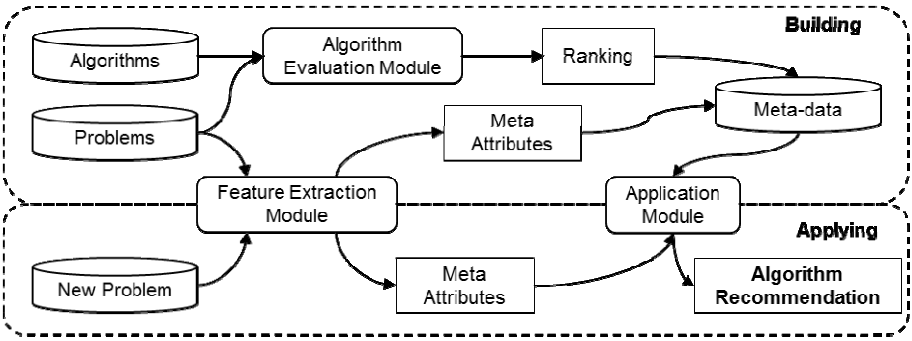


Fig. 1. Conceptual model for a meta-learning system

3 Experiments

The objective of this work is to investigate the feasibility of applying meta-learning techniques in clustering problems characterized by meta-attributes obtained from unlabelled data. Experiments will be run to predict the ranking of the clustering algorithms using datasets from the literature and data characterizations defined for classification problems.

3.1 Datasets

To compose the set of problems used for evaluation, 30 datasets from the UCI Machine Learning Repository [10] were selected. The objects with missing values were removed. The following datasets were selected: Haberman's Survival; Balloons; Balance; Iris; Mammographic; Car Evaluation; Ecoli; Yeast; Tic-Tac-Toe; Glass Identification; Wine; Australian Credit; Zoo; Image Segmentation; Mushroom; Breast Cancer (Diag); Chess (King-Rook vs. King-Pawn); Landsat Satellite; SPECTF Heart; Ionosphere; Lung Cancer; Spambase; Sonar, Mines vs. Rocks; Synthetic Control Chart Time Series; Optical Recognition of Handwritten Digits; Robot Execution Failures LP5; Libras Movement; Hill-Valley; German Credit; Musk (Version1).

3.2 Meta-attributes

To date there is no solid work investigating which meta-attributes should be used for characterizing the data in clustering tasks [3]. As clustering algorithms usually do not work with prior information on the labels of objects, in this work the meta-attributes to be extracted will not consider the labels of objects available for the datasets.

The meta-attributes were chosen based on the StatLog and METAL (www.metal-kdd.org) projects and on the works [11,12], being normalized in the range [0, 1] for the experiments. A sample of the meta-attributes is illustrated in Table 1. The selected meta-attributes are: (1) Log_2 of the number of objects; (2) Log_2 of the number of attributes; (3) Proportion of binary attributes; (4) Proportion of discrete attributes;

(5) Proportion of continuous attributes; (6) Mean absolute correlation between continuous attributes; (7) Mean skewness of continuous attributes; (8) Mean kurtosis of continuous attributes; (9) Mean absolute concentration between discrete attributes; and (10) Mean entropy of discrete attributes.

Table 1. Meta-attribute values for some datasets

Dataset/Value	1	2	3	4	5	6	7	8	9	10
Iris	7.23	2.00	0.00	0.00	1.00	0.59	0.07	2.23	0.00	0.00
Car Eval.	10.75	2.58	0.00	1.00	0.00	0.00	0.00	0.00	0.00	5.38
Ecoli	8.39	2.81	0.00	0.00	1.00	0.18	3.59	54.19	0.00	0.00
Yeast	10.54	3.00	0.00	0.00	1.00	0.09	2.91	31.56	0.00	0.00
Wine	7.48	3.70	0.00	0.00	1.00	0.30	0.35	2.97	0.00	0.00
Tic-Tac-Toe	9.90	3.17	0.00	1.00	0.00	0.00	0.00	0.00	0.01	3.91

3.3 Clustering Algorithms

This work evaluates the following clustering algorithms [1]: K-Means (KM); Single Linkage (SL); Complete Linkage (CL); Medium Linkage (ML); and the Self-Organizing Feature Map (SOM). For the K-means algorithm the value of k was chosen as the number of groups in the dataset and the centroids were initialized using randomly chosen samples. In the hierarchical algorithms (SL, CL and ML) the dendrogram was cut in the point where the number of clusters is the same as the number of groups in the dataset. The SOM used here is bi-dimensional with the number of neurons equals to the number of groups in the data set. All algorithms used the *Euclidean* distance as the dissimilarity metric.

The clustering solution will be evaluated using the metric *FBCubed* (*FBC*) [13], that can be calculated as follows:

$$FBC = \left(0{,}5 \left(\frac{1}{\frac{1}{n}\sum_{i=1}^{n} \frac{CL(i)}{Cluster(i)}} \right) + 0{,}5 \left(\frac{1}{\frac{1}{n}\sum_{i=1}^{n} \frac{CL(i)}{Label(i)}} \right) \right)^{-1} \quad (1)$$

where n is the number of objects in the dataset; $CL(i)$ is the number of objects in the same group as object i that have the same label as object i; $Cluster(i)$ is the number of objects in the same group as object i; $Label(i)$ is the number of objects that have the same label as object i.

The algorithms were executed 30 times for each dataset and the performance was obtained by the *FBC* resulting value. The ranking was set up based on the best performance of each algorithm for each dataset, and used as predictive table values.

Table 2 shows the best *FBC* result value for all clustering algorithms for some datasets, and Table 3 shows the predictive table for the same datasets. It can be seen that the clustering algorithm with the highest *FBC* value occupies the first position in the ranking for a dataset, and the second highest occupies the second position, and so on.

Table 2. Best *FBC* resulting values for the clustering solution

Dataset	KM	SL	CL	ML	SOM
Iris	0.75109	0.79579	0.75899	0.79484	0.75075
Car Eval.	0.42647	0.59709	0.36724	0.41249	0.38790
Haberman	0,70548	0,75917	0,72637	0,75512	0,55101
Ecoli	0.66051	0.46530	0.78537	0.68680	0.58789
Yeast	0.37361	0.38382	0.35382	0.38464	0.37607
Wine	0.93653	0.51704	0.74443	0.50954	0.94668
Tic-Tac-Toe	0.53386	0.70747	0.55915	0.61800	0.53404

Table 3. Predictive table built with ranking values

Dataset	KM	SL	CL	ML	SOM
Iris	5	1	3	2	4
Car Eval.	3	1	4	2	5
Haberman	4	1	3	2	5
Ecoli	3	5	1	2	4
Yeast	4	2	5	1	3
Wine	2	4	3	5	1
Tic-Tac-Toe	5	1	3	2	4

3.4 Meta-algorithms

The meta-algorithm is responsible for learning the relationship between the meta-attributes (Table 1) and the ranking (Table 3). In this work it will be used four machine learning algorithms with distinct learning mechanisms to play the role of meta-algorithm: the *K-Nearest Neighbors* (KNN); the *Multi-Layer Perceptron* (MLP) neural network; the *Decision Tree* (CART); and the *Naïve Bayes* (NB).

To evaluate the prediction quality it will be used the *Spearman Rank Correlation* (*SRC*) [14], which measures the correlation between paired and ordered values. The *SRC* gives results in the interval [–1, +1], where +1 means that both rankings are equal, and –1 means that they are inverted.

$$SRC = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n^3 - n} \quad (2)$$

where n is the rank size; and d_i is the difference between the real and predicted ranking value on the i-th position.

To compare the results between the meta-algorithms, for the KNN and MLP algorithms a parametric evaluation was performed to choose the best parameter value.

In the KNN algorithm the initial parameter k determines the numbers of neighbors that will be considered to calculate the output. An experiment was performed varying k in the range [1,...,6], and it was chosen k=2 to be compared with the other meta-algorithms.

In the MLP algorithm the variation was made in the number of neurons in the hidden layer. An experiment was performed varying he number of neurons in the range [1,…,10], and it was chosen 6 as the number of hidden neurons.

3.5 Results

It was used a 10-fold cross-validation with 30 executions to get the *SRC* value between the predicted and the real ranking for each meta-algorithm. Table 4 shows the mean and standard deviation for the *SRC* values for the final ranking recommendation for all meta-algorithms. Fig. 2 illustrates the variation of the *SRC* values for all meta-algorithms.

Table 4. Spearman Rank Correlation for Ranking Recommendation

Meta-Algorithm	SRC
NB	0.116 ± 0.075
CART	0.017 ± 0.095
KNN	0.212 ± 0.061
MLP	0.033 ± 0.098

A Lilliefors Test [15] was made with the results of the meta-algorithms to determine the normality of the data. The Lilliefors tests the null hypothesis (H_0) that the sample comes from a normally distributed population, without specifying the expected value and variance of the distribution. With 0.01 of significance level, all the results considered that they came from a normal distribution.

To verify the difference between the mean values of the results, a *t*-test was performed. The *t*-test verifies the null hypothesis (H_0) that the samples come from normal distributions with equal means. The result obtained by the KNN meta-algorithm was considered different with 0.01 of significance level. As the KNN obtained the higher mean, it can be considered the best meta-algorithm for the meta-learning system.

The *SRC* has its significance test tabulated with the null hypothesis (H_0) that there is no correlation between the ranks ($SRC = 0$), i.e., the hypothesis verifies if the correlation is equal to zero, statistically. The alternative hypothesis (H_1) will be that there is a positive correlation ($SRC > 0$), or agreement, between the rankings (one-tailed test). Analyzing the KNN meta-algorithm *SRC* between the real and recommended rankings; there is 0.05 significance level for the best *SRC* (*SRC*= 0.327), and there is a 0.25 of significance level for the mean *SRC* (*SRC*= 0.212).

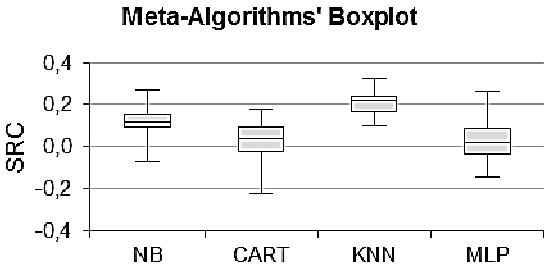


Fig. 2. SRC values boxplot for the meta-algorithms

3.6 Running the Meta-learning System for a New Dataset

After designing the whole meta-learning system, it is possible to use it to recommend an algorithm for a new, unseen dataset. To illustrate this, it was used the Haberman's Survival dataset. Table 5 shows the meta-attributes extracted from the problem data. The system uses these meta-attributes to recommend a ranking for each clustering algorithm (Table 6). So, the system provides information about the algorithms performance without the need to run all algorithms on the new problem.

Table 5. Meta-attributes extracted from the new problem

1	2	3	4	5	6	7	8	9	10
8.26	1.58	0.00	0.00	1.00	0.05	1.06	6.27	0.00	0.00

Table 6. Ranking recommendation for the new problem

KM	SL	CL	ML	SOM
4	1	2	5	3

4 Conclusions

The aim of this work was to analyze the feasibility of applying meta-learning techniques in clustering problems. For this purpose, it was developed a system responsible for extracting the meta-attributes from the problems, obtaining the ranking of the selected clustering algorithms (build the meta-knowledge), and creating a prediction model with the meta-algorithms. Finally, the meta-learning system can make a clustering algorithm recommendation for a new problem just extracting the meta-attributes from it.

As in real-world clustering problems the data labels are not known a priori, and the possibility of recommending clustering algorithms in advance becomes of great importance. The use of meta-learning for clustering can, thus, provide a guide for designing experiments and choosing suitable algorithms for each type of problem based on its features.

The results obtained by the meta-learning system showed that the set of meta-attributes allowed the mapping between the problem features (meta-attributes) and the ranking of the clustering algorithms. The main property of the meta-attributes used in this work is the independence of the objects in relation to their labels, which allows the application of the meta-learning system to any clustering problem. The expansion of this set, keeping this property, may provide better predictions by extracting new dataset intrinsic features.

Acknowledgments. The authors thank CNPq, Capes, Fapesp and MackPesquisa for the financial support.

References

1. Xu, R., Wunsch, D.: Survey of clustering algorithms. IEEE Transactions on Neural Networks, 645–678 (2005)
2. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P.: From data mining to knowledge discovery: an overview. In: Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (eds.) Advances in Knowledge Discovery and Data Mining. American Association for Artificial Intelligence (1996)
3. Brazdil, P., Giraud-Carrier, C., Soares, C., Vilalta, R.: Metalearing - Applications to Data Mining. Springer (2009)
4. Giraud-Carrier, C., Vilalta, R., Brazdil, P.: Introduction to the Special Issue on Meta-Learning. Machine Learning 54(3), 187–193 (2004)
5. Michie, D., Spiegelhalter, D., Taylor, C., Campbell, J. (eds.): Machine Learning, Neural and Statistical Classification, Ellis Horwood, Upper Saddle River, USA (1994)
6. de Souto, M., Prudêncio, R., Soares, R., de Araujo, D., Costa, I., Ludermir, T., Shliep, A.: Ranking and Selecting Clustering Algorithms Using a Meta-Learning Approach. In: IEEE International Joint Conference on Neural Networks, Hong Kong, pp. 3729–3735 (2008)
7. Nascimento, A.C.A., Prudêncio, R.B.C., de Souto, M.C.P., Costa, I.G.: Mining Rules for the Automatic Selection Process of Clustering Methods Applied to Cancer Gene Expression Data. In: Alippi, C., Polycarpou, M., Panayiotou, C., Ellinas, G. (eds.) ICANN 2009, Part II. LNCS, vol. 5769, pp. 20–29. Springer, Heidelberg (2009)
8. Aha, D.: Generalizing from case studies: A case study. In: Proceedings of the 9th Internacional Workshop on Machine Learning, Aberdeen, pp. 1–10 (1992)
9. Brazdil, P., Soares, C., Da Costa, J.: Ranking learning algorithms: Using IBL and meta-learning on accuracy and time results. Machine Learning 50, 251–277 (2003)
10. Frank, A., Asuncion, A.: UCI Machine Learning Repository, `http://archive.ics.uci.edu/ml` (accessed 2010)
11. Kalousis, A.: Algorithm Selection via Meta-Learning. University of Geneva, Genebra (2002)
12. de Souza, B.: Meta-aprendizagem aplicada à classificação de dados de expressão gênica. USP, São Carlos (2010)
13. Amigó, E., Gonzalo, J., Artiles, J., Verdejo, F.: A comparison of extrinsic clustering evaluation metrics based on formal constraints. Information Retrieval 12(5), 461–486 (2009)
14. Spearman, C.: The Proof and Measurement of Association Between Two Things. American Journal of Psychology, 72–101 (1904)
15. Lilliefors, H.: On the Kolmogorov-Smirnov Test for Normality with Mean and Variance Unknown. Journal of the American Statistical Association 62(318), 399–402 (1967)

A Clustering Particle Based Artificial Bee Colony Algorithm for Dynamic Environment

Subhodip Biswas, Digbalay Bose, and Souvik Kundu

Dept. of Electronics and Tele-communication Engineering,
Jadavpur University, Kolkata 700 032, India
{subho.opto.placis91,digbose92,sk210892}@gmail.com

Abstract. Modern day real world applications present us challenging instances where the system needs to adapt to a changing environment without any sacrifice in its optimality. This led researchers to lay the foundations of dynamic problems in the field of optimization. Literature shows different approaches undertaken to tackle the problem of dynamic environment including techniques like diversity scheme, memory, multi-population scheme etc. In this paper we have proposed a hybrid scheme by combining *k*-means clustering technique with modified Artificial Bee Colony (ABC) algorithm as the base optimizer and it is expected that the clusters locate the optima in the problem. Experimental benchmark set that appeared in IEEE CEC 2009 has been used as test-bed and our *ClP*ABC (Clustering Particle ABC) algorithm is compared against 4 state-of-the-art algorithms. The results show the superiority of our *Cl*PABC approach on dynamic environment.

1 Introduction

Nature-inspired metaheuristics [1] have gained prominence in the recent years for real parameter optimization. Swarm Intelligence has its roots in the behaviour of living community and maps their functionality with intelligent adaptation to find solution to problems. Research involving the application of swarm behaviour mostly indulges in using Particle Swarm Optimization (PSO) [2-5] in dynamic environments. Many intelligent modifications were introduced in PSO in the form of learning strategy, multi-swarms and more recently via clustering technique to adapt it to dynamic environment.

ABC devised by Karaboga [6-7], is a new entry into the field of swarm-based metaheuristics and initially dealt with global optimization problems in static environment. It lacked the peak detection ability when an environmental change has occurred. The authors have used a modified form of ABC and synergized the optimizer with *k-means* clustering algorithm [8] to track optima in the new functional landscape.

The adaptation scheme is used by hard-partitioning the swarm agents into predetermined number of cluster and individually optimizes the clusters till they actually converge. On converging they are reinitialized and scattered in the solution space to maintain diversity. It is expected that in an ideal case one of these clusters

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 151–159, 2012.

that converges to the local optima succeeds in tracking the global optima. An additional memory scheme is used in the form of an archive that keeps a track of the locally found optima in each cluster and acts as a reference when a change of environment has occurred. Memory schemes are based on the possibility that local optima in the current environment may be the global optima or the near optimal points in the changed space..The rest of the paper has been grouped into four individual sections each concerning a different aspect of our findings. Section 2 provides a brief review of ABC algorithm while Section 3 discusses the proposed *Cl*P-ABC algorithm highlighting the changes made to the ABC algorithm and how it was incorporated in our approach. Experiment and numerical tabulation is provided in Section 4. Numerical analysis of the algorithm along with a brief discussion is given in Section 5. Lastly, Section 6 concludes our research and deals with further improvements that can be undertaken.

2 Artificial Bee Colony Algorithm: Classical Framework

ABC algorithm is based on the minimal foraging model of honey bees for nectar collection. The basis of ABC lies in dividing the foraging task force into specialized members namely- onlooker bees, forager bees and scouts. The main steps are detailed as below:-

- Unemployed foragers roam about search space in search of food source.
  When found it is promoted to an Employer Bee and local exploitation of the source is carried out, by analyzing food sites based on their nectar content
- A fit food source is chosen , memorizes its position and returns to hive
  where, the nectar is unloaded and in the process the presence of a fit source is communicated to waiting onlookers via waggle dance.
- Onlooker probabilistically selects one of the sources advertized to it
  which is exploited by onlooker and again the steps performed by Employer bees are repeated.
- A food source is abandoned by a forager when it has exhausted its nectar content and it becomes a scout bee performing random walks in search space.

This high exploitation tendency is one of the reasons for high tendency of premature convergence in ABC ,which makes it near impossible to navigate in a dynamic environment to find the global optima, which is tackled by making use of the population topology along with clustering technique to induce a *k*-best motion in our C*l*P-ABC.

3 *Cl*P-ABC Algorithm

The C*l*P-ABC algorithm tries to harmonize two individual ideas- tradeoff between exploration and exploitation, and a hard partitioning of forager population to promote diversity. The steps involved are discussed vividly outlying the modifications made to original ABC followed a brief review of the *k*-means clustering technique used and its implementation.

(a) **Initialize Food Source:** The algorithm begins with generating trial points, in the search space, from where the honey bees begin the task of foraging. Following the formula elucidated initial food sources are set up

$$X^{j}{}_{i} = X^{j}{}_{min} + \mathbf{r}.\,(X^{j}{}_{max} - X^{j}{}_{min}) \tag{1}$$

where $X^{j}{}_{max}$ and $X^{j}{}_{min}$ are the limits of the parameter space for the j^{th} parameter; j∈ [1,D] for a *D*-dimensional problem, i ∈ [1, FN] for *FN* food sources and **r** is a random number uniformly generated in the range [0, 1] and later used as *rand*.

(b) **Employer Phase:** Foragers employed at a particular site carry out exploitation of the food source and are on a constant lookout for fitter food sources. To control and obtain a trade-off between exploration and exploitation (*T: ER & EI*) a less greedy scheme of positional perturbation has been implemented as follows

$$C_i = \begin{cases} \vec{X}_i + \Phi_i \times \left(\vec{X}_i - \vec{X}_k{}^{N}\right) & if\ rand \le pr_{nbd} \\ \vec{X}_i + \Phi_i \times \left(\vec{X}_i - \vec{X}_k{}^{F}\right) & else \end{cases} \tag{2}$$

where $X_k{}^{N}$ and $X_k{}^{F}$ refer to the one of the *k*-nearest or *k*-farthest members based on Euclidean distance from food source X_k that is selected based on the value of the neighbourhood probability pr_{nbd} that is varied according to the equation

$$pr_{nbd} = 0.25 + 0.50 \times \left(\frac{\exp\left(10\frac{Cyc}{MCN}\right)-1}{\exp(10)-1}\right) \tag{3}$$

This modification permits initial exploration followed by exploitation, thus the necessary *T: ER & EI* is obtained. Another important aspect is the frequency of parameter change which was initially fixed for ABC. Slow convergence is the result of single perturbation for every food source, but due to fixed Fitness Evaluations it is desired that the convergence rate is increased. So introduce a selection factor *S* kept constant at 0.6. Now we change a parameter based on the given condition

$$U_{ij} = \begin{cases} C_{ij} & if\ rand \le S \\ X_{ij} & else \end{cases} \tag{4}$$

This is followed by greedy selection between the initial source and new ones found by the foragers based on the fitness value of the food sources. The better one survives the process while the other is rejected obeying laws of evolution. Fitness for the i^{th} food source $\boldsymbol{fit}_{\mathbf{i}}$ is calculated using the given formula (for minimization problem), where $\boldsymbol{f(x)}$ is the associated objective function value.

$$fit_i = \begin{cases} {}^{1}/_{\{1+f(x)\}} & f(x) \ge 0 \\ 1 + |f(x)| & f(x) < 0 \end{cases} \tag{5}$$

(c) **Onlooker Phase**

The onlookers select one among the several food sources advertised before them by a Fitness-proportionate selection method. The probability of selecting a fitter source is more than less fit ones. To map this process in real-world we use the technique of Roulette-Wheel selection. The probability of a source, having fitness value ***fit*** $_i$, being selected is computed as given below

$$prob_i = 0.9 \times \frac{fit_i}{\max(fit_i)} + 0.1 \tag{6}$$

A given food source is calculated if the value of randomly generated number *rand* is less than the value of $prob_i$. Similar processes of perturbation and greedy selection as described in employer phase is carried out again.

(d) **Scout Phase:** Scout phase deals with the rejection of redundant solutions. A parameter called limit is defined and a trial counter is maintained for each solution and is incremented whenever the solution can't be improved. The particular solution whose trial counter exceeds *limit* is reinitialized randomly.

3.1 Clustering and Member Initialization

The ***k-means*** clustering algorithm is used to partition the entire population into four clusters on which the ABC algorithm coupled with the modifications discussed above is made to run separately to explore the regions covered by each cluster. In order that C*l*P-ABC performs satisfactorily in case of each cluster a minimum number of population is maintained in each cluster which is denoted as ***min_mem***. A measure of convergence of each cluster is estimated by calculating the radius of each cluster.. The cluster centroid and cluster radius are calculated as:-

$$centroid_i = \frac{\sum_{j=1}^{N_c} cluster_i^{\,j}}{N_c} \tag{7}$$

$$radius_i = \frac{\sum_{j=1}^{N_c} \left\| centroid_i - cluster_i^{\,j} \right\|}{N_c} \tag{8}$$

i=1,2,3..clust_num; j=1,2,3,….D and N_c=number of members in the i^{th} cluster.

A cluster is considered to be converged if the radius of the cluster becomes less than a predetermined value called R_{conv}. In order that suitable diversity is maintained to track the moving optima, the converged solutions are stored in a memory archive and the corresponding cluster members are reinitialized, since any search process within the converged cluster results in wastage of function evaluations (FEs).

3.2 Archive Operation and Environment Change Detection

The memory archive *mem_Archive* holds the converged solutions of the previous environment which provides a good starting point for the optimization process in a new environment, if the change is not severe. In case a dimensional change occurs the usage of *mem_Archive* becomes redundant and the optimization procedure is started

afresh. The contents of the memory archive *mem_Archive* are emptied after each change in the environment.

For detecting the change in the dynamic environment, a randomly generated particle called *change_detector,* which is initialized within the search space, is maintained and is not subjected to any optimization process and is evaluated continuously. Any mismatch in the objective functional values of the *change_detector* in the current generation when pitted against the previous generation is considered as a change in functional environment. Whenever an environment change occurs, the algorithm reverts back to the initialization phase where a random number from a normal distribution having mean 0 and variance 1 is added to each dimension of population and the worse solutions are replaced by the members of the memory archive *mem_Archive*. After the memory archive has been utilized, the contents of the memory archive are emptied so that it is filled by the converged solutions of the new environment. The k means clustering algorithm is called again to redistribute the population amongst various clusters.

4 Numerical Benchmarks and Evaluation Criteria

4.1 Numerical Benchmark

Literature presents instances of Dynamic Optimization Problems (DOPs) but there is no DOP that initiates dynamic problems across three solution space. For more details on GDBG, the technical report [11] can be referred. There are six test functions overall each with its own instances each involving the change of solution space.

4.2 Parameter Settings

The parametric setting for the proposed C*l*P-ABC framework is given below. The control parameters are based on the performance evaluation of ABC suggested by Karaboga [6]. Test runs have been computed keeping this setting:-

a) Colony Size: - CS=60
b) Cluster number: - clust_num=4
c) Minimum number of cluster members: - min_mem=5
d) Convergence Radius: - R_{conv}=0.01

4.3 Algorithms Compared and Evaluation Criteria

C*l*P-ABC is compared against a set of 4 standard state-of-the-art algorithms, that have exhibited significant performances on GDBG namely-DASA [12], dopt-aiNET [13], CPSO [14] and DynDE [15]. The algorithms have been simulated using their standard parametric set-up on Intel dual-core machine with 2 GB RAM 2.36 GHz speed using MATLAB 7.14.Maximum limit for Function Evaluations have been fixed at (*10000*n*), where n is fixed at 10.

4.4 Numerical Results on Benchmarks

The data tabulated in Table 1, shows the offline error and standard deviation values achieved by the ClP-ABC algorithm along with 4 other algorithms on GBDG system. The participating algorithms are namely DynDE, CPSO, dopt-aiNET and DASA.

Table 1. Offline error and standard deviation values achieved by the C*l*P-ABC and other algorithms tested for benchmark functions over 20 trial runs

Algo / F1(10)	DASA	DynDE	Dopt-aiNET	CPSO	C*l*PABC
T1	0.1854 (1.2501)	0.0732 (2.9567)	0.1353 (1.0061)	0.0351 (0.4262)	**0.0214 (0.3145)**
T2	4.1802 (9.0716)	**2.5567 (8.4313)**	5.8667 (10.2772)	2.7185 (6.5230)	2.8013 (4.1124)
T3	6.3706 (10.7128)	5.4245 (9.2485)	4.2545 (8.1828)	**4.1315 (8.9947)**	6.7245 (9.0123)
T4	0.4873 (1.9504)	0.1263 (0.9426)	5.3563 (8.9414)	**0.0944 (0.7855)**	1.1345 (2.3654)
T5	2.5485 (4.8002)	1.5651 (4.6461)	4.4356 (5.5545)	1.8698 (4.4910)	**0.6714 (4.4231)**
T6	2.3442 (8.6685)	1.3115 (6.2511)	9.9407 (15.8214)	1.1569 (4.8054)	**1.1145 (3.8751)**
T7	4.8452 (8.9662)	4.1137 (8.5249)	4.2110 (8.6873)	4.5401 (9.1194)	**3.7721 (7.8453)**

Algo / F1(50)	DASA	DynDE	Dopt-aiNET	CPSO	C*l*PABC
T1	0.4425 (1.3911)	0.3286 (1.5224)	0.3644 (0.9725)	0.2624 (0.9362)	**0.1345 (0.7645)**
T2	4.8661 (7.0052)	4.6547 (6.3453)	4.7485 (6.7580)	3.2792 (5.3034)	**3.1452 (5.0413)**
T3	8.4247 (9.5682)	6.4641 (9.3523)	**5.2532 (6.6830)**	6.3198 (7.4420)	7.4436 (9.4567)
T4	0.5853 (1.0901)	0.1412 (0.5914)	2.6565 (5.9773)	**0.1255 (0.3859)**	3.3456 (6.7458)
T5	1.1832 (2.1818)	1.0162 (2.6489)	2.8641 (4.1597)	0.8481 (1.7790)	**0.6967 (1.1345)**
T6	2.0728 (5.9719)	**0.9859 (4.8631)**	6.8830 (11.8790)	1.4821 (4.3932)	3.3457 (4.8765)
T7	7.8412 (9.0558)	6.2513 (9.0651)	4.8172 (6.4528)	6.6467 (7.9411)	**3.4572 (5.7684)**

Algo / F2	DASA	DynDE	Dopt-aiNET	CPSO	C*l*PABC
T1	3.3017 (8.7885)	1.3627 (5.0315)	0.0984 (0.0291)	1.2475 (4.1780)	**1.0839 (3.3784)**
T2	25.6105 (83.2124)	13.0179 (48.2532)	8.1209 (14.3832)	10.1055 (35.0601)	**5.2678 (7.7893)**
T3	18.9904 (67.8204)	11.9214 (45.7054)	17.9979 (62.2599)	10.2725 (33.4527)	**2.2411 (4.5673)**
T4	1.4512 (3.8311)	0.7842 (2.2248)	1.0652 (2.8269)	**0.5664 (2.1371)**	22.3456 (45.1745)
T5	49.6022 (112.413)	**20.7842 (64.5341)**	101.384 (134.518)	25.1424 (64.2501)	75.3720 (114.134)
T6	2.1182 (5.2912)	2.1845 (3.9643)	6.5192 (13.8172)	1.9871 (5.2175)	**1.3567 (3.0123)**
T7	3.8752 (8.1285)	**2.4235 (7.1031)**	3.7385 (7.9542)	3.6510 (6.9274)	4.4732 (9.7894)

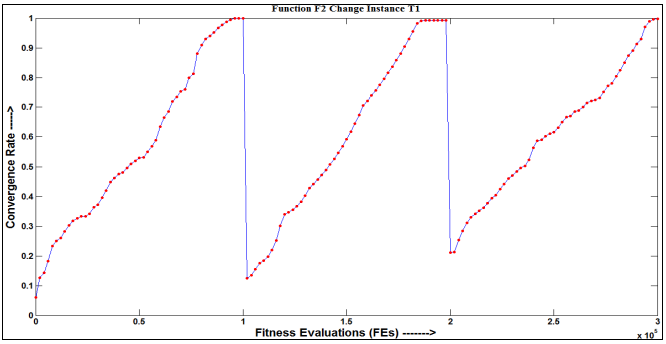
Algo / F3	DASA	DynDE	Dopt-aiNET	CPSO	C*l*PABC
T1	**15.7025 (67.1131)**	21.2512 (73.6549)	810.830 (66.1232)	137.5272 (221.621)	17.4532 (56.4312)
T2	824.389 (204.035)	792.457 (255.616)	1078.321 (64.2114)	855.1229 (161.101)	**617.4312 (202.346)**
T3	688.358 (298.012)	635.614 (342.775)	1071.912 (64.9950)	766.0521 (236.213)	**591.926 (223.272)**
T4	435.488 (441.212)	341,701 (419.811)	1030.521 (259.256)	431.4012 (439.371)	**318.453 (412.354)**
T5	697.210 (315.422)	749.265 (280.918)	1022.911 (58.2131)	859.2704 (363.234)	**648.564 (324.237)**
T6	626.120 (460.621)	**519.550 (438.247)**	1187.010 (291.623)	753.5041 (358.639)	631.342 (253.419)
T7	433.252 (380.223)	**415.324 (390.345)**	1061.33 (121.091)	654.2231 (329.435)	524.681 (745.172)

Algo / F4	DASA	DynDE	Dopt-aiNET	CPSO	C*l*PABC
T1	5.6012 (26.5331)	1.8616 (5.7531)	1.4277 (4.5459)	2.6671 (7.0552)	**1.2936 (6.0523)**
T2	65.6105 (160.128)	39.5923 (98.6312)	122.440 (201.627)	37.1512 (99.4352)	**20.9492 (32.8612)**
T3	53.6158 (140.015)	23.4921 (94.5314)	98.6688 (196.695)	36.6711 (97.1805)	**2.0521 (4.1625)**
T4	3.8512 (4.2258)	0.9691 (3.1723)	4.2623 (9.7255)	**0.7926 (2.7752)**	2.0524 (4.7732)
T5	118.212 (178.251)	**44.6713 (121.716)**	304.556 (203.243)	67.1702 (130.306)	83.2880 (134.679)
T6	2.9812 (7.5942)	1.5624 (6.2151)	12.6491 (55.8367)	4.8814 (15.3965)	**1.4231 (5.4465)**
T7	27.4432 (90.2513)	6.5213 (26.5951)	52.9011 (130.593)	12.7924 (19.2105)	**5.0413 (8.7944)**

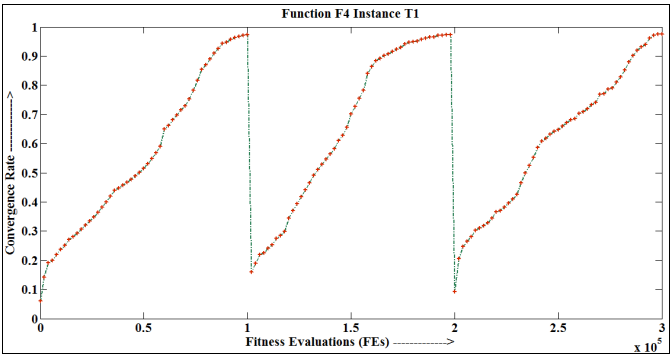
Algo / F5	DASA	DynDE	Dopt-aiNET	CPSO	C*l*PABC
T1	0.9551 (3.4310)	2.9929 (6.8831)	40.8943 (221.212)	1.8559 (5.1812)	**1.7047 (4.3824)**
T2	2.0199 (4.0504)	2.9418 (4.7179)	34.4531 (119.896)	2.8791 (6.7875)	**2.1993 (4.5632)**
T3	1.9492 (3.3135)	2.9125 (5.3886)	34.9420 (115.025)	3.403 (6.4480)	**0.4965 (3.4562)**
T4	**0.3928 (1.6185)**	1.3796 (2.4196)	130.637 (293.542)	1.0954 (4.8651)	2.1367 (5.4431)
T5	2.3052 (6.3610)	**8.4378 (12.1232)**	943.223 (633.318)	7.9869 (13.8171)	2.0762 (5.6432)
T6	0.4671 (1.17346)	2.3049 (3.6182)	480.305 (610.821)	4.0535 (8.3719)	**0.0009 (0.0023)**
T7	1.1128 (3.7621)	**0.5214 (0.7135)**	219.462 (427.817)	6.5278 (22.8120)	2.2345 (3.2456)

F6 / Algo	T1	T2	T3	T4	T5	T6	T7
DASA	8.8752 (13.3191)	37.128 (122.147)	26.7341 (98.4018)	9.7442 (22.0541)	**37.9103 (118.015)**	13.3481 (57.4802)	17.7422 (36.7159)
DynDE	6.0471 (11.0458)	**30.2205 (62.2093)**	19.3782 (67.3585)	8.8731 (26.6683)	43.3514 (136.906)	**12.1784 (25.2617)**	15.3644 (21.0742)
Dopt-aiNET	20.4434 (79.3234)	391.195 (395.435)	456.443 (405.038)	83.9701 (220.177)	845.862 (251.211)	482.201 (434.421)	372.471 (394.662)
CPSO	6.7254 (9.9747)	31.5738 (63.5115)	27.1358 (83.9873)	9.2742 (24.2344)	71.5704 (160.321)	23.6757 (51.5521)	32.5842 (76.9105)
C*l*PABC	**4.5789 (8.9845)**	34.7415 (65.6789)	**23.1345 (52.4567)**	**7.4423 (26.4567)**	102.346 (123.446)	22.0894 (33.5446)	**13.9074 (20.0978)**

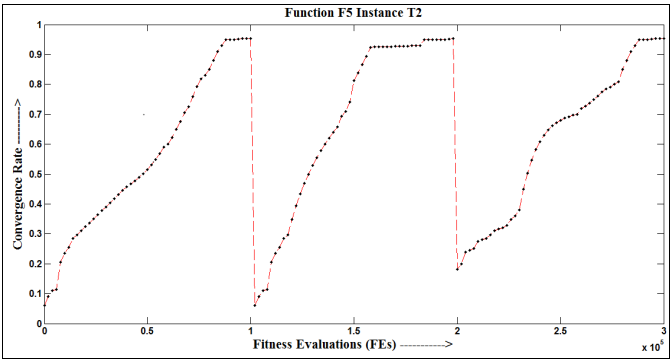
CONVERGENCE RATE PLOT OF C*I*P-ABC ALGORITHM FOR SOME OF THE TEST INSTANCES PLOTTED AGAINST FITNESS EVALUATIONS



Function F2 Instance T1



Function F4 Instance T1



Function F5 Instance T2

Absolute function error value denote the difference between the actual optima of a function (f *) and the best value reached by the algorithm (f_{best}) and is calculated by the formula $|f^* - f_{best}|$. Offline error calculated as follows

$$Offline\ Error = \sum_{i=1}^{runs}\left(\sum_{j=1}^{num_{change}} \frac{E_{i,j}^{last}(t)}{runs - num_{change}}\right) \tag{9}$$

where $E_{i,j}^{last} = |f^* - f_{best}|$ is the absolute function error value for each environment.The mean gives a measure of the uniformity in the perofrmance of the optimizer for a varied set of environents. The tabulated data in Table 1 and 2 is the arithmetic mean taken over 20 test runs for each of the competing optimizers.

5 Evaluation of C*l*P-ABC on GDBG Benchmark

The enhanced version of C*l*P-ABC proposed here overcomes the problem of local trap by the intelligent use of the neighbourhood probability operator, which obtains a perfect synergy between the exploration and exploitation phases. Another significant problem of slow convergence in ABC is solved by the usage of the Selection Factor parameter allowing more than one parameter perturbation to take place simultaneously.

Discussion. The *k*-means clustering technique enable a similarity based grouping guiding towards better local search by the bees in a locality than the participating optimizers. External archiving can be incorporated into any optimizer that may store a fit location and helps to reach the optima in lesser time than otherwise needed. In fact use of the archive improves the performance of ClP-ABC significantly. Among the competing algorithms C*l*P-ABC succeeded in obtaining the best results in 30 out of 49 possible cases which accounts for 61.22%. DynDE comes second with 8 best cases, followed by CPSO and DASA with 5 and 3 best cases respectively. Hence C*l*P-ABC is more efficient in dynamic environment than the four competing algorithms.

6 Conclusion and Future Work

Neighbourhood information relies on population topology and the tendency of optimizers to cluster swarm agents towards the basins of attrcation is used here. To prevent overlapping and ensure distinct sub-population formation we make use of the *k*-means clustering. Invoking dissimilarity by hard partitioning helps in improving the peak detection ability of the optimizer, guiding it towards the global optima. To the best of the authors' knowledge, the proposed C*l*P-ABC framework is the first attempt made to solve dynamc benchmark problems (GDBG) by incorporating clustering into basic ABC framework. Future work would stress on application of self-adaptive scheme to parameters like Selection Factor, Food Number and the magnitude of perturbation. Along with replacing the clustering scheme by the use of sub-population capable of maintaining local diversity. Improving the clustering techniques or use of different clustering methods like k-medioid, hierarchical or fuzzy c-means methods can also be analyzed on the proposed optimizer.

References

1. Ghazali Talbi, E.: Metaheuristics-From Design to implementation. John Wiley and Sons (2009)
2. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, Piscataway, NJ, pp. 1942–1948 (1995)
3. Engelbrecht, A.: Fundamentals of Computational Swarm intelligence. John Wiley and Sons, UK (2005)
4. Clerc, M., Kennedy, J.: The particle swarm-explosion, stability and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation, 58–73 (2002)
5. Kennedy, J., Eberhart, R.: Swarm Intelligence. Morgan Kauffman, San Francisco (2001)
6. Karaboga, D., Basturk, B.: On the performance of artificial bee colony (ABC) algorithm. Applied Soft Computing 8, 687–697 (2008)
7. Karaboga, D., Basturk, B.: A powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm. Journal of Global Optimization 39(3) (2007)
8. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: A review. ACM Computing Survey, 264–333 (1999)
9. Branke, J.: Memory enhanced evolutionary algorithms for changing optimization problems. In: IEEE Congress on Evolutionary Computation (1999)
10. Yang, S., Ong, Y.S., Jin, Y.: Evolutionary Computation in Dynamic and Uncertain Environment. Springer, Berlin (2007)
11. Li, C., Yang, S., Nguyen, T.T., Yu, E.L., Yao, X., Jin, Y., Beyer, H.G., Suganthan, P.N.: Benchmark Generator for CEC 2009 Competition on Dynamic Optimization, University of Leicester, University of Birmingham, Nanyang Technological University, Technical Report (2008)
12. Korosec, P., Silc, J.: The differential ant-stigmergy algorithm applied to dynamic optimization problems. In: IEEE Congress on Evolutionary Computation, pp. 407–410 (2009)
13. de Franca, F.O., Von Zuben, F.J.: A dynamic artificial immune algorithm applied to challenging benchmarking problems. In: IEEE Congress on Evolutionary Computation, pp. 423–430 (2009)
14. Yang, S., Li, C.: A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments. IEEE Transactions on Evolutionary Computation 14(6) (2010)
15. Mendes, R., Mohais, A.S.: DynDE: a differential evolution for dynamic optimization problems. In: IEEE Congress on Evolutionary Computation, pp. 2808–2815 (2005)

A Selective Teaching-Learning Based Niching Technique with Local Diversification Strategy

Souvik Kundu, Subhodip Biswas, Swagatam Das, and Digbalay Bose

Faculty of Engg. & Technology,
Jadavpur University,
Kolkata 700 032, India
{sk210892,subho.opto.placis91,digbose92}@gmail.com

Abstract. Real world problems present instances where more than one optimal solution can be obtained for a system under consideration so as to switch between them without considerably affecting efficiency. In such instances the idea of *niching* provides a solution. In this paper we propose a swarm-based niching technique that enhances diversity by Teaching and Learning strategy that adapts to the local neighbourhood by controlled exploitation and the knowledge learned helps to preserve population diversity. Our algorithm, imitates the local-explorative swarm behaviour to hover around local sites in groups, exploiting the peaks with high degree of accuracy, is called TLB-*l*DS (Teaching-Learning Based Optimization with Local Diversification Strategy), without using any niching parameter. TLB-*l*DS algorithm is compared against sophisticated niching algorithms tested on a set of standard numerical benchmarks.

1 Introduction

The complete definition of *niching*, was first given by Preus [1], as "a two-step procedure that (a) *concurrently or subsequently distributes individuals onto distinct basins of attraction* and (b) *facilitates approximation of the corresponding local optimizers*". Researchers have studied ecological niche for deciphering the inherent dynamics of Evolutionary processes. It was found that Evolutionary Algorithms suffers from the loss of diversity converging to single solution [2, 3]. Shir pointed out that the main focus of niching is "attaining the optimal interplay between partitioning the search-space into niches occupied by stable sub-populations, by means of population diversity preservation, and exploiting the search space in each niche by means of a highly efficient optimizer with local-search capabilities" [4, 6]. Sewall Wright [7] was the first to think of distribution of fitness values as a kind of landscape, *fitness landscape,* which was later mapped onto Evolutionary Optimization [8] techniques.

Following the laws of exclusion principle [9], which states that no two species can remain within a common niche above a predetermined duration of time and must be separated by a realized niché width, many niching algorithms have been devised with *niche radius* as a parameter for maintaining diversity and locating local peaks.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 160–168, 2012.

2 Revisiting Classical Niching Techniques

Niching techniques have been studied over the past years and attempts have been made to categorize them based on their philosophy of functionality. Broadly speaking *sequential niching* and *parallel niching* are two categories of niching techniques, the former needing more computational effort. Crowding niching technique was pioneered by De Jong [10] in 1975, focuses on limiting alterations in the population in subsequent generations by instantiating competition among similar individuals followed by a restricted replacement of randomly sampled members by the superior offspring. Similarity is Euclidean distance-based. Fitness sharing was initially introduced by Holland [11] and later improved by Goldberg and Richardson [12], focuses on splitting the population into a number of species. The members of a niche must share information, via sharing function. Petrowski [13] introduced a greedy version of *fitness sharing* technique, called *clearing* based on the *winner takes it all* philosophy. The members are sorted in descending order of fitness value and those falling within a threshold *clearing radius* $\mathbf{R}_{clear}$ are cleared. Speciation [14] segregates the population into a species based on Euclidean distance with the centre of the species as the species seed and individuals falling within a certain species radius $\mathbf{R}_s$ are treated as members of that species.

Other niching techniques, like clustering, restricted tournament selection and derating, are still in play when it comes to multimodal optimization. Das *et. al* [15] made a comprehensive survey of the niching techniques for multimodal landscapes.

3 TLB-*l*DS Algorithm

Our proposed TLB-*l*DS algorithm is based on Teaching-learning-based optimization recently proposed by Rao *et al.* [5] inspired from the teaching-learning ability of teacher and learners in an institution. It is a population based method technique where the parameters are treated as subjects offered to learners and the *fitness* value is the result of learner. It is divided into two parts-*Teacher phase* and *Learner phase*.

(a) **Teacher Phase:** The teacher motivates the learner to increase the mean of the learners from its existing value $\boldsymbol{M_L}$ to his or her level $\boldsymbol{M_T}$. Difference mean is

$$Mean^{\,\mathrm{DIF}}{}_i = \mathrm{rnd}_i \,.\, (\, M_T - \mathbf{T_F} \,.\, M_L \,) \tag{1}$$

where $\mathbf{T_F}$ is the Teaching Factor set to 1 or 2 with equal probability and rnd_i is the random number in the range [0,1]. Solutions are updated at C+1th iteration as-

$$\beta_i{}^{C+1} = \beta_i{}^{C} + Mean^{\,\mathrm{DIF}}{}_i \tag{2}$$

(b) **Learner Phase:** A learner can interact with a more knowledgeable and learn new ideas from him that will enhance his or her knowledge. At C^{th} iteration, if β_i and β_j are two different learners, then the learning phenomenon is expressed as-

$$\beta_i{}^{C+1} = \beta_i{}^{C} + \mathrm{rnd}_i \,.\, (\beta_i{}^{C} - \beta_j{}^{C}) \quad \text{if } \mathrm{f}(\beta_i) < \mathrm{f}(\beta_j) \tag{3}$$

$$\beta_i{}^{C+1} = \beta_i{}^{C} + \mathrm{rnd}_i \,.\, (\beta_j{}^{C} - \beta_i{}^{C}) \quad \text{if } \mathrm{f}(\beta_j) < \mathrm{f}(\beta_i) \tag{4}$$

$\beta_i{}^{C+1}$ is accepted if it provides better functional value.

TLB-*l*DS: Diversifying TLBO Algorithm for Niching. In original TLBO proposed, the Teacher Phase exerts an attractive pull on the Learners assimilating them to best-fit solution. Although Learner phase compensates for this pull by randomizing interaction between the learners, but instances of missing peaks suggests that the compensation is not adequate for making TLBO a strong niching algorithm by its own.

In *local* Diversification Strategy (*l*DS) proposed here, a distance based matrix *Dist* holding the members $\boldsymbol{\theta_j}$ sorted in ascending order with respect to member $\boldsymbol{\theta_i}$, (i≠j), is created. A *local* colony, comprising of member $\boldsymbol{\theta_i}$ and its nearest *k*-best neighbours, is generated and the TLBO optimizer is applied on this colony and it returns the fittest member $\mathbf{X}_{\text{best}}$ in the colony along with the updated *k*-best colony **X.** Trade-off: exploration and exploitation (*T:ER&EI*) is obtained by varying the value of *k* as-

$$k = NP \times \left\{0.35 - 0.25\left(\frac{-1 + e^{10\left(\frac{iter}{iter^{max}}\right)}}{-1 + e^{10}}\right)\right\}$$

from 35 percent to 10 percent of total population *NP*. The updated colony and it's best fit member is utilized in perturbing the position of the concerned member as-

$$V = \vec{\theta}_{\text{iter}}{}^{I} + C_1 \times \left(\vec{X}_{\text{iter}}{}^{\text{best}} - \vec{\theta}_{\text{iter}}{}^{I}\right) + C_2 \times \left(\vec{X}_{\text{iter}}{}^{\text{r1}} - \vec{X}_{\text{iter}}{}^{\text{r2}}\right)$$

X^{r1} and X^{r2} are two randomly chosen individuals from the updated *k*-best colony. The value of C_1 is 0.005 and C_2 is 0.08 based on experimental observations. A greedy selection between member $\boldsymbol{\theta_i}$ and $\boldsymbol{V}$ occurs to choose the fitter one between them.

The advantage of TLB-*lDS* algorithm is that it does away with the use of any niching parameters that require before-hand knowledge of fitness landscapes.

3.1 Pseudo-code of TLB-*lDS* Algorithm

```
 const  c1,c2: Real;  NP, it_max: Integer;
var    k:= 0.35*NP;
begin
  iter := 1;
  Initialize Population pop;
  Evaluate Fitness Matrix; WriteLn;
  repeat
       member=pop(i);
       repeat
          colony := GenColony(member, pop, k);
          X := TLBO(member, colony, fitness);
         [r1, r2]:= ceil(k*rand(1,2));
   v:= member+ c1*(Xbest-member)+ c2*(X(r1)-X(r2));
         v := checkBound(v, UBnd, LBnd);
            if fitness(v)>fitness(member)
               pop(i):=v,     fitness(i):=fitness(v);
            end if
       i:=i+1;
       until i=NP
  until iter=it_max
end.
```

4 Numerical Benchmarks and Comparative Study

4.1 Numerical Benchmark Set

The test functions used in this paper for the comparison of TLB-*l*DS algorithm consists of 18 multimodal functions and 7 composite functions, encompassing ruggedness, sink basin, local trapping, irregularity, unequal optima etc. Since number of local peaks varies greatly with the dimension of problem, we deal with global peaks only.

4.2 Algorithms Compared

TLB-*l*DS is made to compete against a standard set of niching algorithms, followed by a comparative study of its performance from Tables 4 and 5. Algorithms taken into consideration are CMA-ES with Self Adaptive Niche Radius (SCMA-ES) [16], Sharing DE (ShDE) [17], Crowding DE (CDE) [18], Speciation-based DE (SDE) [19], Speciation-based PSO (SPSO) [20], Fitness-Euclidean distance Ratio PSO (FER-PSO) [20] and *l*best-PSO variants with ring-topology [20] namely r2pso, r3pso, r2pso-lhc and r3pso-lhc (lhc versions do not have overlapping neighborhood).

Table 1. Mathematical Description of Test Function Set 1

Name	Dim	Test Function	Range	No. of global peaks
f_1: Two-peak trap	1	$f_1(x) = \begin{cases} \frac{160}{15}(15-x) & for\ 0 \le x \le 15 \\ \frac{200}{5}(x-15) & for\ 15 \le x \le 20 \end{cases}$	$0 \le x \le 20$	1
f_2: Central Two-peak trap	1	$f_2(x) = \begin{cases} \frac{160}{10}x & for\ 0 \le x \le 10 \\ \frac{160}{5}(15-x) & for\ 10 \le x \le 15 \\ \frac{200}{5}(x-15) & for\ 15 \le x \le 20 \end{cases}$	$0 \le x \le 20$	1
f_3: Five-uneven-peak trap	1	$f_3(x) = \begin{cases} 80(2.5-x) & for\ 0 \le x \le 2.5 \\ 64(x-2.5) & for\ 2.5 \le x \le 5 \\ 64(7.5-x) & for\ 5 \le x \le 7.5 \\ 28(x-7.5) & for\ 7.5 \le x \le 12.5 \\ 28(17.5-x) & for\ 12.5 \le x \le 17.5 \\ 32(27.5-x) & for\ 22.5 \le x \le 27.5 \\ 80(x-27.5) & for\ 27.5 \le x \le 30 \end{cases}$	$0 \le x \le 30$	2
f_4: Equal maxima	1	$f_4(x) = \sin^6(5\pi x)$	$10 \le x \le 1$	5
f_5: Decreasing maxima	1	$f_5(x) = \exp\left[-2\log(2).\left(\frac{x-1}{0.8}\right)^2\right].\sin^6(5\pi x)$	$0 \le x \le 1$	1
f_6: Uneven Maxima	1	$f_6(x) = \sin^6(5\pi(x^{\frac{3}{4}} - 0.05))$	$0 \le x \le 1$	5
f_7: Uneven Decreasing Maxima	1	$f_7(x) = \exp\left[-2\log(2).\left(\frac{x-1}{0.8}\right)^2\right].\sin^6(5\pi(x^{\frac{3}{4}} - 0.05))$	$0 \le x \le 1$	1
f_8: Himmelblau's function	2	$f_8(\vec{X}) = 200 - (x_1^2 + x_2 - 11)^2 - (x_2^2 + x_1 - 7)^2$	$-4 \le x_1, x_2 \le 4$	4
f_9: Six-hump Camel Back	2	$f_9(\vec{X}) = -4\left[\left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1.x_2 + (-4 + 4x_2^2)x_2^2\right]$	$-1.9 \le x_1 \le 1.9$ $-1.1 \le x_2 \le 1.1$	2
f_{10}: Shifted Rastrigin's	2	$f_{10}(\vec{X}) = \sum_{i=i}^{D}(x_i^2 - 10\cos(2\pi x) + 10) + f\_bias$	$-5 \le x \le 5$	1
f_{11}: Brainin RCOS	2	$f_{11}(\vec{X}) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	$-5 \le x_1 \le 10$ $0 \le x_2 \le 15$	3
f_{12}: Shekel's Fox-hole	2	$f_{12}(\vec{X}) = 500 - \frac{1}{0.002 + \sum_{i=0}^{24}\frac{1}{1+i+(x_1-a(i))^6+(x_2-b(i))^6}}$ where a(i) = 16(i mod 5 -2)and b(i) = 16$\left(\left\lfloor\frac{i}{5}\right\rfloor - 2\right)$	$-65.536 \le x_1 x_2, \le 65.536$	1
Inverted Vincent Function				
f_{13}: 1D	1	$f_{13-15}(\vec{X}) = \frac{1}{n}\sum_{n=1}^{\infty}(\sin(10\log x_i))$ where n is the dimensionality of the problem	$0.25 \le x_i \le 10$	6
f_{14}: 2D	2			36
f_{15}: 3D	3			216
f_{16}: Waves	2	$f_{16}(\vec{X}) = (0.3x_1)^3 - (x_1^2 - 4.5x_2^2)x_1x_2 - 4.7\cos(3x_1 - x_2^2(2 + x_1)\sin(2.5\pi x_1))$	$-0.9 \le x_1 \le 1.2$ $-1.2 \le x_2 \le 1.2$	1
f_{17}: Michaelwicz	2	$f_{17}(\vec{X}) = \sin(x_1)\sin^{20}\left(\frac{x_1^2}{\pi}\right) + \sin(x_2)\sin^{20}\left(\frac{2x_2^2}{\pi}\right)$	$0 \le x_i \le \pi$	1
f_{17}: Ursem F1 in [Ursem 1999]	2	$f_{17}(\vec{X}) = \sin(2x_1 - 0.5\pi) + 3\cos(x_2) + 0.52x_1$	$-2.5 \le x_1 \le 3$ $-2 \le x_2 \le 2$	1

Table 2. List of Test Function Set 2

Name	Dim	Test functions	Range	No. of global peaks
CF1	10	Corresponds to CF1 of [6]	$-5 \le x_i \le 5$	8
CF2	10	Corresponds to CF2 of [6]	$-5 \le x_i \le 5$	6
CF3	10	Corresponds to CF3 of [6]	$-5 \le x_i \le 5$	6
CF4	10	Corresponds to CF4 of [6]	$-5 \le x_i \le 5$	6
CF5	10	Corresponds to CF5 of [6]	$-5 \le x_i \le 5$	6
CF6	10	Corresponds to CF6 of [6]	$-5 \le x_i \le 5$	6
CF7	10	Corresponds to CF9 of [6]	$-5 \le x_i \le 5$	6

4.3 Performance Criteria

Niching algorithms are judged based on two criteria of performance as given below:-

1) **Success Rate:** This is expressed in percentage of total runs in which all global optima were found successfully as tabulated in Table 4.
2) **Average Number of Optima Found: -** This performance criterion attributes the robustness of niching algorithms. Table 5 presents this data.

All algorithms are executed using MATLAB 7.5 on Intel i5-760 machine with 2 GB RAM and 2.76 GHz speed. The data assimilated based on 50 independent runs.

4.4 Parametric Set Up

To normalize performances of competing algorithms it is mandatory to initialize them using the same number of initial population and execute them till maximum number of Function Evaluations are met. Details corresponding to each function are specified in Table 3. The level of accuracy (ε) and niche radius (r for SPSO and SDE) are also specified. ε determines the accuracy of detection and is kept very small for precision.

Table 3. Parametric Set-up for Benchmark Execution

Function Number	ε	r	Population Size	Maximum FES
$f_1 - f_3$	0.05	0.5	50	10000
$f_4 - f_7$	0.000001	0.01	50	10000
f_8	0.0005	0.5	50	10000
f_9	0.000001	0.5	50	10000
f_{10}	0.1	1	600	300000
f_{11}, f_{12}	0.1,0.00001	0.5	50	10000
f_{13}	0.0001	0.2	100	20000
f_{14}	0.001	0.2	500	200000
f_{15}	0.001	0.2	1000	400000
f_{16}	0.1	0.5	600	300000
f_{17}, f_{18}	0.1	0.5	50	10000
$CF1 - CF7$	0.5	1	600	300000

Table 4. Success rate of the competing algorithms on Benchmark Set

Func	SCMA-ES	ShDE	CDE	SDE	FER-PSO	SPSO	r2pso	r3pso	r2pso-lhc	r3pso-lhc	TLB-*l*SE
f_1	**100**	**100**	**100**	**100**	72	48	76	84	56	60	**100**
f_2	**100**	**100**	**100**	**100**	**100**	44	88	96	44	56	**100**
f_3	**100**	**100**	**100**	96	20	4	8	8	4	8	**100**
f_4	0	4	28	72	84	88	92	88	**100**	92	**100**
f_5	100	44	72	100	**100**	**100**	**100**	**100**	**100**	**100**	**100**
f_6	0	8	28	60	**100**	92	88	72	92	92	**100**
f_7	96	40	60	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**
f_8	44	0	0	72	72	0	28	24	28	24	**100**
f_9	**100**	0	0	**100**	96	0	56	60	56	52	**100**
f_{11}	74	64	82	78	76	70	72	66	60	62	**92**
f_{12}	4	96	52	32	**100**	56	88	76	72	60	**100**
f_{13}	0	68	56	48	60	72	68	56	52	48	**100**
f_{14}	0	**92**	8	0	0	0	0	0	0	0	**92**
f_{15}	0	0	0	0	0	0	0	0	0	0	0
f_{16}	60	56	80	64	62	52	48	50	46	54	**96**
f_{17}	70	84	84	82	76	64	58	52	42	48	**92**
f_{18}	72	88	86	78	80	68	62	58	52	56	**100**

4.5 Numerical Results on Benchmarks and Comparative Study

The data tabulated in Table 4, shows TLB-*l*DS to be efficient in maitaining population diversity with a high success rate on 17 out of 18 cases. The Diversification Strategy incorporated makes it robust to cope with the varied terrain maintaining balanced *T(ER&EI),* enabling it to locate peaks with sufficient accuracy. The average number of peaks found, as given in Table 5, is an indicator of the

Table 5. Average Number of Peaks Detected

Func	SCMA-ES	ShDE	CDE	SDE	FER-PSO	SPSO	r2pso	r3pso	r2pso-lhc	r3pso-lhc	TLB-*l* SE
f_1	**1**	**1**	**1**	**1**	0.72	0.48	0.76	0.84	0.56	0.60	**1**
f_2	**1**	**1**	**1**	**1**	**1**	0.44	0.88	0.96	0.44	0.56	**1**
f_3	2	2	2	1.96	0.8	0.24	0.48	0.60	0.48	0.60	**2**
f_4	0.04	3.28	3.84	4.72	4.84	4.88	4.92	4.88	5	4.92	**5**
f_5	**1**	0.44	0.72	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
f_6	0	3.28	3.96	4.6	5	4.92	4.88	4.72	4.92	4.88	**5**
f_7	0.96	0.4	0.6	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
f_8	3.44	0.16	0.32	3.72	3.68	0.84	2.92	2.76	3	3.12	**4**
f_9	**2**	0.04	0.04	**2**	1.96	0.08	1.44	1.56	1.56	1.48	**2**
f_{10}	0.04	0.12	0.12	0.16	0.08	0	0	0	0	0	**0.24**
f_{11}	2.56	2.64	**2.76**	2.72	2.64	2.48	2.52	2.44	2.36	2.40	2.60
f_{12}	0.04	0.96	0.52	0.32	**1**	0.56	0.88	0.60	0.72	0.60	**1**
f_{13}	1.52	5.6	5.56	4.88	5.36	5.6	5.52	5.16	5.36	5.28	**6**
f_{14}	1.4	**35.92**	33.8	22.8	23.6	25.7	21.8	22.2	22.5	23.1	**35.92**
f_{15}	0.04	**197.88**	152	50.6	68.6	70.1	40.6	45.4	42.2	43.3	140.4
f_{16}	0.60	0.76	0.80	0.64	0.62	0.52	0.48	0.50	0.46	0.54	**1**
f_{17}	0.70	0.84	0.84	0.82	0.76	0.64	0.68	0.52	0.54	0.58	**0.92**
f_{18}	0.72	0.78	0.86	0.78	0.80	0.68	0.62	0.58	0.52	0.56	**0.92**
CF1	**2**	0	0	1.79	1.08	0	0	0	0	0	**2**
CF2	1.9	1.1	1.2	1.2	2	0	0	0	0	0	**2.8**
CF3	2.7	1.11	0.7	1.5	2.5	0	0	0	0	0	1.8
CF4	0.2	0	0	0	0	0	0	0	0	0	**4.8**
CF5	1.9	1.3	1.1	1.3	**2**	0	0	0	0	0	**2**
CF6	2.6	0	0	1.4	1.2	0	0	0	0	0	**3**
CF7	1.7	0	0	**1.8**	1.5	0	0	0	0	0	0

Distribution of Members with Function Evaluations for 1D functions

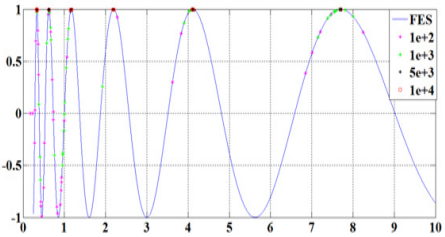


Fig. 1a. 1D Inverted Vincent function (f_{13})

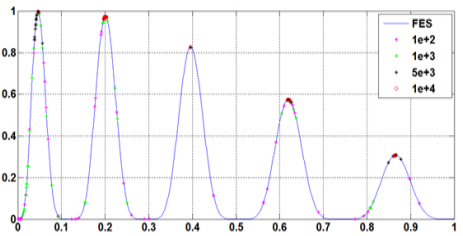


Fig. 1b. Decreasing Maxima (f_5)

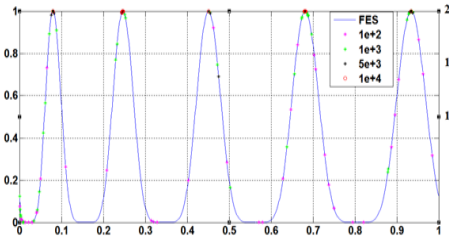


Fig 1c. Unequal Maxima (f_6)

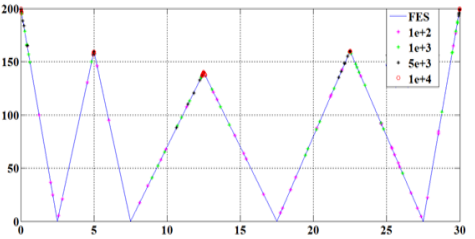


Fig. 1d. Five Un-even-Peak-Traps (f_3)

Distribution of members (2D) plotted with Increasing Function Evaluations

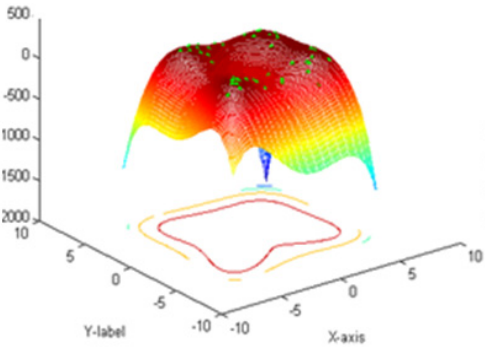


Fig. 2a. After 500 Function Evaluations

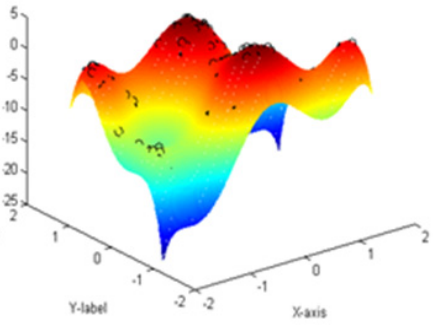


Fig. 3a. After 500 Function Evaluations

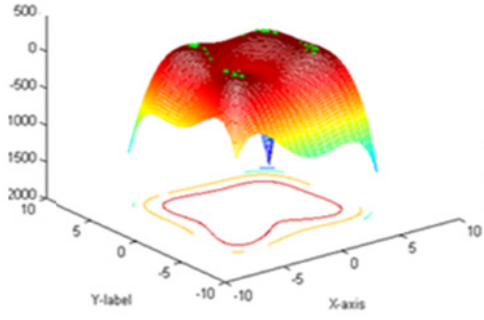


Fig. 2b. After 1500 Function Evaluations

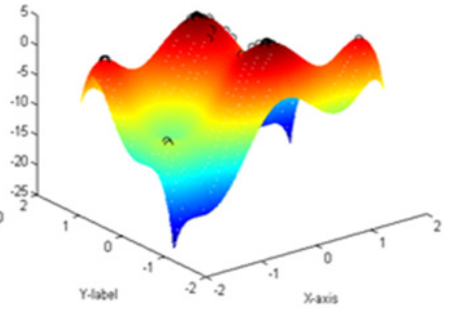


Fig. 3b. After 1500 Function Evaluations

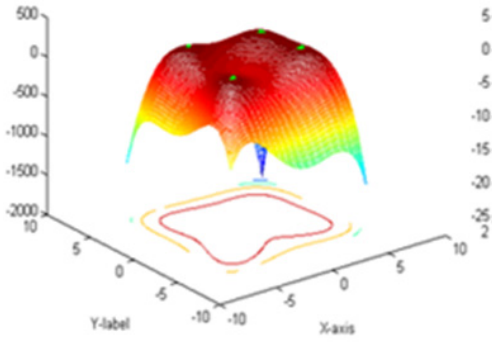


Fig 2c. After 1500 Function Evaluations

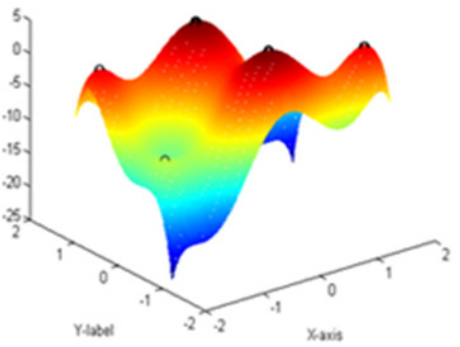


Fig. 3c. After 1500 Function Evaluations

Fig. 2. Himmelblau's Function (f_8) and **Fig. 3.** Six-hump- Camel Back (f_9)

uniformity in the performance of niching methods.Nearer is it's value to the no. of global peaks, better is it's peak detectability. TLB-*l*DS obtains a majority of best cases, 21 out of 25 cases.

5 Conclusion

The TLB-*l*DS algorithm proposed in this paper, presents a niching algorithm that is free from the usage of any niching parameters (like *radius*) and yet provides a superior functionality on a wide variety of fitness landscapes pitted against standard niching algorithm. The teaching-learning strategy coupled with the *l*DS-scheme enables it to use no parameter, yet adapt to the multimodal landscape adeptly. It can be inferred that TLB-*l*DS opens a domain for transforming ordinary optimizers for niching through learning and stochastically improving its performance.

References

[1] Preuss, M.: Niching Prospects. In: Proc. of the Int'l Conf. on Bioinspired optimization Methods and their Applications, BIOMA, pp. 25–34. Jozef Stefan Institute, Slovenia (2006)
[2] Bäck, T.: Evolutionary Algorithms in Theory and Practice. Oxford University Press, New York (1996)
[3] Mahfoud, S.: Niching Methods for Genetic Algorithms. Ph.D dissertation, University of Illinois at Urbana Champaign (1995)
[4] Shir, O.M.: Niching in Derandomized Evolution Strategies and its Applications in Quantum Control, A Journey from Organic Diversity to Conceptual Quantum Designs, Ph.D thesis, Universiteit Leiden, ISBN: 987-90-6464-256-2, Printed in Netherlands
[5] Rao, R.V., Savsani, V.J., Vakharia, D.P.: Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. Computer-Aided Design 43, 303–315 (2011)
[6] Qu, B.Y., Suganthan, P.N.: Differential Evolution with Neighbourhood Mutation for Multimodal Optimization. IEEE Transactions on Evolutionary Computation (2010)
[7] Wright, S.: The roles of mutation, inbreeding, crossbreeding, and selection in evolution. In: Proceedings of the Sixth International Congress on Genetics, pp. 355–366 (1932)
[8] Mitchell, M.: Introduction to Genetic Algorithms (1996)
[9] Hardin, G.: The Competitive Exclusion Principle. Science 131, 1292–1297 (1960)
[10] de Jong, K.A.: An Analysis of the Beahaviour of a Class of Genetic Adaptive Systems, Ph.D. dissertation, University of Michigan, Ann Arbor (1975)
[11] Holland, J.H.: Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor (1975)
[12] Goldberg, D.E., Richardson, J.: Genetic Algorithms with sharing for multimodal function optimization. In: Proceedings of the Second International Conference on Genetic Algorithms, pp. 41–49 (1987)
[13] Pètrowski, A.: A clearing procedure as a niching method for genetic algorithms. In: Proc. of 3rd IEEE Congress on Evolutionary Computation, pp. 798–809 (1996)
[14] Pètrowski, A.: An efficient hierarchical clustering technique for speciation, Institute National des Telecommunications, Evry, France, Tech. Rep. (1997)
[15] Das, S., Maity, S., Qu, B.-Y., Suganthan, P. N.: Real-parameter evolutionary multimodal optimization — A survey of the state-of-the-art. Swarm and Evolutionary Computation 1(2), 71–88 (2011)
[16] Shir, O.M., Emmerich, M., Bäck, T.: Adaptive niche radii and niche shapes approaches for niching with CMA-ES. In: Evolutionary Computation, vol. 18, pp. 97–126 (2010)
[17] Qu, B.Y., Suganthan, P.N.: Differential Evolution With Neighborhood Mutation for Multimodal Optimization. IEEE Trans. on Evo. Comp. 16(5), 601–614 (2012)
[18] Thomsen, R.: Multimodal optimization using Crowding-based Differential Evolution. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 1382–1389 (2004)
[19] Li, X.: Efficient Differential Evolution using speciation for multimodal function optimization. In: Proceedings of the Conference on Genetic and Evolutionary Computation, Washington, D.C., USA, pp. 873–880 (2005)
[20] Li, X.: Niching without Niching parameters: particle swarm optimization using ring topology. IEEE Transactions on Evolutionary Computation 14 (February 2010)

Efficient Dynamic Routing on Large Road Networks

Mohanasuram Geetha[1], G.M. Kadhar Nawaz Gulammohien[1], and S. Saravanan[2]

[1] Sona College of Technology, Salem-5
geethasaravananjaswa@gmail.com
nawazse@rediffmail.com
[2] Jayalakshmi Institute of Technology, Thoppur
kprsaravana@gmail.com

Abstract. Route selection plays a very important role in road networks. It is a major problem for city travelers. This paper proposes a hierarchical community mining system which helps to model the road network in static and introduce a dynamic technique for fast route planning in large road networks. The foundation of our dynamic method is new approach that generalizes and combines fuzzy logic for local pheromone updating of an ant colony system in detection of optimum multi parameter direction between two desired points, origin to destination. The hierarchical community based routing algorithm significantly reduces the search space. We then propose a new Hierarchical community -Fuzzy Ant colony system supports dynamic efficient route computation on large road networks.

Keywords: hierarchical community structure, dynamic parameters, fuzzy logic, ant colony system.

1 Introduction

Computing fastest routes in road networks is one of the showpieces of real-world applications of algorithmic. The most classical shortest path algorithm Dijikstra's [1] .But for large road networks this would be far too slow. There have been many speed-up techniques for route planning [2] The most successful methods are static i.e., they assume that the network –including its edge weights-does not change. However, the computational effort is still quite high as the network size becomes large and real world network routing plan change all the time.

In this paper, we address two such methods: hierarchical approach and dynamic scenarios.

Hierarchical Approach. The computational effort is complex as the network size becomes large, making it unsuitable for real time routing[3]. At one extreme, we could precompute and store all pairs of shortest paths in a distance table. However, this would require a huge amount of storage space, which would exceed memory limit when large road maps are considered. A better approach would be to precompute and store some helpful hints [4] , Hierarchical approach have been proposed to seize some important vertices and arcs in road networks by using the community mining algorithm.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 169–180, 2012.

Dynamic Scenarios. The objective is to find a route with the least cost, based on the costs calculated for different possible directions. During the past years, many researchers have proposed methods for optimum route selection bases on some important parameters and for static scenarios only. However many drivers are now becoming increasingly concerned with fuel costs, waste of time in traffic congestions, and pollutant emissions.

In addition here, most users nowadays not only need routes with the shortest distance, but also require routes which can satisfy their other desires. Such users mostly need safe, low-traffic, and scenic routes with fewest numbers of junctions to avoid traffic lights.

The proposed system uses a combination of fuzzy logic and ant colony system [5] in order to find an optimum multiparameter route between a source and destination. The optimum routes that attempts to satisfy all desired parameters of a user. Individual edge weight updates, based on Traffic signal, Quality, Road condition, Tollgate charge, road restrictions.

2 Hierarchical Community – Fuzzy Ant Based Routing

Development of Hierarchical community fuzzy-ant based dynamic routing algorithm for large road networks architecture presented in Fig. 1. First we draw or import a graph from excel. Second we partition the graph into sub graph, next we check the source and destination vertices are same community or different, based on we use the algorithm. we describe the details of mechanism in this section.

2.1 Hierarchical Community Algorithm

The hierarchical community algorithm has two phases. Distance based community phase and high level community phase presented in Fig. 2.

The Partitioning phase used to split a given network $G = (V, E, W)$ into sub graph $G_u^l(V_u^l, E_u^l, W_u^l)$ based on Neighbourhood Random Walk Distance definitions and high level community phase used for constructing G^P from the sub graph G_u^l. Community phase based on user's region value. We have to calculate total weight and check it is less than region value, if it is less mean add G_u^l group otherwise add G_v^l group.

In high level community phase, we select the border node based on color vertex. Mostly same colored vertex forms a one group. For any node $i \in V_u^l$, if there exists a node , $j \in ADJ(i)$ i *colored vertex* $\neq$ j *colored vertex*,then i is named border node of G_u^l and j is named border node of G_v^l.The inter community edges denoted by solid line. Add the so-called community edges between every pair of border nodes of each sub graph, this one denoted by dashed line. So these edges construct the high level community graph.

2.1.1 Hierarchical Graph Model

A wide variety of methods have been developed for detecting hierarchical community in networks [6] ,here we proposed distance based community detection algorithm. The basic idea of community detection algorithm involves a number of definitions [7] A network can be modeled as a graph $G = (V, E, W)$,where V is the set of nodes and E is the set of links and w is the set of edge weights. The Graph G is partitioned into p communities at level l, with each community corresponding to a sub graph $G_u^l(V_u^l, E_u^l, W_u^l) \in G$.

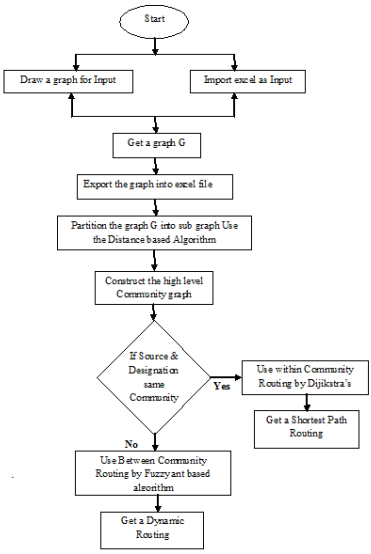


Fig. 1. Flowchart of Hierarchical-fuzzy ant based routing

A Partition of $P = \{G_1^l, G_2^l, \ldots, G_p^l\}$ of G. Two sub graphs G_u^l, G_v^l said to be adjacent if G_u^l, G_v^l is an edge of G. The given partition $P = \{G_1^l, G_1^l, \ldots, G_p^l\}$ of G, edges that link adjacent sub graphs G_u^l and G_v^l are called the intercommunity edge set, the set is denoted by

$$\mathrm{INTERCOM}(G_u^l, G_v^l) = \left\{ \langle i,j \rangle \;\middle|\; \begin{array}{l} (\langle i,j \rangle \in E) \wedge \left(i \xrightarrow{f_c(i,j)} j \text{ in } G \right) \\ \wedge \left(i \in \mathrm{BORDER}(G_u^l) \right) \\ \wedge \left(j \in \mathrm{BORDER}(G_v^l) \right) \end{array} \right\} \quad (1)$$

The intercommunity edges can be forming the bottlenecks between sub graphs. The cost function $f_c(i,f)$ gives the shortest path cost from node i to j .

Community Graph

A subgraph called as a community, the community edge set defined by

$$COMU\left(G_u^l\right)=\left\{\langle i,j\rangle \middle| \begin{array}{l}\left(\langle i,j\rangle \in BORDER\left(G_u^l\right)\right)\wedge \\ \left(i \xrightarrow{\xi_c(i,j)} j \text{ in } G\right)\wedge(i\neq j)\end{array}\right\} \quad (2)$$

High Level Community Graph

Add so-called community edges between every pair of border nodes. Finally link up adjacent sub graphs through intercommunity edges.

Input: Graph G with edge weight

Output: Number of sub graph

Partitioning phase:

1. Get a region value as user input
2. Initiate with starting node
3. For each vertex
4. Calculate the total edge weight
   +=connected edge weight
5. If total edge weight <= user's region value
6. Add into one group
7. Else other group
   End for

High level community phase:

1. Find the border node based on colored vertex
2. Add the so-called community edge
   between every pair of border nodes.
3. With in community edges denoted by dashed line
4. Between community edges denoted by solid line.

Fig. 2. Distance based community formation

2.1.2 Community Detection Phase

A community within a network is a group of vertices densely connected to each other but less connected to the vertices outside. Vertices tend to organize themselves in groups (called communities or clusters) such that the intersections that locate close in small regions are more likely to form a community. The network is then decomposed, with adjacent sub networks being loosely connected by the intergroup edges. In this approach, each sub networks forms an isolated part and different parts are connected through boundary or border nodes. All the shortest paths between different communities should go along one of these few edges. In this paper, we propose a distance estimation mechanism [8]. The optimality of the algorithm is guaranteed by the following definitions.

2.1.2.1. Structural Closeness Measure. In a large graph G, some vertices are close to each other while some other vertices are far apart based on connectivity. If there are multiple paths connecting two vertices vi and vj, then they are close. On the other hand, if there are very few or no paths between vi and vj, then they are far apart. In this paper, we use neighbourhood random walk distances to measure vertex closeness.

Definition 1. [Neighbourhood Random Walk Distance] Let P be the N × N transition probability matrix of a graph G. Given l as the length that a random walk can go, c ∈ (0, 1) as the restart probability, the neighbourhood random walk distance d(vi, v j) from vi to v j is defined as

$$d(vi, v\,j) = \sum_{\substack{\tau vi \rightsquigarrow v\,j \\ length(\tau) \le l}} p(\tau)c(1-c)^{length(\tau)} \tag{3}$$

where τ is a path from vi to v j whose length is length(τ) with transition probability p(τ).

The matrix form of the neighbourhood random walk distance is

$$R^l = \sum_{\gamma=1}^{l} c(1-c)\gamma^{P\gamma} \tag{4}$$

Here, P is the transition probability matrix for graph G, and R is the neighbourhood random walk distance matrix.Then the structural closeness between two vertices vi and v j is

$$d_s(vi, vj) = R^l(i,j) \tag{5}$$

2.1.3 High-Level Community Phase

In high level community phase, we select the border node based on colour vertex. Mostly same coloured vertex forms a one group. The inter community edges denoted by solid line. Add the so-called community edges between every pair of border nodes of each sub graph, this one denoted by dashed line. So these edges construct the high level community graph.

2.2 Fuzzy- Ant Based Algorithm

The propose fuzzy-ant based routing using a fuzzy logic technique to solve the network routing problem [9] , which allows multiple constraints to be considered in a simple and intuitive way.

2.2.1 Overview of Ant Based Routing

Ant based is a meta-heuristic, meaning that it's a general framework that can be used to create a specific algorithm to solve a specific graph path problem. Although ant based was proposed in a 1991 doctoral thesis by M. Dorigo, the first detailed description of the algorithm is generally attributed to a 1996 follow-up paper by M. Dorigo, V. Maniezzo and A. Colorni. Ant based routing algorithms derive from recent

understandings of basic principles underlying the operation of biological swarms, often containing thousands of elements, routinely perform extraordinarily complex tasks of global optimization and resource allocation using only local information. These properties make ant colony very attractive for network routing [10] ,routing in telecommunication network [11].

2.2.2 Fuzzy Logic for Fuzzy Ant Based Algorithm

In this paper, fuzzy ant based algorithm is constructed with the inspiration of a large road network observed in ant colonies combined with the capabilities of the fuzzy logic technique. This algorithm first determines the crisp path ratings for all eligible paths between the source and destination nodes from the viewpoint of fuzzy inference. The path with the highest rating is then chosen to route the shortest path. The fuzzy inputs are chosen as the 'Traffic ',road condition', 'quality', and 'tollgates' of the direction which ant k has selected. By considering computing complexities, only two input fuzzy sets, "low" , "high" are defined for each input. The fuzzy rule base is shown in Table 1.

There are 16 rules defined for this fuzzy system. The membership functions for the fuzzy input are shown in Fig.3. The universes of discourse for the fuzzy variables are all normalized between (0,1). The membership function for the fuzzy sets of inputs are chosen to be trapezoidal-shaped because this type of membership function has good features such as easiness in computation.

There is also a fuzzy set for output variable as shown in Fig.4. All of the membership functions for the fuzzy sets of output are chosen to be a triangular for its easiness in computation.

Table 1. Fuzzy rule base for fuzzy ant base algorithm

Rule no.	IF				THEN
	Quality	Road condition	Traffic	Tollgate	local pheromone update
1	High	High	Low	Low	Very strong
2	High	High	Low	High	Strong
3	High	High	High	Low	Strong
4	High	Low	Low	Low	Strong
5	High	Low	Low	High	Strong
6	Low	High	High	Low	Strong
7	Low	High	Low	High	Strong
8	Low	High	Low	Low	Strong
9	Low	High	High	High	Weak
10	High	Low	High	High	Weak
11	High	High	High	High	Weak
12	High	Low	High	Low	Weak
13	Low	Low	Low	Low	Weak
14	Low	Low	Low	High	Weak
15	Low	Low	High	Low	Weak
16	Low	Low	High	High	Very Weak

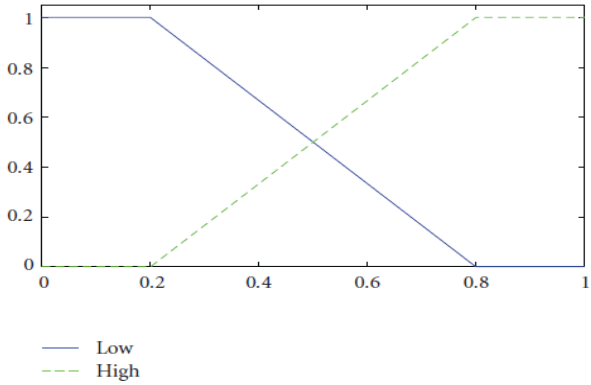


Fig. 3. Membership function for Quality, Road condition, Traffic and Tollgate

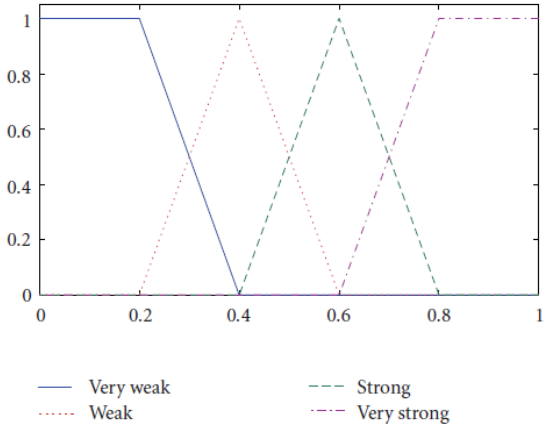


Fig. 4. Membership function for output(local pheromone updating)

The output variable has four membership functions titled as “Very strong”, “Very weak” , “Strong”, “Weak”. The fuzzy operator used for the AND method in “if-then rules” such a s “IF a is a1 and b is b1 then c is c1”. The defuzzification is the process of conversion of fuzzy output set into a single number.

2.2.3 Fuzzy Ant Based Algorithm

The fuzzy ant based routing algorithm shown in Fig.5. It as follows.

Initialize. It is consisted of initial of the algorithm parameters such as number of ants, evaporation coefficient,

Locate ants. Ants are located on the start point in this stage. Here we check the ant is blocked or not. Since each ant can traverse each junction once in each iteration.

Construct Probability. The probability of each possible direct route is calculated based on its total number of edges for each active ant. The probability of displacing from junction i to junction j for ant k is as

$$P_{ij}^{k}=\begin{cases} \dfrac{\tau_{ij}^{\alpha}\Pi_{leparameters}\ \xi_{ij}^{-\alpha_l}}{\Sigma_{h\notin tabu_k}\tau_{ih}\ \Pi_{leparameters}\ \xi_{ih}^{-\alpha_l}} & j\notin tabu_k \\ 0 & otherwise \end{cases} \tag{6}$$

Where τ_{ij} is the direct route pheromone intensity from junction i to j. parameters α controls the importance of τ_{ij}.The $tabu_k$ list is the set of direct blocked routes. Parameter set is a collection of most important parameters for drivers taking journeys in metropolises. For more simplicity, the 'Traffic ',’road condition', 'quality', and 'tollgates' parameters are considered in this set. Cost function of each parameter l is adjustable by α. Traffic and tollgate high, the total cost increases and consequently it decrease the probability of selecting that route. Road condition and quality low, it decreases the total cost and increase the probability of selecting that route.

Select route. Active ant selects the route with highest probability. Otherwise it selects next junction through probabilities.

Update Tabu List. In this step, the route which ant k has been chosen is added to the tabu list in order not to be selected again.

Update Pheromone. The Ant pheromone system is consisted of two main rules: first is applied local pheromone update rule and global pheromone update rule applied after all ants have finished constructing a solution. The pheromone amount of the route junction i and j is updated for ant k as

$$\tau_{ij}^{new}=\tau_{ij}^{old}+(10X\Delta\tau) \tag{7}$$

Where $\Delta\tau$ is the amount of local pheromone updating. The value of $\Delta\tau$ is the output of a fuzzy logic system. By considering the fuzzy logic in section 3.2.2, and Mamdani's implication as our approach. The total last step of each completed loop is global pheromone updating defined as

$$\tau_{ij}^{new}=\rho\tau_{ij}^{old} \tag{8}$$

Where0 < ρ < 1 is the evaporation coefficient and is usually set to 0.9.

Select Best Direction. After m loops, direction with the lowest cost from origin to destination is recommended by the system.

3 Experimental Evaluation

To verify the validity of our hierarchical routing algorithm, consider the road network with up to 25 vertices and 34 edges. All algorithms were developed in C#.net framework and conducted on Intel(R) Core(TM) i3-2120 CPU @ 3.30GHz. The system ran Microsoft Windows.

```
Procedure Fuzzy ant
  Initialize
  For each loop
    Locate ants
    For each iteration
      For each
        If ant is active
          Construct probability
          Select route
          Update tabu list
        End
      Next ant
    Next iteration
    Update pheromone
  Next loop
  Select best direction
End Fuzzy ant
```

Fig. 5. Fuzzy ant based algorithm

3.1 Preprocessing

The community detection algorithm used with a variation of edge weight in the number of vertices divided by the distance of the road such close intersections are more likely to form the same community. The proposed system is applied on road network. The network consisted of 25 vertices and 34 edges. The Fig.6. Shows the primary input of creating the graph and exporting the excel file with existing source , target and weight. It shows the secondary input from importing the excel file using exported file to avoid recreation of the graph. In Fig.7. Shows the community graph with various colors vertices. Fig.8. shows the high-level community graph with 6 communities and 15 border nodes. The hierarchical approach used to limit the storage space and computationalcost. Get the additional values about the edges from user for all selected border nodes. The input like low or high for the user's parameter Traffic, Tollgate, Quality and Road condition. For example shown in the Table 2. Based on this table, the ants select very strong.

Table 2. Local pheromone updating

No of border node edges	Border edge	User preference				dynamic result
		Quality	Road cond	Traffic	Toll gate	
1.	7-14	High	High	Low	Low	Very strong
2.	7-13	Low	High	Low	Low	Strong

3.2 Dynamic Shortest Path

Select the source and destination pair randomly and select the user's parameter. Here we select source is one community and destination is in other community. Get "Traffic=low", "Tollgate=low", "Quality=high", "Road condition=high" and the

result is “very strong”. The fuzzy ant algorithm updates the local pheromone with the result of “very strong”. Here we select 7/24 vertices are source and destination using the Fuzzy-Ant based algorithm. The Table 2 shows the possible way of routing. Here we have the number of edges six and five , the user parameter output is “very strong” and “strong”. Based on it efficient routing highlighted in this graph. It shows in the Fig.9.

Table 3. Efficient Dynamic shortest path routing

Route selection system	s/t pair	Possible Routing	Number of edges with dynamic result
Fuzzy-Ant	7/24	7-14-16-23-22-24 7-13-21-19-24	6 with Very strong 5 with Strong

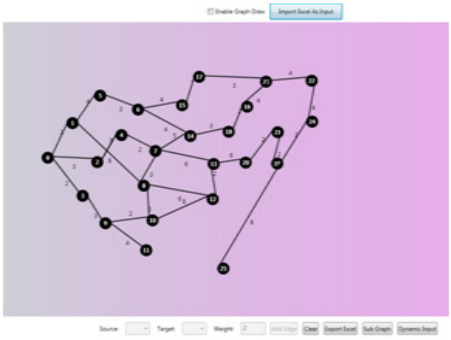


Fig. 6. Originalgraph

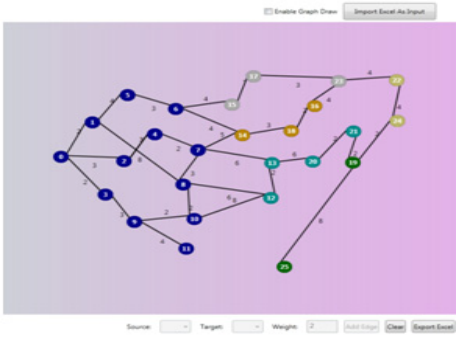


Fig. 7. Communityformation

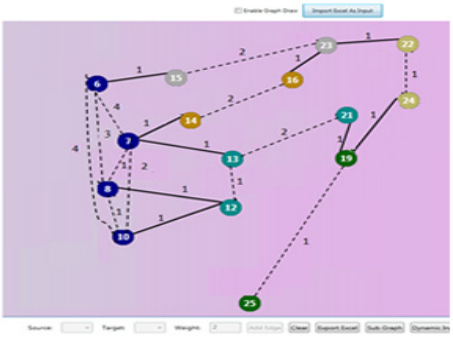


Fig. 8. High level community

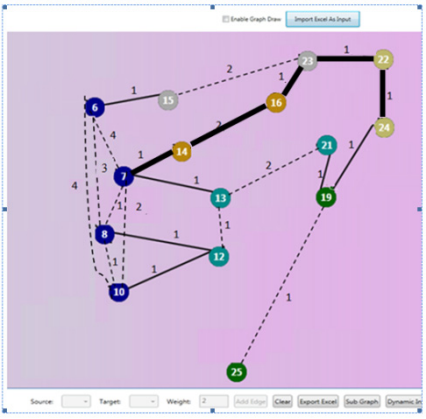


Fig. 9. Dynamic shortest path routing

4 Conclusion

The problem of route selection in large road networks has been widely investigated. Several efficient algorithms have been proposed. The main drawback of existing system introduced static scenarios and dynamic scenarios with high storage, computational cost. The proposed system in this paper developed a hierarchical approach that supports efficient route computation on large road networks. Instead of using hierarchical community mining algorithm to retrieve a hierarchical graph model, which could compute optimal dynamic routes for same community nodes pair and different community nodes pair on large road networks, based on user parameters of Traffic, Tollgate, Quality, Road condition by using fuzzy logic and ant colony system. Fuzzy logic is considered as a management mechanism for the proposed ant colony system local pheromone updating. The experimental results demonstrate that our algorithm used lots of real–time applications for emergency services, tourist guides, and generally for anyone who wants to have a low-cost, safe and comfortable journey in large road networks.

References

1. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numer. Math. 1(1), 269–271 (1959)
2. Sanders, P., Schultes, D.: Engineering Fast Route Planning Algorithms. In: Demetrescu, C. (ed.) WEA 2007. LNCS, vol. 4525, pp. 23–36. Springer, Heidelberg (2007)
3. Song, Q., Wang, X.: Efficient Routing on Large Road Networks using Hierarchical communities. IEEE Transactions on Intelligent Transportation Systems 12(1) (March 2011)
4. Chen, Y.Y., Bell, M.G.H., Bogenberger, K.: Reliable pretrip multi-path planning and dynamic adaptation for a centralized road navigation system. IEEE Trans. Intell. Transp. Syst. 8(1), 14–20 (2007)
5. Salehinejad, H., Talebi, S.: Dynamic fuzzy logic-ant colony system-based route selection system. Applied Computational Intelligent and Soft Computing 2010, Article ID 428270, 13 pages (2010)
6. Fortunato, S.: Community detection in graphs. Phys. Rep. 486(3-5), 75–174 (2010)
7. Essam, J.W., Fisher, M.E.: Some basic definitions in graph theory. Rev. Mod. Phys. 42(2), 271–288 (1970)
8. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech. 2008(10), P10 008 (2008)
9. Mirabedini, S.J., Teshnehlab, M., Shenasa, M.H., Movaghar, A., Rahmani, A.M.: AFAR: adaptive fuzzy ant-based routing for communication networks. Journal of Zhejiang University Science A (2008) ISSN 1673-565X (print): ISSN 1862-1775
10. Sim, K.M., Sun, W.H.: Multiple Ant-Colony Optimization for Network routing. In: Proc. Ist Int. Symp. on Cyber Worlds, pp. 277–281 (2002)
11. Akon, M.M., Goswami, D., Jyoti, S.A.: Routing in Telecommunication network with controlled Ant Population. In: Proc. Ist IEEE Consumer Communication and Netorking Conf., pp. 665–667 (2004)

Performance of Informative Differential Evolution Algorithm with Self Adaptive Re-clustering Technique on the Problems of Electromagnetism –The Linear Array Antenna Synthesis

Dipankar Maity, Udit Halder, and Sheli Sinha Chaudhuri

Dept. of Electronics and Telecomm. Engineering, Jadavpur University, Kolkata, India
{dipankarmaity1991,udithalder99}@gmail.com,
sschaudhuri@etce.jdvu.ac.in

Abstract. In this paper we are going to investigate the properties of Informative Differential Evolution with Self Adaptive Re-clustering technique (IDE_SR) algorithm on the problems of electromagnetic domain. Problems of electromagnetism domain generally exhibit multimodal characteristic as well as the high dimensionality and non-smooth attributes. IDE_SR is a modified Differential Evolution (DE) to overcome the problems of multimodality and complexity of the problem under consideration. In this paper we describe the salient features of IDE_SR and compare the result with other state-of-art algorithms.

Keywords: self Adaptive, Re-clustering technique, multimodality.

1 Introduction

Function optimization by means of Evolutionary Algorithm (EA) has attracted a great deal of research in last few decades or so. Researchers have put considerable effort to develop efficient algorithms to optimize a function that cannot be done using classical techniques such as gradient decent, nonlinear programming methods. In the last few decades of twentieth century, researchers were keen to develop optimization techniques inspired by natural foraging strategies of ants, bees, schooling of fish, or even by the Darwinian principle. Particle Swarm Optimization (PSO) [1], Bacterial Foraging Optimization (BFO) [2], Ant Colony Optimization (ACO) [3], Artificial Bees Colony Optimization (ABC) [4], Artificial Immune Algorithm (AIA) [5] and Genetic Algorithm (GA) [6] are the examples of the above mentioned type bio-inspired algorithm. Research has been done to study the functionalities of neurons in brain, and consequently they developed Artificial Neural Network (ANN) [7] which was later used for function optimization. Now researchers have incorporated local search techniques with Genetic Algorithm and formed a new class of algorithms, known as Memetic Algorithm (MA) [8]. Apart from the development of all these bio-inspired algorithms, Price and Storn proposed Differential Evolution [9] which was

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 181–188, 2012.

not inspired by nature. Despite of the simplicity of Differential Evolution (DE), it is one of the most promising algorithms proposed in the last part of twentieth century. In many applications such as Pattern Recognition, Communication, Mechanical engineering, Real world optimization [10, 11] the effectiveness and efficiency of DE and other algorithms have been successfully demonstrated. Despite of being a good optimization technique, like other existing algorithms DE also suffers from some major problems e.g. premature convergence, trapping into local optimum etc. However, there exists a lot of research in literature which provides a means to alleviate these problems.

Optimization problem in electromagnetic domains generally include large number of design parameters, and sometimes there may exist constrains among these parameters in the search range. These problems are generally very intricate in nature and thus call for a very sophisticated algorithm which can overcome the trapping into local optimum as well as must possess a satisfactory convergence power. Use of EAs to solve electromagnetic problems is not a new idea, many works can be found in literature that took the help of EAs. EAs such as Particle swarm Optimization [12, 13], Invasive weed Optimization [14] and Genetic Algorithms [15] used extensively in last few decades or so.

In this paper we consider the linear array antenna synthesis which is a standard benchmark problem in the domain of electromagnetic design. The main purpose of this paper is to highlights the attributes and new features of IDE_SR [16, 17] algorithm for electromagnetic problems. The performances of other state-of-art algorithms on the same problem have been studied and compared with IDE_SR.

2 Informative Differential Evolution with Self Adaptive Re-clustering Technique (IDE_SR)

IDE_SR [16, 17] is a modified version of DE developed to overcome the problems associated with DE. IDE_SR uses a multi-population based strategy along with a local search technique to find finer solutions. The algorithm starts with initializing the population in the search range as uniformly as possible. The total population is divided into several subpopulations using K-means clustering algorithm. Using the multi-population strategy alleviates the problem of trapping into local optimum. Though one may argue that these subpopulations may prematurely converge to any optimum but still the fact is they altogether will not converge to a single optimum rather different optima will be explored by these subpopulations. We apply DE/best/1 strategy to each of these subpopulations separately. DE/best/1 has better convergence power than other schemes. During evaluations these subpopulation cannot communicate among them. After a certain number of iterations, we allow the subpopulations to exchange the information gained by them. Thus intermittently allowing them to rearrange and re-cluster accelerates the process of optimization while avoiding the local optima. The information exchange is achieved by re-clustering the whole population again in a definite number of subpopulation. The number of subpopulation is not fixed throughout the optimization process rather it changes self adaptively as per the requirement. A fitness feedback scheme was used to change the subpopulation number, i.e. if the algorithm performs satisfactorily then

the cluster number was reduced and if the algorithm is performing not well then to increase the exploration power the subpopulation number is increased. The local search technique is invoked when the algorithm is suspected for trapping into a local optimum. The local search algorithm generates some new members in that subpopulation and if the generated members find a better solution, for that subpopulation the algorithm keeps the best solutions among the members of that subpopulation and the generated ones. Otherwise the subpopulation is deleted if it does not contain the global best solution. For a detailed description of IDE_SR algorithm, the readers are directed to see the references [16, 17]; here we provide some key features of it.

2.1 Initialization

The algorithm randomly initializes the members covering the whole search space as uniformly as possible. The population is divided using K-means algorithm into a number of subpopulations.

2.2 Subpopulation Interaction

The subpopulations are periodically merged into a single population and the re-clustering process is done. After a certain number of iterations –known as refreshing gap g_r –the algorithm is re-clustered into a specific number of subpopulation. The number of subpopulation is determined self adaptively by the performance of the algorithm in its previous refreshing gap. If the performance is satisfactory then the subpopulation number is decreased otherwise it is increased up to a certain limit on both side; i.e. cluster number is varied between *subpopulation_max* (or *cluster_max*), to *subpopulation_min* (or *subpopulation_min*). The subpopulation number is changed as follows:

$$Subpopulation\_no = \min(subpopulation\_\max,\ prev\_subpopulation\_no + 1).$$

$$Subpopulation\_no = \max(subpopulation\_\min,\ prev\_subpopulation\_no - 1).$$

2.3 Local Search Technique

If the radius of the subpopulation becomes smaller than a threshold value then the local search technique is applied to that subpopulation. In this phase, as stated above, some new particles are generated randomly in the vicinity of some of the members of the subpopulation. For the detailed description –how radius are calculated and how the particles are chose around which we generate new particles –the readers are directed to see the references of IDE_SR given above. The number of new members generated around a member of the subpopulation is determined by the fitness of the parent member. The radius, within which the new members are generated, is also dependant of the fitness of the parent member.

2.4 Pseudo Code of IDE_SR

//Step 1: Initialization

1. Initialize the parameters *NP, F, Cr, subpopulation_max, subpopulation_min, exploiters_max, exploiters_min, r_ini, p_ref.*
2. $k \leftarrow subpopulation\_\max$
3. Initialize a population of NP in the search space randomly.
4. Divide the population into *k* number of subpopulations.
5. $Subpopulation\_no \leftarrow k$, $t \leftarrow 1$.

//Step 2: Subpopulation interaction

6. **While** stopping criterion is not satisfied **do**
7. **for** i=1 to *Subpopulation_no* do
8. Calculate radius R_i by (7)
9. **if** $R_i > R_{conv}$ **do**
10. Improve cluster *i* using "DE/best/1/bin" scheme.
11. **else**
12. Call the local search algorithm
13. Delete subpopulation *I* if it does not contain the global best particle
14. **end if**
15. **end for**

//Step3 : Performance Evaluation and Redistribution

16. $percent \leftarrow percent + 100 * |global\_best - prev\_global\_best| / prev\_global\_best,$
17. **if** *modulo(t,g_r) = 0* **do**
18. **if** $percent \geq p\_ref$ **do**
19. $Subpopulation\_no \leftarrow \max(subpopulation\_\min,\ prev\_subpopulation\_no - 1).$
20. **else**
21. $Subpopulation\_no \leftarrow \min(subpopulation\_\max,\ prev\_subpopulation\_no + 1).$
22. **end if**
23. Re-cluster the whole population into *Subpopulation_no* of subpopulations.
24. **end if**
25. **end while**

3 The Linear Array Antenna Synthesis

The consideration focuses on the optimization of array antennas to achieve the desired radiation pattern given by user defined function. For an array antenna with *N* elements, separated by a distance *d*, the normalized array factor is given by [17, 18],

$$AF(\theta) = \frac{1}{AF_{\max}} \sum_{i=1}^{N} I_i e^{j2\pi i\left(\frac{d}{\lambda}\right)\sin(\theta)}, \tag{1}$$

where I_i is the amplitude of current excitation, θ is the angle from the normal to the array axis, $AF_{\max}$ is the maximum magnitude of the array factor, and *d* is assumed to be $\lambda/2$, where λ is the wavelength.

The requirement is to get a desired pattern of $AF(\theta)$ with the variation of θ in the range $[-\pi/2,\ \pi/2]$. The objective function is same as that of [18] to optimize the desired radiation pattern of a 40 element array antenna. The main purpose is to obtain side lobe level less than the tapered side lobe mask that decreases linearly from -40dB to -50dB. The beam width of the array pattern was kept to be fixed as 11^0 and the number of sampling points was also taken as 359. We also used the "Don't exceed criterion" to formulate the objective function as stated in [18]. In [18] Shaya Karimkashi *et al*. used IWO and substantiated the better performance of IWO over different PSO schemes. The performance is measured by the fact that how fast the algorithms converges to meet the requirement. Here we used IDE_SR in place of IWO and investigated the performance of IDE_SR over IWO and it was observed that IDE_SR performs much better than IWO. The experimental results section contains a detailed comparison among these algorithms.

4 Experimental Results

The population size and number of iteration was same for all the algorithms. The population size was assumed to be 50. The other parameters for IDE_SR were selected as *subpopulation_max=5, subpopulation_min=2, exploiters_max=5, exploiters_min=2, g_r=10, F=0.5(rand+1), Cr=0.9, p_ref=2.5, r_ini=5e-03.* As stated in [], the maximum number of plant population was fixed 10 and the number of seeds increases linearly from 0 to 5, *n* is taken to be 3 and the initial and final standard deviation are set as 0.015 and 0.00005 for IWO. The cognitive attraction coefficient (c_1) and the social attraction coefficient (c_2) were taken to be 0.2 and the inertial weight factor is varied linearly from 0.9 to 0.2 as described in [18]. The results reported are the average of 50 independent runs.

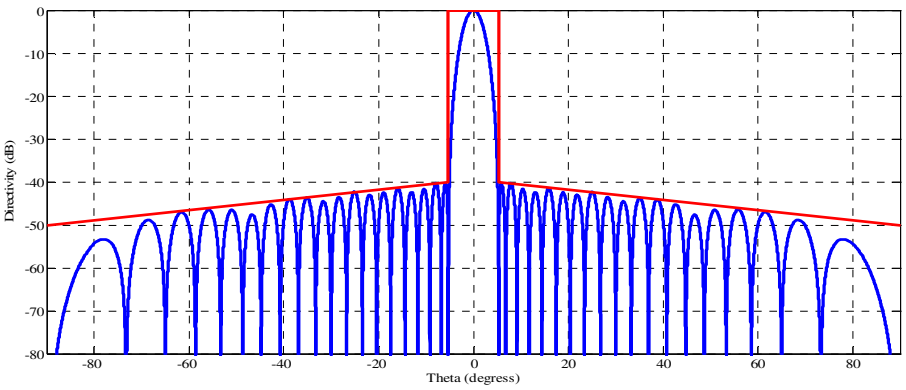


Fig. 1(a)

Fig. 1. Amplitude only synthesis of a 40 element array – (a) radiation pattern obtained by using IDE_SR, (b) the convergence plot of the IDE_SR algorithm

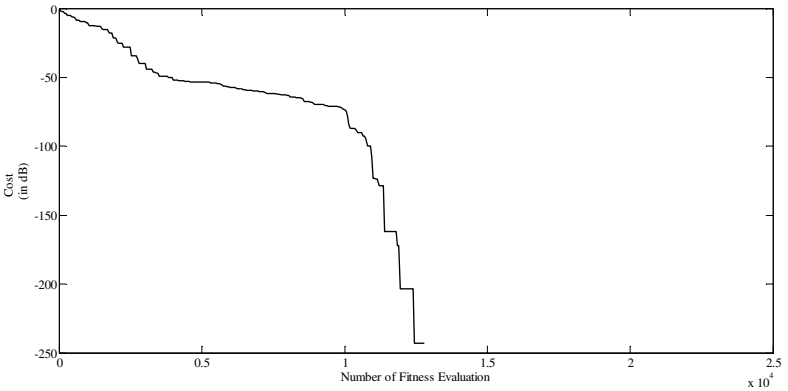


Fig. 1(b)

Fig. 1. (*continued*)

From the array pattern we obtained it is clear that there is no part of the array pattern above the tapered side lobe mask i.e. IDE_SR successfully achieved the desired pattern. The second plot (fig 1(b)) shows how fast IDE_SR converges to obtain the desired array pattern. With the results given in [18], it is clearly stated there that PSO undergoes the problem of trapping into local optima frequently. Different PSO schemes illustrated the dramatically change in the performance of the PSO. Different boundary conditions and different settings of maximum velocity result in different type of convergence. The PSO-RBC technique where V_{max} was set as low as 0.1 showed the best performance. Though it was shown there that IWO is the best choice for that optimization problem but here we successfully prove that the performance of IDE_SR is much better than IWO. We provide a table below to make a comparison among the performances of these algorithms.

Table 1. Comparison of average number of fitness evaluations required per Successful run in the IDE_SR, the PSO and the IWO algorithms for the linear 40-element array optimization

Algorithms	Average number of fitness evaluation required per successful run	Time required to complete a single run (in Sec)
PSO-ABC	-	521.67
PSO-DBC	162328	576.23
PSO-RBC(Vmax=1)	92485	545.24
PSO-RBC(Vmax=0.3)	62677	546.19
PSO-RBC(Vmax=0.1)	42329	545.98
PSO-IBC	52496	534.71
IWO –Restricted BC	14692	678.45
IWO –Invisible BC	18212	621.67
IDE_SR	**12754**	**495.24**

Table 1 successfully shows that IDE_SR performs better than the other algorithms as it has a better convergence rate which does not lead to any premature trapping into local optimum.

5 Conclusion

In this paper we explored the behavior of IDE_SR algorithm on the benchmark problem of electromagnetic. The self adaptive nature, the fitness feedback scheme and a novel local search technique has empowered IDE_SR in all aspects of convergence and population diversity. The simulation result successfully justifies the extraordinary performance of IDE_SR. Though it may seem at a first look that the algorithm contains some parameters but one thing must be noted that some of them are change self adaptively, such as the subpopulation number changed as per the performance of the algorithm in the last *g_r*. The number of generated exploiter members and their spatial distributions are determined by the functional value of their respective parent members. We took the population of only 50 individuals which incurs the general optimization strategy but still it gives better result.

References

1. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proceedings of IEEE International Conference on Neural Networks, Piscataway, NJ, pp. 1942–1948 (1995)
2. Passino, K.M.: Biomimicry of Bacterial Foraging for Distributed Optimization and Control. IEEE Control Systems Magazine 22, 52–67 (2002)
3. Dorigo, M., Stützle, T.: Ant Colony Optimization. MIT Press, Cambridge (2004)
4. Karaboga, D.: An Idea Based on Honey Bee Swarm for Numerical Optimization, technical REPORT-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department (2005)
5. Farmer, J.D., Packard, N., Perelson, A.: The Immune System, Adaptation and Machine Learning. Physica D 22, 187–204 (1986)
6. Holland, J.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975)
7. Aihara, A., Takabe, T., Typoda, M.: Chaotic neural networks. Phys. Lett. A 144, 333–340 (1990)
8. Chen, X.S., Ong, Y.S., Lim, M.H., Tan, K.C.: A Multi-Facet Survey on Memetic Computation. IEEE Transactions on Evolutionary Computation 15(5), 591–607 (2011)
9. Storn, R., Price, K.V.: Differential Evolution-A simple and efficient Heuristic for Global Optimization over continuous Spaces. Journal of Global Optimization 11, 341–359 (1997)
10. Ilonen, J., Kamarainen, J.K., Lampinen, J.: Differential Evolution Training Algorithm for Feed-Forward Neural Networks. Neural Processing Letters 7, 93–105 (2003)
11. Storn, R.: Differential evolution design of an IIR-filter. In: Proceedings of IEEE Int. Conference on Evolutionary Computation, ICEC 1996, pp. 268–273. IEEE Press, New York (1996)
12. Boeringer, D.W., Werner, D.H.: Particle swarm optimization versus genetic algorithms for phased array synthesis. IEEE Transactions on Antennas and Propagation 52(3), 771–778 (2004)

13. Khodier, M.M., Eberhart, R.C.: Linear Array Geometry Synthesis with Minimum Sidelobe Level and Null Control Using Particle Swarm Optimization. IEEE Transactions on Antennas and Propagation 53(8), 2674–2679 (2005)
14. Mallahzadeh, A.R., Eshaghi, S., Alipour, A.: Design of an E-shaped MIMO antenna using IWO algorithm for wireless application at 5.8 GHz. Progress In Electromagnetics Research 90, 187–203 (2009)
15. Jain, R., Mani, G.S.: Dynamic thinning of antenna array using genetic algorithm. Progress In Electromagnetics Research B 32, 1–20 (2011)
16. Maity, D., Halder, U., Dasgupta, P.: An Informative Differential Evolution with Self Adaptive Re-clustering Technique. In: Panigrahi, B.K., Suganthan, P.N., Das, S., Satapathy, S.C. (eds.) SEMCCO 2011, Part I. LNCS, vol. 7076, pp. 27–34. Springer, Heidelberg (2011)
17. Maity, D., Halder, U., Das, S., Vailakos, A.V.: An Informative Differential Evolution Algorithm With Self Adaptive Re-Clustering Technique for the Optimization of Phased Antenna Array. Progress In Electromagnetics Research B 40, 361–380 (2012)
18. Karimkashi, S., Kishk, A.A.: Invasive Weed Optimization and its Features in Electromagnetics. IEEE Transactions on Antennas and Propagation 58(4), 1269–1278 (2010)

Differential Evolution with a Relational Neighbourhood-Based Strategy for Numerical Optimization

Souvik Kundu, Digbalay Bose, and Subhodip Biswas

Dept. of Electronics and Communication Engineering,
Jadavpur University, Kolkata 700 032, India
{sk210892,digbose92,subho.opto.placis91}@gmail.com

Abstract. Differential Evolution is a competitive optimizer, with a simplified framework, for numerical optimization problems. Many research works have been done to enhance the performance of Differential Evolution by developing the evolutionary operators. One of the major challenges in DE is performing intelligent search based on population topology. To maintain the diversity in the population as well as to improve the convergence rate, we have introduced a mutation strategy based on relative mapping of the members in population topology. Also Gamma and Cauchy distribution have been adapted in the control parameter framework to include randomness and thorough search. The proposed DE framework is referred to as the Relational Neighbourhood Differential Evolution **(R*e*Nbd-DE)** and its performance is reported on the set of CEC2005 benchmark functions.

1 Introduction

Differential Evolution (DE), proposed by Storn and Price [1], is an effective Evolutionary Algorithm (EA) with a relatively simple framework. It has been modified to adapt its performance for handling constrained, multi-objective, dynamic, large scale and combinatorial optimization problems. DE is used in various scientific fields now-a-days for its rapidly increasing popularity. One of the major issues in DE algorithm is to determine the optimal value of the control parameters so that the major percentage of population can search the solution space without being trapped in order to find the global optimum. Many state-of-the-art techniques have been proposed to balance explorative and exploitative behavior throughout the population. But the mutation operators used have a tendency to be snared in unexpected regions.

Motivated by the specified drawbacks we have introduced a new mutation strategy i.e. relational neighbourhood guided strategy to maintain both the exploration and exploitation throughout the population. The crossover used in our algorithm is inspired from the mDE-*p*Bx algorithm [2]. Two probability distribution functions i.e. Gamma function and Cauchy functions [3] have been used to adapt the parameters values.

The remaining portions of the paper are organized as follows. Section 2 describes the basic DE algorithm which is followed by our proposed algorithm in Section- 3. Section 4 contains the experimental study on the numerical benchmarks used along with comparative study and Section 5 concludes the paper.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 189–197, 2012.

2 Differential Evolution Frameworks

The integral dynamics of DE has been outlined in this section as follows.

2.1 Initialization

The search process begins by initializing trial solutions in the real parameter space $\Re^D$ called genomes. It is a population based model with *NP* D-dimensional vectors. If the optimizer runs for G_{max} generations with G denoting the present generation, the i^{th} vector is represented as $X_{i,G}$= { $\boldsymbol{x}^1_{i,G}$, $\boldsymbol{x}^2_{i,G}$, ……., $\boldsymbol{x}^D_{i,G}$ }. The solution space is bounded by search constraints defined by maximum and minimum limits X_{min}= { $\boldsymbol{x}^1_{min}$, $\boldsymbol{x}^2_{min}$, $\boldsymbol{x}^D_{min}$ } and X_{max}= { $\boldsymbol{x}^1_{max}$, $\boldsymbol{x}^2_{max}$, ……., $\boldsymbol{x}^D_{max}$ }. The j^{th} parameter of the i^{th} vector is initialized as $\boldsymbol{x}^j_{i,0}= \boldsymbol{x}^j_{min} + \mathbf{r}.(\boldsymbol{x}^j_{max} - \boldsymbol{x}^j_{min})$, where **r** is an uniformly generated random number in the range [0, 1].

2.2 Mutation

The search process proceeds through the mutation strategy where the i^{th} vector in the present generation $X_{i,G}$ is mutated to form the donor vector $V_{i,G}$. Literature is abound in variety of mutation strategies but the most basic ones are discussed here.

$$\text{DE/current-best/1: } V_{i,G}= X_{i,G} + \boldsymbol{F}.(X_{best,G} - X_{i,G}) \tag{1a}$$

$$\text{DE/rand/1: } V_{i,G}= X_{r1,G} + \boldsymbol{F}.(X_{r2,G} - X_{r3,G}) \tag{1b}$$

$$\text{DE/rand/2: } V_{i,G}= X_{r1,G} + \boldsymbol{F}.(X_{r2,G} - X_{r3,G}) + \boldsymbol{F}.(X_{r4,G} - X_{r5,G}) \tag{1c}$$

$$\text{DE/current-to-rand/1: } V_{i,G}= X_{i,G} + \boldsymbol{F}.(X_{r1,G} - X_{r2,G}) \tag{1d}$$

We have used the basic DE/rand/1 strategy based in the ReNbd-DE owing to its superior performance on a wide variety of test benchmarks.

2.3 Crossover

DE uses the crossover strategy to create an offspring vector by combining components from both the parent $X_{i,G}$ and its donor vector $V_{i,G}$. Generally there are two methods of crossover used – binomial and exponential. Our proposed ReNbd-DE makes use of binomial crossover as elucidated below

$$\textit{If } \mathbf{r} \le Cr \textit{ or } j=\boldsymbol{j}_{rand} \quad \boldsymbol{u}^1_{i,G}= \mathbf{v}^1_{i,G} \tag{2a}$$

$$\textit{Otherwise} \quad \boldsymbol{u}^1_{i,G}= \mathbf{x}^1_{i,G} \tag{2b}$$

where **r** is a randomly generated number in the range [0, 1], *Cr* is crossover probability and $\boldsymbol{j}_{rand}$ is an index generated randomly in the range [1, D].

2.4 Selection

The last stage in the algorithmic framework is the usage a greedy selection technique that compares between the original parent vector and the newly generated offspring. It can be mathematically written as

$$\text{If} \quad f(V_{i,G}) \leq f(X_{i,G}) \qquad X_{i,G} = V_{i,G} \tag{3a}$$

$$\text{Otherwise} \qquad X_{i,G} = V_{i,G} \tag{3b}$$

Where $f(X_{i,G})$ is the function to be optimized.

Recently a need was felt to use parameter adaptation in DE framework since they ensure good performance based on statistical evaluations. Sugnathan and Das [13] *et. al* made a comprehensive survey of the state-of-the-art DE algorithms in literature.

3 R*e*Nbd-DE Algorithm

In this section R*e*Nbd-DE algorithm and its various steps are outlined here.

- Initially the population will be initialized according to their search domain for all the corresponding dimensions as discussed in the section 2.1.
- In the second stage we have introduced a new strategy, i.e. relational-neighbourhood mutation strategy for carrying out the search process. To demonstrate this concept let us consider a member X_i and X_{best} in the individual which provides the best value (provided $X_{best} \neq X_i$) in generation G. Now the *t*-value is calculated as, i.e.

$$\boldsymbol{t = abs\,(1 - d / d_{best})*\Theta} \tag{4}$$

where d and d_{best} are the Euclidean distances between i^{th} and j^{th} individuals ($i \neq j$) and between X_i and X_{best}. The value of angle Θ is calculated according to Eq. 5 using the dot product rule -

$$\Theta = \arccos\left\{\frac{(\vec{X}_j - \vec{X}_i).(\vec{X}_{best} - \vec{X}_i)}{|\vec{X}_j - \vec{X}_i|.|\vec{X}_{best} - \vec{X}_i|}\right\} \tag{5}$$

Considering $X_{da,best}$ as the member which has *least t-value* among all the members and $X_{da,worst}$ as the member which has *highest t-value*. Now our approach for the generation of mutant vector can be mathematically represented as –

$$\boldsymbol{V_{i,G} = X_{i,G} + F.(X_{da,best,G} - X_{da,worst,G}) + F.(X_{r1,G} - X_{r2,G})} \tag{6}$$

Where $X_{r1,G}$ and $X_{r2,G}$ are two randomly chosen vectors from all other individuals excluding X_i.

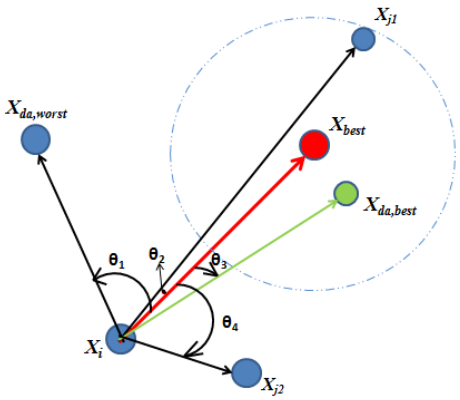


Fig. 1. Representation of Relational Neighborhood in Population Topology of i[th] member

In Fig. 1, the 2 dimensional representation of the relational neighbourhood is depicted to clarify the method of distance angle approach. Here θ_2 is the smallest angle but the distance between X_i and this individual is quite large compared to the distance between X_i and X_{best}. So the t value of this individual is larger than the individual $X_{da,best}$ shown according to the equation **(4).** In case of $X_{da,best}$ the angle θ_3 is slightly larger than θ_2 but the distance is more similar to X_{best} than it. Similar case arises to determine $X_{da,worst}$.

The relational neighbourhood technique maps the rest of the population with respect to the vector joining the i^{th} individual to the best fit member and sorts their indices based on the calculated t-value. This technique helps in balancing the pull from fittest individual as well as undesired attraction towards the worst fit member.

The scale factor F is generated by applying *Cauchy Distribution* as given below-

$$F = Cauchy\ (F_m, 0.1) \tag{7}$$

F is regenerated if F < 0 or F > 1. F_m is initialized as $F_m = 0.5$ and can be updated as $F_m = w.F + (1\text{-}w).mean_{power}\ (F_{success})$, where w is between [0.8, 1]. This updating strategy is taken from mDE_pBX [2] as follows:

$$mean_{power}(Success) = \sum_{y \in Fsuccess} \left(\frac{y^n}{Fsuccess}\right)^{1/n} \tag{8}$$

- In the crossover part, we applied the pbest crossover [2]. For each member one individual is randomly taken from the p-fittest members. Then normal binomial crossover operation is operated. Now for each generation the crossover probability is independently generated as –

$$Cr_i = Gammarand\ (0.2,\ 2) \tag{9}$$

Here the gamma random number is used to maintain the diversification. It is regenerated if it is not between 0 and 1. The two parameters used for Gammarand is shape parameter and scale parameter.

- In the last stage, a greedy selection method is applied as discussed in section 2.4.

4 Numerical Benchmarks and Comparative Study

4.1 Numerical Benchmark Set

The test functions used in this paper for the comparison of R*e*Nbd-*DE* algorithm consists of 25 benchmark functions, given in **CEC 2005 [4]**.

4.2 Algorithms Compared and Parametric setup

R*e*Nbd-*DE* is compared with some well-known DE variants which are mentioned below. 25 sample runs are taken to get the mean and standard deviation of error.

Table 1. Comparison table for D = 30 - mean and standard deviation of the error values are represented as mean (STD)

Algo / Fun	DE/rand/1	DE/current-to-best/1	DE/current-to-rand/1	DE/rand/2	MDE-pBX	DEGL	SaDE	JADE	*ReNbd-DE*
f1	2.45e-05 (3.48e-05)	2.45e-16 (3.48e-25)	1.26e-14 (5.61e-16)	9.52e-01 (2.57e-01)	2.27e-14 (2.78e-14)	2.35e-20 (5.62e-20)	6.78e-19 (2.39e-30)	1.32e-16 (9.24e-54)	**2.57e-24 (1.36e-26)**
f2	5.50e-02 (1.28e-02)	6.30e-08 (1.29e-07)	4.00e-04 (1.87e-04)	8.23e+03 (1.65e+03)	**1.25e-13 (4.25e-14)**	1.18e-07 (6.56e-08)	9.72e-08 (4.86e-07)	2.51e-06 (3.42e-26)	1.69e-11 (2.31e-12)
f3	2.89e+05 (1.94e+05)	1.89e+05 (1.04e+05)	1.43e+07 (2.04e+06)	5.00e+07 (1.27e+07)	5.48e+04 (1.78e+04)	2.31e+05 (1.03e+05)	5.05e+04 (1.58e+05)	4.74e+04 (1.62e+04)	**4.37e+04 (1.36e+03)**
f4	5.04e-01 (8.59e-01)	1.07e-02 (2.78e-02)	1.26e-01 (5.50e-02)	1.47e+04 (2.84e+03)	2.76e-08 (2.73e-08)	1.57e+03 (9.50e+00)	5.81e-06 (1.44e-05)	5.11e-07 (4.01e-07)	**2.15e-08 (1.62e-08)**
f5	1.27e+03 (2.84e+02)	5.76e+02 (1.25e+02)	**1.04e+01 (5.80e+00)**	8.07e+03 (7.14e+02)	6.16e+02 (7.44e-04)	5.07e+02 (5.80e+02)	7.88e+02 (1.24e+03)	3.27e+02 (1.84e+02)	2.01e+02 (1.31e+01)
f6	2.78e+01 (1.02e+01)	9.27e+00 (8.28e+00)	1.27e+02 (5.57e+02)	6.03e+03 (3.58e+03)	7.97e-01 (1.59e+00)	4.78e-01 (1.32e+00)	2.12e+01 (1.34e+01)	5.60e+00 (1.94e+01)	**2.09e-04 (2.92e-05)**
f7	9.66e-01 (9.15e-02)	8.42e-01 (9.14e-02)	4.70e+03 (9.07e-05)	5.47e+03 (6.72e+01)	4.69e+03 (5.75e-13)	6.99e-01 (4.54e-03)	8.27e-03 (1.14e-02)	**6.95e-03 (4.48e-03)**	3.59e+03 (2.36e+03)
f8	2.09e+01 (6.26e+01)	2.09e+01 (5.01e-06)	2.10e+01 (5.86e-02)	2.10e+01 (4.79e-02)	2.00e+01 (5.81e-02)	2.01e+01 (2.30e-02)	2.01e+01 (5.72e-02)	2.09e+01 (2.66e-02)	**1.89e+01 (2.35e-01)**
f9	4.37e+01 (9.73e+00)	2.35e+01 (8.63e+00)	1.69e+02 (8.63e+00)	2.18e+02 (1.21e+01)	1.79e+01 (4.91e+00)	1.76e+01 (3.02e+00)	2.27e+01 (1.13e-14)	1.92e+01 (8.95e-22)	**1.28e+01 (2.32e-01)**
f10	1.56e+02 (4.57e+01)	1.96e+02 (2.17e+01)	1.67e+02 (8.32e+00)	2.41e+02 (1.10e+01)	2.27e+01 (8.33e+00)	3.74e+01 (5.29e+00)	3.57e+01 (6.08e+00)	3.03e+01 (8.35e+00)	**1.45e+01 (2.33e+00)**
f11	3.26e+01 (1.10e+01)	3.17e+01 (7.60e+00)	3.95e+01 (1.34e+00)	3.95e+01 (1.28e+00)	1.46e+01 (1.65e+00)	2.73e+01 (1.57e+00)	2.65e+01 (1.12e+00)	2.64e+01 (1.91e+01)	**1.23e+01 (2.02e+00)**
f12	8.43e+04 (6.25e+04)	9.84e+04 (6.23e+03)	2.67e+03 (2.06e+03)	5.41e+05 (5.77e+04)	5.05e+03 (3.49e+03)	2.54e+04 (2.89e+03)	1.73e+02 (9.33e+02)	2.69e+04 (6.80e+03)	**1.01e+03 (8.36e+02)**
f13	4.51e+00 (2.27e+00)	3.52e+00 (2.27e+00)	1.44e+01 (8.92e-01)	2.06e+01 (1.13e+00)	3.14e+00 (7.15e-01)	2.36e+00 (5.28e-01)	3.20e+00 (1.34e-01)	3.62e+00 (4.87e-02)	**2.02e+00 (1.25e-02)**
f14	1.33e+01 (3.48e-01)	1.35e+01 (3.48e-01)	1.32e+01 (1.32e-01)	1.35e+01 (1.08e-01)	**1.23e+01 (8.69e-01)**	1.30e+01 (4.11e-01)	1.27e+01 (2.59e-01)	1.27e+01 (2.20e-01)	1.31e+01 (3.21e-01)
f15	4.84e+02 (2.15e+01)	3.84e+02 (5.14e+01)	4.13e+02 (5.07e+01)	4.04e+02 (2.41e+00)	3.60e+02 (4.89e+01)	3.44e+02 (5.07e+01)	3.27e+02 (9.64e+01)	2.88e+02 (9.05e+01)	**2.00e+02 (6.34e-01)**
f16	2.82e+02 (1.13e+01)	2.23e+02 (1.64e+02)	1.89e+02 (1.01e+01)	2.69e+02 (1.19e+01)	1.17e+02 (1.42e+02)	2.04e+02 (1.21e+02)	1.37e+02 (1.70e+01)	7.43e+01 (3.94e+01)	**1.03e+02 (3.26e+01)**
f17	3.09e+02 (1.57e+01)	2.35e+02 (1.57e+01)	2.13e+02 (1.98e+01)	2.99e+02 (1.15e+01)	8.34e+01 (4.21e+01)	1.45e+02 (7.32e+01)	1.50e+03 (9.36e+02)	8.46e+01 (3.57e+01)	**7.69e+01 (3.26e+01)**
f18	9.13e+02 (8.43e-01)	9.50e+02 (2.11e+01)	8.21e+02 (4.24e+01)	9.39e+02 (3.38e+00)	9.07e+02 (8.56e-02)	9.23e+02 (1.58e+00)	9.54e+02 (3.43e+01)	8.16e+02 (1.65e-01)	**7.05e+02 (3.26e-02)**
f19	9.20e+02 (1.22e+00)	9.52e+02 (2.11e+01)	8.53e+02 (5.39e+01)	9.41e+02 (4.01e+00)	9.04e+02 (7.57e-01)	9.20e+02 (1.34e+00)	8.45e+02 (6.21e+01)	8.16e+02 (1.54e-01)	**8.86e+02 (2.36e+01)**
f20	9.13e+02 (1.16e+00)	9.41e+02 (2.93e+01)	8.36e+02 (5.15e+01)	9.40e+02 (3.04e+00)	9.05e+02 (1.09e+00)	9.20e+02 (3.07e+01)	2.04e+03 (8.76e+02)	**8.16e+02 (1.72e-01)**	8.69e+02 (2.35e+02)
f21	5.81e+02 (2.62e+01)	8.32e+02 (2.62e+02)	**5.00e+02 (1.39e-05)**	**5.00e+02 (7.85e-02)**	**5.00e+02 (1.71e-13)**	7.57e+02 (1.01e+01)	1.73e+03 (5.11e+02)	8.58e+02 (1.10e+00)	**5.00e+02 (2.36e-14)**
f22	9.64e+02 (1.14e+01)	9.34e+02 (4.15e+01)	9.10e+02 (8.46e+00)	1.02e+03 (1.22e+01)	8.78e+02 (3.49e+01)	9.22e+02 (9.84e+01)	1.58e+03 (4.25e+02)	9.07e+02 (1.65e+00)	**8.36e+02 (3.68e+01)**
f23	6.21e+02 (3.06e+01)	8.60e+02 (2.88e+02)	**5.34e+02 (1.02e+00)**	**5.34e+02 (1.00e+00)**	**5.34e+02 (5.26e-04)**	7.52e+02 (4.00e+01)	5.50e+02 (2.48e+01)	8.65e+02 (6.61e-01)	**5.34e+02 (2.35e-05)**
f24	3.14e+02 (3.22e+01)	3.05e+02 (3.56e+01)	**2.00e+02 (0.00e+00)**	**2.00e+02 (1.65e-01)**	**2.00e+02 (0.00e+00)**	6.56e+02 (5.01e+01)	2.09e+02 (1.24e-04)	2.11e+02 (1.56e+01)	**2.00e+02 (0.00e+00)**
f25	9.86e+02 (2.17e+01)	9.68e+02 (1.31e+01)	1.66e+03 (3.11e+00)	1.70e+03 (3.95e+00)	**2.00e+00 (0.00e+00)**	9.88e+02 (3.02e+01)	5.00e+02 (5.68e-02)	2.10e+02 (2.55e+00)	**2.00e+02 (0.00e+00)**

a. DE/current-to-best/1/bin with $F = 0.8$ and $Cr = 0.9$;
b. DE/rand/1/bin with $F = 0.8$ and $Cr = 0.9$; [1]
c. DE/current-to-rand/1/bin with $F = 0.8$ and $Cr = 0.9$;
d. DE/rand/2
e. MDE-pBX [2]
f. DEGL/SAW with $\alpha = \beta = F = 0.8$, $Cr = 0.9$, and nbd size =0.1*Np. [5]
g. SaDE [8]
h. JADE [6]

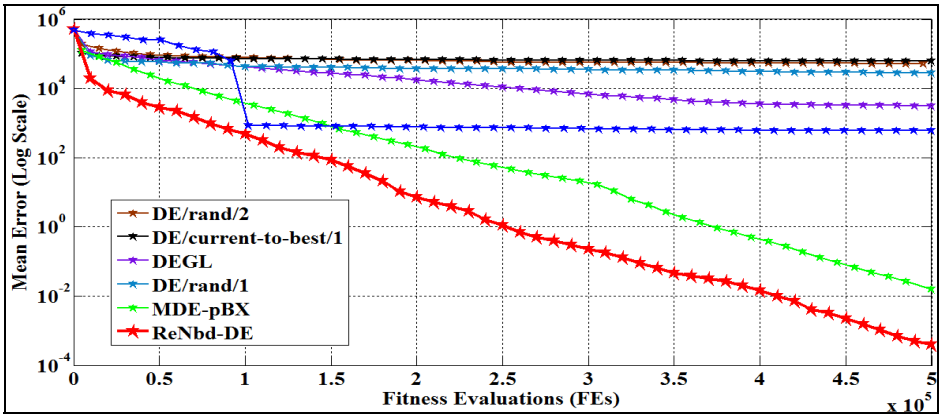
Table 2. Comparison table for D = 50 - mean and standard deviation of the error values

Algo / Fun	DE/rand/1	DE/current-to-best/1	DE/current-to-rand/1	DE/rand/2	MDE-pBX	DEGL	SaDE	JADE	*ReNbd-DE*
f1	9.60e-05 (1.28e-05)	2.14e-06 (6.13e-06)	2.56e-16 (2.34e-17)	4.38e+02 (1.41e+02)	**5.68e-14 (0.00e+00)**	6.47e-10 (9.86e-11)	1.48e-11 (2.83e-12)	7.46e-14 (2.41e-14)	**5.68e-14 (0.00e+00)**
f2	3.96e+03 (9.31e+02)	2.14e+03 (1.13e+03)	1.58e+01 (6.94e+00)	7.37e+04 (7.64e+03)	5.40e-13 (8.53e-14)	1.30e-05 (9.56e-06)	2.28e-03 (8.54e-03)	5.63e-04 (7.82e-06)	**1.25e-13 (3.26e-15)**
f3	5.47e+07 (1.33e+07)	1.03e+07 (6.38e+06)	4.04e+07 (6.87e+06)	4.46e+08 (7.24e+07)	6.49e+04 (2.13e+04)	2.31e+05 (1.03e+05)	7.17e+05 (1.00e+06)	8.71e+04 (3.68e+04)	**3.25e+04 (1.35e+04)**
f4	1.18e+04 (3.33e+03)	8.57e+04 (2.86e+03)	6.05e+02 (1.67e+02)	9.03e+04 (8.76e+03)	1.03e-02 (2.07e+00)	2.89e+04 (1.89e+01)	9.77e+04 (9.83e+01)	3.16e+03 (4.13e-01)	**2.30e-04 (2.05e-03)**
f5	8.71e+03 (6.94e+02)	7.46e+03 (1.33e+03)	9.64e+02 (5.92e+02)	2.13e+04 (1.70e+03)	**1.93e+03 (5.86e+02)**	6.09e+03 (6.84e+02)	5.99e+03 (4.46e+02)	3.05e+03 (5.48e+02)	2.03e+03 (6.29e+02)
f6	4.92e+01 (1.18e+01)	1.08e+07 (1.24e+07)	2.10e+05 (9.25e+05)	7.47e+06 (4.50e+06)	7.10e-01 (4.05e-01)	1.35e+01 (1.11e+01)	1.13e+01 (1.04e+01)	1.54e+01 (1.06e+01)	**6.10e-01 (3.00e-01)**
f7	**6.20e+03 (4.59e-12)**	6.67e+03 (1.80e+02)	**6.20e+03 (1.57e-04)**	8.65e+03 (1.70e+02)	**6.20e+03 (6.43e-13)**	**6.20e+03 (4.60e-12)**	**6.20e+03 (4.59e-12)**	**6.20e+03 (1.84e+00)**	**6.20e+03 (3.65e-13)**
f8	2.11e+01 (3.33e-02)	2.11e+01 (4.84e-02)	2.11e+01 (2.54e-02)	2.11e+01 (3.71e-02)	**2.01e+01 (6.24e-02)**	2.11e+01 (3.92e-02)	2.11e+01 (3.45e-02)	2.11e+01 (3.25e-02)	2.02e+01 (3.65e-02)
f9	3.45e+02 (1.20e+01)	2.41e+02 (2.94e+01)	3.47e+02 (1.54e+01)	4.68e+02 (1.92e+01)	5.08e+01 (1.46e+00)	1.62e+02 (1.74e+01)	1.14e+02 (1.26e+01)	1.35e+02 (2.59e+00)	**4.82e+01 (9.26e+00)**
f10	3.76e+02 (1.58e+01)	2.55e+02 (7.66e+01)	3.44e+02 (1.71e+01)	5.24e+02 (2.32e+01)	4.44e+01 (3.87e-01)	1.02e+02 (3.56e+01)	6.34e+01 (1.28e+01)	1.93e+02 (2.06e+01)	**4.04e+01 (2.06e-01)**
f11	7.26e+01 (1.21e+00)	4.95e+01 (4.45e+00)	7.25e+01 (1.66e+00)	7.29e+01 (1.09e+00)	4.21e+01 (2.51e+00)	6.29e+01 (1.36e+01)	6.63e+01 (1.4e+00)	6.20e+01 (1.74e+00)	**4.01e+01 (3.26e-02)**
f12	2.05e+06 (5.93e+05)	2.51e+05 (1.14e+05)	9.11e+03 (1.02e+04)	2.62e+06 (2.36e+05)	2.56e+04 (1.06e+02)	5.78e+04 (4.56e+04)	8.78e+03 (7.09e+03)	1.76e+05 (7.10e+04)	**6.26e+03 (5.32e+03)**
f13	3.60e+01 (1.45e+00)	3.41e+01 (8.90e+00)	2.96e+01 (1.11e+00)	4.33e+01 (2.17e+00)	8.31e+00 (4.61e-01)	3.06e+01 (4.36e+00)	2.77e+01 (4.10e+00)	2.31e+01 (4.78e-01)	**4.32e+00 (3.25e-02)**
f14	2.34e+01 (1.49e-01)	2.29e+01 (4.09e-01)	2.29e+01 (1.58e-01)	2.32e+01 (1.58e-01)	**2.15e+01 (8.36e-01)**	2.26e+01 (3.38e-01)	2.28e+01 (2.06e-01)	2.20e+01 (2.56e-01)	2.20e+01 (2.32e-01)
f15	6.39e+02 (7.98e+01)	4.79e+02 (5.00e+01)	3.97e+02 (5.07e+01)	4.04e+02 (6.90e+00)	2.66e+02 (1.49e+02)	3.98e+02 (4.93e+01)	3.88e+01 (1.07e+02)	3.76e+02 (8.76e+01)	**2.00e+02 (2.36e-06)**
f16	2.73e+02 (1.05e+01)	2.54e+02 (1.48e+01)	2.42e+02 (9.53e+00)	3.53e+02 (1.36e+01)	**4.18e+01 (8.17e+00)**	1.32e+02 (2.00e+01)	1.54e+02 (6.16e+01)	1.43e+02 (5.22e+01)	9.85e+01 (9.36e+00)
f17	3.73e+02 (3.13e+01)	2.52e+02 (5.73e+01)	2.66e+02 (1.01e+01)	4.16e+02 (1.90e+01)	2.27e+02 (1.76e+02)	**1.77e+02 (2.37e+01)**	1.93e+02 (2.96e+00)	1.89e+02 (3.87e+01)	1.88e+02 (2.00e+01)
f18	9.90e+02 (4.87e+01)	9.21e+02 (5.66e+01)	8.00e+02 (2.07e+02)	1.04e+03 (9.71e+00)	9.37e+02 (1.79e-01)	9.61e+02 (2.85e+01)	9.44e+02 (5.20e+01)	9.20e+02 (1.89e+00)	**9.09e+02 (3.25e-01)**
f19	9.41e+02 (3.80e+01)	9.27e+02 (5.99e+01)	8.47e+02 (1.19e+02)	1.03e+03 (1.33e+01)	9.46e+02 (1.52e+01)	9.14e+02 (2.01e+01)	9.34e+02 (1.96e+01)	9.60e+02 (2.56e+01)	**8.72e+02 (2.36e+01)**
f20	9.85e+02 (4.50e+01)	9.36e+02 (5.40e+01)	8.34e+02 (1.56e+02)	1.03e+03 (1.01e+01)	9.47e+02 (9.77e+00)	9.22e+02 (4.59e+01)	9.31e+02 (2.01e+01)	9.86e+02 (1.86e+02)	**9.13e+02 (6.25e+01)**
f21	9.11e+02 (5.75e+02)	8.95e+02 (1.75e+02)	5.20e+02 (7.61e+01)	6.61e+02 (4.74e+01)	**5.00e+02 (5.68e-13)**	8.36e+02 (2.18e+02)	8.64e+02 (1.57e+02)	8.52e+02 (3.51e+02)	**5.00e+02 (3.06e-13)**
f22	9.95e+02 (1.35e+01)	9.34e+02 (1.10e+01)	9.49e+02 (1.39e+01)	1.14e+03 (1.05e+01)	9.05e+02 (2.91e+01)	9.42e+02 (3.56e+01)	9.72e+02 (3.33e+01)	9.13e+02 (2.43e+01)	**8.65e+02 (3.66e+01)**
f23	9.12e+02 (2.60e+01)	9.26e+02 (2.60e+02)	5.47e+02 (4.02e+01)	7.17e+02 (5.41e+01)	7.26e+02 (1.69e+02)	8.39e+02 (1.66e+02)	8.64e+02 (1.56e+02)	8.10e+02 (2.45e+02)	**5.39e+02 (1.02e+01)**
f24	7.98e+02 (2.46e+01)	7.95e+02 (1.36e+01)	**2.00e+02 (4.01e-06)**	6.38e+02 (9.29e+01)	**2.00e+02 (0.00e+00)**	7.25e+02 (8.31e+01)	**2.00e+02 (0.00e+00)**	**2.00e+02 (0.00e+00)**	**2.00e+02 (0.00e+00)**
f25	1.76e+02 (4.54e+00)	1.78e+03 (6.77e+00)	1.69e+03 (9.21e+00)	1.80e+03 (6.27e+00)	**2.00e+00 (0.00e+00)**	1.67e+03 (6.51e+00)	1.75e+03 (3.14e+00)	1.66e+03 (5.52e+00)	**2.00e+02 (0.00e+00)**

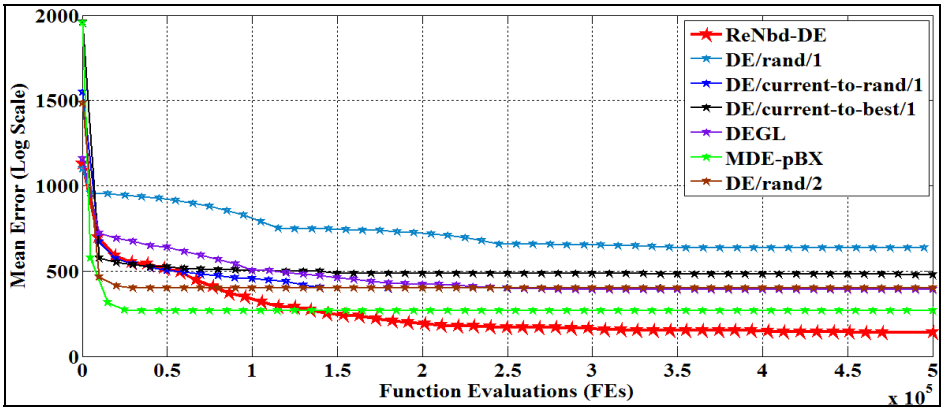
4.3 Comparative Study on Numerical Benchmarks

The data tabulated in Table-1 and Table-2, depict that the proposed algorithm has outperformed several DE incorporated mutation strategis and four well known adaptive variants in most of the cases. The *ReNbd-DE* has provided 18 best

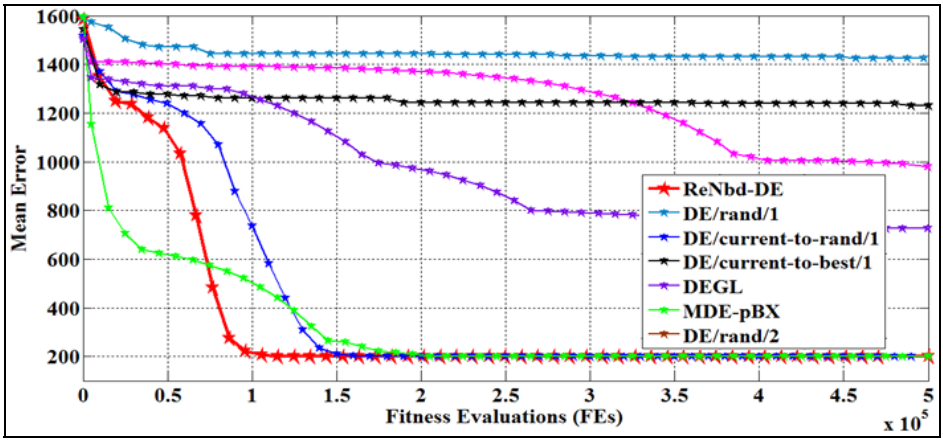
Supplementary Attachment: Convergence graph of Benchmark functions (50D)



(a) $f4$: Shifted Schwefel's Problem with Noise in Fitness



(b) $f15$: Hybrid Composition Function



(c) $f24$:Rotated Hybrid Composition Function

Fig. 2. Convergence graph of $f4$, $f15$ and $f24$

performance out of 25 functions in 30D and 20 best performance out of 25 functions in 50D. The usage of population topology in addition to adaptive parameter tuning have helped in providing promising results. The euclido-angular perturbation strategy helps to balance exploration and exploitation (*T:ER&EI)*. In fact the direction information obtained from population topology guides the *Re*-NbdDE into intelligent directions in the solution sapce. It can be safely concluded that combining paramter adaptation with intelligent search makes *Re*-Nbd DE a novel optimizing algorithm. The convergence graphs of the proposed ReNbd framework in DE along with some competing algorithms have been illustrated in Fig 2 for test functions $f4$, $f15$ and $f24$.

5 Conclusion and Future Work

We conclude that the *ReNbd* framework proposed in this paper improves the performance of DE by utilizing neighbourhood-based information. This technique helps in partitioning the dependence on both distance as well as angular displacement. The main crux of *ReNbd* lies in replacing Gaussian distribution used in crossover is replaced by Gamma distribution since it is more distributive in nature and permits greater mixing of parameters. We wish to investigate the implementation of the *ReNbd* framework in the classical as well as the state-of-the-art DE variants and make use of performance based adaptation by modifying the probability distributions used in near future.

References

[1] Storn, R., Price, K.V.: Differential evolution—A simple and efficient adaptive scheme for global optimization over continuous spaces, ICSI, Berkeley, CA, Tech. Rep. TR-95-012, `http://http.icsi.berkeley.edu/~storn/litera.html`

[2] Islam, S.M., Das, S., Ghosh, S., Roy, S., Suganthan, P.N.: An Adaptive Differential Evolution Algorithm With Novel Mutation and Crossover Strategies for Global Numerical Optimization. SMC-B (2011)

[3] Weber, M., Tirronen, V., Neri, F.: Scale factor inheritance mechanism in distributed differential evolution. Soft Comput., Fusion Found. Methodologies Appl. 14(11), 1187–1207 (2010)

[4] Suganthan, P.N., Hansen, N., Liang, J.J., Deb, K., Chen, Y.P., Auger, A., Tiwari, S.: Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Nayang Technol. University, Singapore (2005)

[5] Das, S., Abraham, A., Chakraborty, U.K., Konar, A.: Differential evolution using a neighbourhood based mutation operator. IEEE Trans. Evol. Comput. 13, 526–553 (2009)

[6] Zhang, J., Sanderson, A.C.: JADE: Adaptive differential evolution with optional external archive. IEEE Trans. Evol. Comp. 13(5), 945–958 (2009)

[7] Brest, J., Greiner, S., Bošković, B., Mernik, M., Žumer, V.: Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. IEEE Trans. Evol. Comput. 10(6), 646–657 (2006)

[8] Qin, A.K., Huang, V.L., Suganthan, P.N.: Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans. Evol. Comput. 13(2), 398–417 (2009)
[9] Liu, J., Lampinen, J.: Adaptive parameter control of differential evolution. In: Matoušek, R., Ošmera, P. (eds.) Proc. 8th MENDEL, pp. 19–26 (2002)
[10] Gong, W., Cai, Z., Ling, C.X., Li, H.: Enhanced differential evolution with adaptive strategies for numerical optimization. IEEE Trans. Syst., Man, Cybern. B, Cybern. 41(2), 397–413 (2011)
[11] Weber, M., Neri, F., Tirronen, V.: Distributed differential evolution with explorative–exploitative population families. Genetic Programm. Evol. Mach. 10(4), 343–371 (2009)
[12] Mallipeddi, R., Suganthan, P.N., Pan, Q.K., Tasgetiren, M.F.: Differential evolution algorithm with ensemble of parameters and mutation strategies. Appl. Soft Comput. 11(2), 1679–1696 (2011)
[13] Das, S., Suganthan, P.N.: Differential Evolution: A Survey of the State-of-the-art. IEEE Transactions on Evolutionary Computation 15(1), 4–31 (2011), doi:10.1109/TEVC.2010.2059031

A Simulated Annealing Heuristic for Minimizing Makespan in Parallel Machine Scheduling

Dipak Laha

Department of Mechanical Engineering,
Jadavpur University, Kolkata, India
dipaklaha_jume@yahoo.com

Abstract. This paper deals with the problem of scheduling a set of n independent jobs to be processed on m identical parallel machines in order to minimize makespan. This problem is known to be NP-complete. A SA based heuristic is presented to solve this problem. Empirical results with a large number of randomly generated problem instances demonstrate that the proposed method produces solutions that are fairly superior to that of the best-known method in the literature while not affecting its computational effort.

Keywords: Parallel machine scheduling, makespan, simulated annealing, heuristic, optimization, lower bound.

1 Introduction

The problem of parallel machine scheduling (PMS) has long received the attention of researchers since the seminal work of McNaughton [1]. Apart from theoretical research, these problems are also routinely encountered in many real-life situations such as production lines, shipping docks, university, hospitals, and computer systems. A comprehensive survey of this scheduling research and applications are given by Chen and Sin [2].

In this paper, we present a SA heuristic to minimize makespan for the n-job, m-machine identical PMS problem. The heuristic is compared with the SA of Lee et al. [3].

The remainder of this paper is organized as follows: Section 2 defines the problem and briefly discusses the review of its solution algorithms. The proposed SA heuristic is presented in Section 3. The computational results are provided in Section 4. Finally conclusions are made in Section 5.

2 Problem Formulation and the Existing Algorithms

This paper considers a PMS problem in which a set $N = \{1,2,\ldots,n\}$ of n independent jobs available at time zero, to be assigned on a set $M = \{1,2,\ldots,m\}$ of m identical parallel machines. The processing time (including the setup times required) for each job $i \in N$ is p_i. We assume that a machine processes one job at a time and once a job

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 198–205, 2012.

starts processing on either of the m machines it must be completed without preemption. Let S_j be the subset of jobs to machine M_j for a schedule of jobs with the completion time $C(S_j) = \sum_{i \in S_j} p_i$. The makespan or the maximum completion time of a schedule S of n jobs is given by

$$MS = C_{max}(S) = \max_{1 \le j \le m} C(S_j) \quad (1)$$

The problem is to determine m subsets containing n jobs such that the MS given in Equation (1) is minimized.

Since the PMS problems belong to NP-hard [4], several heuristics with polynomial complexity have been developed for obtaining near-optimal solutions. Some noteworthy heuristics for the MS minimization problem under consideration have been proposed by Graham [5], Coffman et al. [6], Lee and Massey [7], Ghomi and Ghazvimi [8], and Gupta and Ruiz-Torres [9]. It may be noted that the well-known LPT heuristic [5] has been intensely studied because it is very simple, fast, and yields considerably good solution. The computational results [3] have shown that the heuristic of Gupta and Ruiz-Torres [9], and Ghomi and Ghazvimi [8] are superior compared to other heuristics, however, at the cost of their some additional computational effort.

Similar to other soft computing methods like GA, PSO, and ACO, SA has also been successfully used to many combinatorial problems. It has been found effective in well-known flow shop scheduling [10-11]. This algorithm has also been applied to a variety of different PMS problems [3, 12 -21]. It has been, and continues to be, used by itself as well as a part of hybrid configurations in the scheduling literature.

Low [12] proposed a SA heuristic to minimize the total flow time for the multistage flow shops with unrelated parallel machines. Radhakrishnan and Venture [13] proposed a new SA for the parallel machine with SDST problems. Ruiz-Torres et al. [14] applied SA in PMS with bi-criteria. Behnamian et al. [15] developed an effective hybrid metaheuristic combining ACO, SA, and VNS in SDST PMS problems. Chang et al. [16] proposed a SA approach to minimize MS for identical parallel batch-processing machines. Their approach outperforms CPLEX on most of the instances. Kim et al. [17] suggested improved SA and TS heuristics for PMS problems with SDST and distinct ready times. Damodaran et al. [18] presented a SA algorithm to minimize MS on identical parallel batch processing machines with non-identical size and non-zero ready times of jobs. The computational results showed that their approach is comparable to GRASP. Li et al. [19] designed a SA to minimize MS for identical PMS with controllable processing times assuming limited total resource consumption. Li et al [20] proposed SA to the uniform PMS to minimize the maximum lateness for a large set of instances. Lee and Pinedo [21] presented a SA method in the third phase of their three phase heuristic for minimizing the sum of the weighted tardiness in SDST PMS problems. Lee et al. [3] proposed a SA heuristic for solving identical PMS to minimize MS. They have shown that their approach outperforms the heuristics of Gupta and Ruiz-Torres [9], and Ghomi and Ghazvimi [8]. However, the comparative time-complexity analysis has not been reported in their article.

3 Simulated Annealing

SA [22-24] based on ideas drawn from statistical physics, is a structured yet randomized local search and optimization algorithm that leads to near-optimal solution.

It begins with a randomly generated initial solution, as the current solution and is further improved by generating a neighborhood solution of this current solution. If the neighborhood is better than the current solution, it is unconditionally accepted as the new current solution. Otherwise, it is not rejected outright, but accepted with a certain probability $e^{-\delta E/T}$.

At the beginning of an SA run, the probability of accepting a worse solution is kept high to reduce the chance of the algorithm getting trapped in a local optimum. As the number of iterations increases, this probability is reduced according to an annealing schedule. A control parameter, called temperature is used to alter this probability. Usually, this temperature is started at high value and is gradually reduced as the run progresses.

3.1 The Proposed Heuristic

We now describe the proposed SA heuristic to minimize *MS* in an identical PMS problem.

Inputs. A set $N = \{1,2,\ldots,n\}$ of n jobs, a set $M = \{1,2,\ldots,m\}$ of m identical parallel machines, processing time t_i for each job $i \in N$ and $P = \sum_{i=1}^{n} p_i$.

Step 1. Obtain the initial schedule (S) using the LPT algorithm [5] and its MS (r). Consider it the current schedule as S_c and the corresponding *MS* (r_c). Let *MS* of the best-so-far schedule (S_b) be r_b. Set $S_b \leftarrow S_c$, $r_b \leftarrow r_c$. Initialize *max-count* and the starting temperature ($T1$). $T1 = 1000$ (in this experiment). Set $k = 1$.

Step 2. Determine the *LB* on the *MS* given as

$$LB = \left\lceil \max\left\{ \max_{1 \le i \le n} p_i ; P/m \right\} \right\rceil \tag{2}$$

Step 3. Identify S_j, the subset of jobs assigned to machine j among $\{S_1, S_2, \ldots, S_m\}$ where its $C(S_j)$ for the schedule S_c, $C(S_c) = r_c$.

Step 4. If $r_c = LB$, then the procedure terminates and the optimum schedule is obtained.

Step 5. Select each job from S_c obtained in Step 3 and correspondingly interchange with another job picked randomly from the subsets of jobs of the remaining machines (each case being independent of the others) to produce a set of schedules. In this case, the total number of generated schedules will be same as the number of jobs in the subset S_c. Select the best schedule among the generated schedules and consider as the new generated schedule ($S_{c'}$) with the *MS* ($r_{c'}$).

Step 6. if $r_{c'} < r_c$

{

6.1 accept $S_c \leftarrow S_{c'}$ and $r_c \leftarrow r_{c'}$.

6.2 if $r_{c'} < r_b$ then set $S_b \leftarrow S_{c'}$, $r_b \leftarrow r_{c'}$ and return to Step 4;

else return to Step 5.
}
else
6.3 with probability $exp((r_c - r_{c'})/(r_c.T_k))$, Set $S_c \leftarrow S_{c'}$ and $r_c \leftarrow r_{c'}$.

Step 7. Set $T_{k+1} = 0.9T_k$.

Step 8. Set $k = k + 1$.

Step 9. If $k \leq$ *max-count* and the total number of *MS* evaluations are less than *max-MS-eval*, return to Step 5.

Outputs. Best-so-far schedule (S_b) with the corresponding *MS* (r_b)

Note that every call to the *MS* evaluation function results in the number of *MS* evaluations being incremented by one. The purpose of using both parameters, *max-count* and *max-MS-eval* is to study the quality of solution given a sufficiently large number of *max-count*, a number that exceeds *max-MS-eval*. In practice, only one of the two would be needed.

3.2 Computational Complexity

The complexity of the proposed algorithm can be computed as follows. Step 1 can be performed in O($n$log$n$). Steps 2 and 4 take constant time. Step 3 requires O(m) time. Step 5 can be executed in O(n^2) time. Steps 6 through 9 take a constant amount of time. Since the loop spanning Steps 2 to 9 is performed max-count times, the total complexity of the proposed algorithm is O(n^2) where the constant max-count is hidden in the asymptotic notation.

4 Computational Experimentation

Both the method of Lee et al. [3] and the proposed heuristic were coded in C and run on a PC with Intel (R) Core i7, 6 GB RAM and 2.30 GHz.

To compare their performance, we carried out the same experimental framework as was used by Lee et al. [3] with n = 30, 50, 100 and m = 2, 3, 4, 6, 8, 10 and for n = 10, m = 2, 3. One hundred independent problem instances were generated for each combination of m and n. hence, a total number of 2000 problem instances were considered. As is typically used in SA [3], the processing time probability distribution follows a discrete U(100,800).

To investigate the relative performance of these heuristics in situations dealing with high variability of processing times and large number of jobs, we considered another computational experiment with U(1,100), and U(100, 800) by varying n = 200, 500 and m = 5, 10, and 20. Each problem size comprised one hundred problem instances, resulting in a total of 600 problem instances.

In order to compare their performance, we considered two metrics for the performance measures, namely, the maximum ratio and the average ratio of the *MS* over its *LB* obtained from Equation (2) for a given problem size are defined as follows.

$$m_h = \max_{1 \le i \le m} \left({}^{MS(H_i)}/_{LB_i} \right) \tag{3}$$

$$r_h = \sum_i^{np} \left({}^{MS(H_i)}/_{LB_i} \right) / np \tag{4}$$

Where *np* is the number of problem instances for a given job and machine combination. $MS(H_i)$ is the *MS* value obtained by the heuristic *H* for the *i*-th problem instance.

Table 1. Comparison of r_h values available in [3] and obtained as output by coding SA [3]

n	*m*	No. of problem instances	r_h available in [3] (*A*)	r_h obtained by coding SA [3] (*B*)	percent error $\left(\frac{B-A}{A}\right) \times 100$
10	2	100	1.0015	1.0017	0.02
	3	100	1.0090	1.0118	0.28
30	2	100	1.0000	1.0001	0.01
	3	100	1.0000	1.0013	0.13
	4	100	1.0000	1.0060	0.60
	6	100	1.0004	1.0134	1.29
	8	100	1.0015	1.0330	3.15
	10	100	1.0071	1.0367	2.94
50	2	100	1.0000	1.0000	0
	3	100	1.0000	1.0006	0.06
	4	100	1.0000	1.0025	0.25
	6	100	1.0000	1.0124	1.24
	8	100	1.0002	1.0215	2.13
	10	100	1.0005	1.0162	1.57
100	2	100	1.0000	1.0000	0
	3	100	1.0000	1.0002	0.02
	4	100	1.0000	1.0006	0.06
	6	100	1.0000	1.0044	0.44
	8	100	1.0000	1.0095	0.95
	10	100	1.0000	1.0043	0.43
	Average:				0.78

In order to check the accuracy of results of the coded SA [3] in this study, we considered the same experimental framework as was done by Lee et al. [3]. The detailed comparative results of the program output (*B*) for the SA [3] and the corresponding published results (*A*) are presented in Table 1. For each problem size, the percent error is computed considering these two values. The data set *A* is found to be in good agreement with the data set *B*. The overall average percent error is found to be 0.78. It may be noted that since Lee et al. [3] coded their heuristic in FORTRAN and the same was coded in C in the present study, there is some variation in r_h values which are obtained as outputs of these programs due to different randomly generated processing times of jobs for a particular problem size.

Table 2. Comparison of the SA [3] and the proposed heuristic considering $U(100,800)$

n	m	SA [3]			proposed heuristic		
		r_h	m_h	Average *MS* evaluations	r_h	m_h	Average *MS* evaluations
10	2	1.0017	1.0067	579	1.0042	1.0253	531
	3	1.0118	1.0375	832	1.0151	1.0508	801
30	2	1.0001	1.0003	1325	1.0002	1.0011	783
	3	1.0013	1.0034	1944	1.0006	1.0019	894
	4	1.0060	1.0147	2744	1.0016	1.0056	900
	6	1.0134	1.0315	1484	1.0043	1.0125	900
	8	1.0330	1.0551	1608	1.0111	1.0294	900
	10	1.0367	1.0819	1306	1.0177	1.0530	900
50	2	1.0000	1.0002	1714	1.0000	1.0002	1032
	3	1.0006	1.0014	4331	1.0002	1.0006	1450
	4	1.0025	1.0056	4600	1.0005	1.0015	1500
	6	1.0124	1.0222	3547	1.0015	1.0041	1500
	8	1.0215	1.0330	2812	1.0034	1.0075	1500
	10	1.0162	1.0353	1853	1.0056	1.0118	1500
100	2	1.0000	1.0001	2474	1.0000	1.0001	1741
	3	1.0002	1.0005	8303	1.0000	1.0001	2631
	4	1.0006	1.0014	4475	1.0001	1.0004	2809
	6	1.0044	1.0075	5743	1.0004	1.001	3000
	8	1.0095	1.0139	4565	1.0009	1.0024	3000
	10	1.0043	1.0123	3352	1.0013	1.0031	3000

To obtain a fair comparison of the proposed heuristic versus SA [3] for each problem instance, we need to have a cut-off that can be provided by the number of trial solutions generated or number of *MS* evaluations before the final solution is obtained. Since the number of *MS* evaluations required by the SA [3] varies from one problem instance to another for a problem size, we compute the average number of *MS* evaluations for each problem size. Then, the same number or less than this is used to terminate the execution of the proposed heuristic to allocate slightly more time to the SA [3], deliberately erring on the advantageous side.

Table 2 displays the comparative evaluations of the proposed method and the SA [3]. The results show that the proposed heuristic produces better results than those obtained by the existing method on most of the cases. The r_h values for the proposed heuristic ranges from 1.0000 to 1.0177, while the m_h is obtained varies from 1.0001 to 1.0530 compared with the corresponding values of 1.0000 -1.0367 and 1.0001-1.0819 for the SA heuristic [3]. Clearly, the proposed heuristic outperforms the SA [3]. It is also noted that the overall average number of *MS* evaluations required by the proposed method is less than that required by the SA [3].

As depicted in Table 3, an identical metrics of the performance of the proposed heuristic carries over to large –size problems as well.

Table 3. Comparison of the SA [3] and the proposed heuristic for large problems

n	m	SA [3]			proposed heuristic		
		r_h	m_h	Average *MS* evaluations	r_h	m_h	Average *MS* evaluations
				$U(1,100)$			
200	5	1.00051	1.00105	5816	1.00040	1.00053	5033
	10	1.00179	1.00417	6357	1.00107	1.00210	5899
	20	1.00664	1.01240	6326	1.00353	1.00622	6001
500	5	1.00016	1.00040	12038	1.00015	1.00021	9361
	10	1.00041	1.00083	14317	1.00038	1.00083	12902
	20	1.00133	1.00238	15358	1.00099	1.00166	15001
				$U(100,800)$			
200	5	1.00029	1.00077	7121	1.00006	1.00017	4885
	10	1.00113	1.00365	6377	1.00038	1.00119	6001
	20	1.00408	1.00908	6347	1.00206	1.00554	6001
500	5	1.00005	1.00016	15916	1.0002	1.00005	10672
	10	1.00018	1.00050	15435	1.00008	1.00018	14001
	20	1.00069	1.00150	15390	1.00036	1.00107	15001

It is evident from the results of Table 3 that the proposed heuristic outperforms the existing heuristic for large-size problems as well.

As regard to the effectiveness of the proposed heuristic, the results in Tables 1-2 depict that overall relative performance of the proposed heuristic versus SA [3] increases as the ratio n/m increases.

5 Conclusions

This paper considers an identical parallel machine scheduling problem of n-job, m -machine with the objective of minimizing makespan. We have presented an improved simulated annealing heuristic for this scheduling problem. It has been shown to yield results that are significantly better than that produced by the best-known heuristic in the literature. The other advantages of the proposed heuristic include efficient, a very simple annealing schedule, fewer parameters, and ease of implementation.

References

1. McNaughton, R.: Scheduling with deadlines and loss functions. Management Science 6, 1–8 (1959)
2. Chen, T.C.E., Sin, C.C.S.: A state-of-the-art review of parallel-machine scheduling research. Euro. J. Oper. Res. 47, 271–292 (1990)
3. Lee, W.-C., Wu, C.-C., Chen, P.: A simulated annealing approach to makespan minimization on identical parallel machines. Int. J. Adv. Manuf. Technol. 31, 328–334 (2006)

4. Garey, M.R., Johnson, D.S.: Computers and intractability: A guide to the theory of NP-completeness. W.H. Freeman, New York (1979)
5. Graham, R.L.: Bounds on multiprocessor timing anomalies. SIAM J. Appl. Math. 17, 416–429 (1969)
6. Coffman, E.G., Garey, M.R., Johnson, D.S.: An application of bin-packing to multi- processor scheduling. SIAM J. Comput. 7, 1–17 (1978)
7. Lee, C.Y., Massey, J.D.: Multiprocessor scheduling combining LPT and MULTIFIT. Discrete Appl. Math. 20, 233–242 (1988)
8. Ghomi, S.M.T., Ghazvini, F.J.: A pairwise interchange algorithm for parallel machine scheduling. Prod. Plan Control 9, 685–689 (1998)
9. Gupta, J.N.D., Ruiz-Torres, J.: A LISTFIT heuristic for minimizing makespan on identical parallel machines. Prod. Plan Control 12, 28–36 (2001)
10. Osman, I.H., Potts, C.M.: Simulated annealing for permutation flowshop scheduling. Omega 17, 551–557 (1989)
11. Laha, D., Chakraborty, U.K.: An efficient stochastic hybrid heuristic for flowshop scheduling. Eng. Appl. Artif. Intell. 20, 851–856 (2007)
12. Low, C.: Simulated annealing heuristic for flow shop scheduling problems with unrelated parallel machines. Comput. Oper. Res. 32, 2013–2025 (2005)
13. Radhakrishnan, S., Ventura, S.: Simulated annealing for parallel machine scheduling problem with earliness-tardiness penalties and sequence-dependent set-up times. Int. J. Prod. Res. 38, 2233–2252 (2000)
14. Ruiz-Torres, A.J., Enscore, E.E., Barton, A.A.: Simulated annealing heuristics for the average flow-time and number of tardy jobs bi-criteria identical parallel machine problem. Comput. Ind. Eng. 33, 257–260 (1997)
15. Behnamian, J., Zandieh, M., Ghomi, S.M.T.: Parallel-machine scheduling problems with sequence-dependent setup times using ACO, SA and VNS hybrid algorithm. Exp. Sys. Appl. 36, 9637–9644 (2009)
16. Chang, P.Y., Damodaran, P., Melouk, S.: Minimizing makespan on parallel batch processing machines. Int. J. Prod. Res. 42, 4211–4220 (2004)
17. Kim, S., Choi, H.S., Lee, D.H.: Scheduling algorithms for parallel machines with sequence-dependent set-up and distinct ready times: Minimizing total tardiness. J. Eng. Manuf. 221, 1087–1096 (2007)
18. Damodaran, P., Velez-Gallego, M.C.: A simulated annealing algorithm to minimize makespan of parallel batch processing machines with unequal job ready times. Exp. Sys. Appl. 39, 1451–1457 (2012)
19. Li, K., Shi, Y., Yang, S.-L., Cheng, B.-Y.: Parallel machine scheduling problem to minimize the makespan with resource dependent processing times. Appl. Soft. Comp. J. 11, 5551–5557 (2011)
20. Li, K., Yang, S.-L., Ma, H.-W.: A simulated annealing approach to minimize the maximum lateness on uniform parallel machines. Math. Comp. Model 53, 854–860 (2011)
21. Lee, Y.H., Pinedo, M.: Scheduling jobs on parallel machines with sequence-dependent setup times. Euro. J. Oper. Res. 100, 464–474 (1997)
22. Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P.: Optimization by simulated annealing. Science 220, 671–680 (1983)
23. Aarts, E.H.L., Korst, J.H.M.: Simulated annealing and Boltzman machines, Chichester (1989)
24. Van Laarhoven, P.J.M., Aarts, E.H.L.: Simulated annealing: Theory and applications. Reidel, Dordrecht (1987)

Rule Extraction from DEWNN to Solve Classification and Regression Problems

Nekuri Naveen[1,2], Vadlamani Ravi[1], and Chillarige Raghavendra Rao[2]

[1] Institute for Development and Research in banking Technology, Castle Hills, #1, Masab Tank, Hyderabad – 500057 (A P) India
[2] Department of Computer & Information Sciences, University of Hyderabad, Hyderabad – 500046 (A P) India
naveen.nekuri@gmail.com, rav_padma@yahoo.com, crrcs@uohyd.ernet.in

Abstract. This paper proposes a method to extract rules from differential evolution trained wavelet neural network (DEWNN) [1]. for solving classification and regression problems. The rule generation methods viz., Decision Tree (DT), Ripper and Classification and Regression Tree (CART) and Dynamic Evolving Neuro Fuzzy Inference System (DENFIS) are employed to extract rules from DEWNN for classification and regression problems respectively. The feature selection algorithm adapted by Chauhan et al., [1] is used in the present study. The effectiveness of the proposed hybrid is evaluated on Iris, Wine and four bankruptcy prediction datasets namely Spanish banks, Turkish banks, US banks, UK banks and Auto MPG dataset, Body fat dataset, Boston Housing dataset, Forest Fires dataset, Pollution dataset, by using 10-fold cross validation. From the results, it is concluded that the proposed hybrid method performed well in terms of sensitivity in classification problems.

Keywords: Differential Evolution Trained Wavelet Neural Network, Classification, Regression, Rule Extraction, Decision Tree, Ripper, Classification and Regression Tree, Dynamic Evolving Neuro Fuzzy Inference System.

1 Introduction

For the last two decades, artificial neural networks (ANN) such as Multilayer Perceptron (MLP) and Radial Basis Function Network (RBF) have been used with great success in solving classification and regression problems. However, ANN are treated as black boxes because, the knowledge learned by them cannot be interpreted by the human. To overcome this drawback, many researchers proposed methods in extracting knowledge from trained ANN in the form of *if-then* rules. Advantages in extracting *if-then* rules are that they can be used as an early warning expert system, which is easy to interpret and human comprehensible. In real-time applications like Bankruptcy Prediction [2], Churn Prediction, Medical Diagnosis [3], Credit approval in banks, Fraud detection, it is very much important to understand the knowledge gained by ANN. Gallant [4] first introduced rule extraction from the trained neural network.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 206–214, 2012.

Then, the works of Andrews et al., [3], Fu [5] and Towel and Shavlik [6] followed. Arbatli and Akin [7] extracted rules from trained neural network using genetic algorithms, while Fan and Li [8] extracted diagnostic rules from trained feed forward neural network. Krishnan et al., [9] extracted rules from trained neural network using combination tree. Recently, Naveen et al., [2] extracted rules from Differential Evolution trained RBF (DERBF) using GATree. Later, Naveen et al., [2] extracted rule from Group Method of Data Handling network (GMDH) using decision tree (DT) and Ripper to predict churn in bank credit cards.

Another class of powerful networks includes Wavelet Neural Network (WNN), which found applications in chemistry [10], texture classification [11], feature selection and recognition of text independent speaker [12], ECG beat classification [13], software reliability prediction [14] and analysis of sound pattern [15].

WNN uses a gradient descent technique for training which suffers from well known drawbacks such as entrapment in local minimum, slow convergence and the need of differentiability of the objective function that are associated with calculus based optimization techniques. Literature abounds with the applications of metaheuristics to train ANN and in particular WNN. For instance, the readers may refer to Yu et al.,[16], Pan et al., [17], TAWNN [18] and DEWNN [1].

The remainder of the paper is as follows: In Section 2, we overview the intelligent methods i.e., DEWNN, DT, Ripper, CART and DENFIS. Section 3 describes proposed hybrid. In section 4, the results and discussions are presented. Finally Section 5 concludes the work.

2 Overview of the Intelligent Techniques

DEWNN [1]

Differential Evolution (DE) trained WNN (DEWNN) proposed by Chauhan et al., [1] consists of three layers namely input, hidden and output layers. Chauhan et al., [1] used Gaussian wavelet function viz., $f(t) = e^{-t^2}$ as an activation function in the hidden layer, while the output layer is retained as linear one. Here, we employed Morlet wavelet function viz.,$f(t) = \cos(1.75 * t)\, e^{-\frac{t^2}{2}}$ in addition to Gaussian wavelet function. Chauhan et al., [1] adapted Garson's feature selection in DEWNN. We also employed the same feature selection algorithm in the present study. For more details of DEWNN, the reader is referred to [1].

Decision Tree (DT)

DT was introduced by the Quinlan [19]. DT is too well known to be described in detail here, one can refer to [19] for further details.

Ripper

Ripper, proposed by Cohen [20], is a rule induction algorithm which yields rules that are good at generalization and easy to interpret. For details, one can refer to [20].

Classification and Regression Tree (CART)

CART [21] is used to generate decision tree to solve both classification and regression problems. CART applies binary recursive split in building a tree. For the sake of brevity, details of CART are not presented here.

Dynamic Evolving Neuro-Fuzzy Inference System (DENFIS)
Kasabov and Song [22] proposed neuro-fuzzy inference system for adaptive online learning which generates fuzzy rules. DENFIS works with both on-line and off-line models and uses Takagi-Sugeno type fuzzy inference engine to generate fuzzy rules. By using the local information, it generates fuzzy rules and updates them during the iterations. For more details, one can refer to [22].

3 Proposed Hybrid

In this study, proposed hybrid consists of two phases working in tandem. In phase 1, DEWNN is trained by using a training set. Then, feature selection [1] is performed to identify the salient features. We thoroughly exploited DEWNN [1] by using Gaussian and Morlet wavelet functions separately as activation functions for classification and regression problems. Once DEWNN is trained, its predicted output label along with the actual input features is fed to the phase 2. This results in a modified dataset. In phase 2, rule generation methods like DT, Ripper for classification and CART, DENFIS for regression problems respectively are invoked used to extract rules from the modified dataset. These rules signify the knowledge learnt by the DEWNN. These rules are applied on the validation dataset. The dataflow diagram of the proposed hybrid is depicted in Figure 1. This paper is different from Chauhan et al., [1] as follows: (i) They used DEWNN in solving only classification problems like bankruptcy predication in banks, whereas we solved classification and regression problems. (ii) They used only Gaussian wavelet function as activation function, but we used both Gaussian and Morlet wavelet functions separately. (iii) We extracted rules from DEWNN by employing DT, Ripper and CART, DENFIS for classification problems and regression problems respectively, where they did not extract rules from DEWNN. Thus, DEWNN is a black-box, whereas the proposed hybrid is a transparent system (iv) The experimental setup is different in both studies.

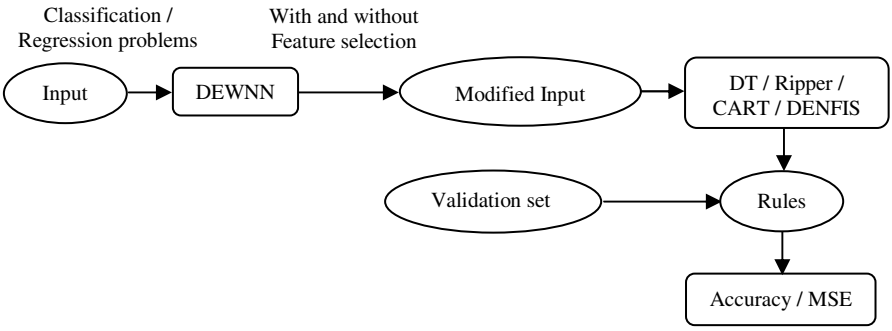


Fig. 1. Dataflow diagram of the proposed hybrid

4 Results and Discussion

The effectiveness of the proposed hybrid is analyzed on classification datasets viz., bank bankruptcy datasets such as Spanish banks [23], Turkish banks [24], UK banks [25] and US banks [26]; benchmark problems such as IRIS [27] and WINE [27] datasets. In addition, we also analyzed regression problems viz., Auto MPG dataset [27], Boston Housing dataset [27], Forest Fires dataset [27] and Body Fat dataset, Pollution dataset (http://lib.stat.cmu.edu).

All dataset are first divided into two parts of 80%: 20% ratio. 10 Fold Cross Validation (10 FCV) was performed on 80% part and 20% of the data, called validation set, is stored for validation purpose. The rules generated using DT, Ripper, CART and DENFIS are tested against the validation set. We used KNIME 2.0 version (http://www.knime.org/) to run J48 (DT) and rule induction algorithm Ripper to extract rules for classification problems. Then, we used CART (http://www.salford-systems.com/) and NEUCOM student version (http://www.kedri.aut.ac.nz/areas-of-expertise/data-mining-and-decision-support-systems/neucom) to run DENFIS to extract rules for the regression problems. The parameters of the DEWNN viz., crossover rate and mutation rate are varied between 0.3 and 0.8, 0.3 and 0.65 respectively. Population size is fixed at 30 and the number of hidden neurons is varied between 3 and 7.

The average results of 10FCV of the proposed hybrid with full features and reduced features datasets using Gaussian and Morlet wavelet activation function for classification problems are presented in Tables 1 & 2 respectively. From Tables 1 & 2, it is observed that the proposed hybrids DEWNN + DT and DEWNN + Ripper

Table 1. Average results of 10FCV of full features datasets for classification problems

Dataset	DEWNN			DEWNN +DT			DEWNN+RIPPER			t-TEST VALUES
Gaussian function										
	Sens	Spec	Acc	Sens	Spec	Acc	Sens	Spec	Acc	
IRIS	-	-	92.99	-	-	91.99	-	-	93.99	2.012
WINE	-	-	92.56	-	-	92.56	-	-	93.99	0.462
SPANISH	86.66	48.56	66.14	64.99	98.57	83.07	68.33	98.57	84.60	0.406
TURKISH	84.16	97.5	90	50	100	75	50	100	75	0
US	93.84	83.07	88.45	97.69	83.07	90.37	94.61	85.38	89.99	1.323
UK	74.99	74.99	74.99	66.66	71.66	69.16	60.83	63.33	65.83	0.518
Morlet function										
IRIS	-	-	93.99	-	-	85.99	-	-	89.32	0.442
WINE	-	-	90.56	-	-	96.56	-	-	95.13	0.734
SPANISH	89.99	54.23	70.76	71.66	100	86.91	71.66	100	86.91	0.001
TURKISH	85	95	90	50	100	75	55	100	77.5	1.5
US	92.23	76.14	83.84	97.69	85.37	91.53	96.92	83.84	90.37	0.324
UK	79.99	72.06	75.83	71.66	69.99	70.83	66.66	71.66	69.16	0.502

performed well against the validation dataset. t-test is performed on the sensitivity between the hybrids and t-statistic value is tabulated in Tables 1 & 2. From the t-statistic values, it is observed that there is no statistically significant difference between the hybrids at 1% level of significance. Further, from the t-test performed separately, it is observed that in majority of the cases in classification problems without feature selection, there is no statistically significant difference between the hybrids and the DEWNN at 1% level of significance. The same is observed in the case of reduced feature set also.

Further, the average mean squared error (MSE) of 10FCV of full features and reduced features datasets using Gaussian and Morlet functions for regression problems is presented in Tables 3 & 4 respectively. From Tables 3 & 4, it is observed that DEWNN+CART and DEWNN+DENFIS performed well on the validation set.

Table 2. Average results of reduced features datasets for classification Problems

Dataset	DEWNN			DEWNN +DT			DEWNN+RIPPER			t-TEST VALUES
Gaussian function										
	Sens	Spec	Acc	Sens	Spec	Acc	Sens	Spec	Acc	
WINE	-	-	96.28	-	-	93.42	-	-	91.90	0.671
SPANISH	93.33	88.56	90.76	83.33	98.57	91.53	83.33	98.57	91.53	0
TURKISH	95	95	95	50	100	75	50	100	75	0
UK	71.66	56.66	64.16	75	53.33	64.16	68.33	51.66	59.99	0.82
Morlet function										
WINE	-	-	92.85	-	-	92.56	-	-	91.42	0.305
SPANISH	88.33	69.995	78.45	73.33	100	87.68	75	100	88.45	0.264
TURKISH	87.5	92.5	90	62.5	100	81.25	52.5	100	76.25	2.057
UK	78.32	63.329	70.83	76.66	66.66	71.66	78.33	65	71.66	0.292

Table 3. Average MSE values of 10FCV of full features datasets for regression problems

Dataset	DEWNN	DEWNN+CART	DEWNN+DENFIS	t-TEST VALUES
Gaussian function				
AUTOMPG	0.0111	0.109873	0.10819	0.491
BODYFAT	0.0056	0.075554	0.06196	1.288
BOSTONHOUSING	0.0179	0.189722	0.12261	1.032
FORESTFIRES	0.009	0.060594	0.08172	0.257
POLLUTION	0.0125	0.116408	0.10526	1.112
Morlet function				
AUTOMPG	0.0161	0.120471	0.1209	0.051
BODYFAT	0.0106	0.104273	0.08386	1.094
BOSTONHOUSING	0.0256	0.199331	0.14498	1.024
FORESTFIRES	0.0089	0.059748	0.09381	0.375
POLLUTION	0.0167	0.11909	0.12551	0.544

t-test is performed on sensitivity between the hybrids and values are presented in Tables 3 & 4. From the t-statistic values, it is observed that there is no statistically significant difference between the hybrids at 1% level of significance. Further, from the t-test performed separately, it is observed that the performance of DEWNN is statistically significant between compared to hybrids at 1% level of significance.

Tables 5 & 6 present the rule base size of the hybrids with full features datasets, reduced features datasets of the classification and regression problems respectively. From Table 5, in the case of full features datasets, it is clearly shown that DEWNN+Ripper achieved less number of rules when Morlet activation function is used in all datasets. In the case of reduced features datasets, from Table 5, the average rule base size of DEWNN+Ripper yielded less number of rules when Gaussian activation function is used in all datasets. From Table 6, in the case of regression

Table 4. Average MSE values of 10FCV of reduced features datasets for regression problems

Dataset	DEWNN	DEWNN+CART	DEWNN+DENFIS	t-TEST VALUES
Gaussian function				
AUTOMPG	0.0512	0.192196	0.19426	0.037
BODYFAT	0.0849	0.222867	0.21294	0.108
BOSTONHOUSING	0.0509	0.192392	0.19028	0.038
FORESTFIRES	0.002	0.028401	0.02865	0.015
POLLUTION	0.024	0.129867	0.1426	0.504
Morlet function				
AUTOMPG	0.0252	0.154617	0.15394	0.061
BODYFAT	0.0237	0.160615	0.1315	2.749
BOSTON HOUSING	0.0323	0.176827	0.17214	0.440
FORESTFIRES	0.0024	0.028749	0.02845	0.016
POLLUTION	0.0367	0.15929	0.16188	0.06

Table 5. Average rules of 10FCV for classification problems

Dataset	Gaussian function in DEWNNN		Morlet function in DEWNNN	
	DEWNN + DT	DEWNN + RIPPER	DEWNN + DT	DEWNN + RIPPER
Full features				
IRIS	4.1	3.4	4.9	3.3
WINE	8.1	4.5	7.1	4.1
SPANISH	4.5	2.5	3.4	2.5
TURKISH	2	2	2.4	2.1
US	4.9	3.1	5.1	2.9
UK	4.5	2.8	5.1	2.4
Reduced features				
WINE	6.3	4.4	5.1	3.8
SPANISH	2.4	2.3	5.1	2.5
TURKISH	2	2	2.4	2.4
UK	3.6	2.1	2.6	2.4

problems, it is observed that DEWNN + DENFIS obtained less number of rules when compared to that of DEWNN + CART no matter which wavelet activation function is used over all datasets. Further, the average rule base size of reduced features datasets from Table 6, it is shown that DEWNN + DENFIS yielded less number of rules, when Morlet activation function is used over Body fat dataset, Boston housing dataset, Forest fires dataset. However, for the Auto MPG and pollution datasets, DEWNN + DENFIS achieved less number of rules when Gaussian activation function is used in DEWNN and DEWNN + CART achieved less number of rule base size, when Morlet activation function is used in DEWNN. From the above discussion, it is concluded that we can use both the hybrids for rule extraction purpose.

Further, from a t-test performed at 1% level of significance separately, it is observed that in majority of the cases, in regression problems, with or without feature selection, DEWNN outperformed the hybrids,

Table 6. Average rules of 10FCV for regression problems

Dataset	Gaussian function		Morlet function	
	DEWNN+CART	DEWNN+DENFIS	DEWNN+CART	DEWNN+DENFIS
Full features				
AUTOMPG	49.4	4.8	43.7	4.8
BODYFAT	19.1	2.9	20.5	2.9
BOSTONHOUSING	56	5	50	5
FORESTFIRES	19.8	9.8	22.1	9.8
POLLUTION	5.1	5.8	4.9	5.8
Reduced features				
AUTOMPG	54.4	2.9	47.8	3.9
BODYFAT	16	3.4	21.3	2.9
BOSTONHOUSING	66	5.6	66	3.3
FORESTFIRES	64.6	6.1	60.8	5.3
POLLUTION	6.6	4.8	4.3	5.8

5 Conclusions

In this paper, we present a hybrid method to extract rules from a trained DEWNN using DT, Ripper to solve classification problems viz., Spanish banks, Turkish banks, UK banks and US banks and iris, wine dataset and CART, DENFIS to solve the regression problems viz., Auto MPG, Body Fat, Boston Housing, Forest Fires and Pollution. The average rule base size obtained by the proposed hybrids turned out to be smaller with feature selection. Further, based on t-test, it is inferred that the hybrids not only make the DEWNN transparent but also are not very far from it in terms of sensitivity in classification problems, whereas DEWNN outscored the hybrids in regression problems in terms of MSE.

References

1. Chauhan, N., Ravi, V., Karthikchandra, D.: Differential evolution trained wavelet neural networks: application to bankruptcy prediction in banks. Expert Systems with Application 36, 7659–7665 (2009)
2. Naveen, N., Ravi, V., Raghavendra Rao, C., Chauhan, N.: Differential evolution trained radial basis function network: application to bankruptcy prediction. International Journal of Bio-Inspired Computation 2(3/4), 222–232 (2010)
3. Andrews, R., Diederich, J., Tickle, A.B.: A survey and critique of techniques for extracting rules from trained artificial neural networks. Know. Based Systems 8(6), 373–389 (1996)
4. Gallant, S.I.: Connectionist expert systems. Communications of the ACM 31(2), 152–169 (1988)
5. Fu, L.M.: Rule generation from neural networks. IEEE Transactions on Systems, Man and Cybernetics 24(8), 1114–1124 (1994)
6. Towlell, G.G., Shavlik, J.W.: The extraction of refined rules from knowledge-based neural networks. Machine Learning 13(1), 71–101 (1993)
7. Arbatli, A.D., Akin, H.L.: Rule extraction from trained neural networks using genetic algorithms. Nonlinear Analysis, Theory, Methods & Applications 30(3), 1639–1648 (1997)
8. Fan, Y., Li, C.-J.: Diagnostic rule extraction from trained feed forward neural networks. Mechanical Systems and Signal Processing 16(6), 1073–1081 (2002)
9. Krishnan, R., Sivakumar, G., Bhattacharya, P.: A search technique for rule extraction from trained neural networks. Pattern Recognition Letters 20, 273–280 (1999)
10. Zhang, X., Qi, J., Zhang, R., Liu, M., Hu, Z., Xue, H.: Prediction of programmed-temperature retention values of naphthas by wavelet neural networks. Computers and Chemistry 25, 25–133 (2001)
11. Avci, E.: An expert system based on wavelet neural network-adaptive norm entropy for scale invariant texture classification. Expert Systems with Applications 32, 919–926 (2007)
12. Lung, S.-Y.: Efficient text independent speaker recognition with wavelet feature selection based multilayered neural network using supervised learning algorithm. Pattern Recognition 40, 3616–3620 (2007)
13. Yu, S.-N., Chen, Y.H.: Electrocardiogram beat classification based on wavelet transformation and probabilistic neural network. Pattern Recognition Letters 28, 1142–1150 (2007)
14. Rajkiran, N., Ravi, V.: Software reliability prediction using wavelet neural networks. In: International Conference on Computational Intelligence and Multimedia Applications, Sivakasi, Tamilnadu, India (2007)
15. Dimoulas, C., Kalliris, G., Papanikolaou, G., Petridis, V., Kalampakas, A.: Bowel-sound pattern analysis using wavelets and neural networks with application to long-term, unsupervised, gastrointestinal motility monitoring. Expert Systems with Applications 34, 26–41 (2008)
16. Dong, L., Xiao, D., Liang, Y., Liu, Y.: Rough set and fuzzy wavelet neural network integrated with least square weighted fusion algorithm based fault diagnosis research for power transformers. Electric Power Systems Research 78, 129–136 (2008)
17. Pan, C., Chen, W., Yun, Y.: Fault diagnostic method of power transformers based on hybrid genetic algorithm evolving wavelet neural network. IET Electric Power Applications 2(1), 71–76 (2008)
18. Vinaykumar, K., Ravi, V., Carr, M., Raj Kiran, N.: Software cost estimation using wavelet neural networks. Journal of Systems and Software 8(11), 1853–1867 (2008)

19. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo (1992)
20. Cohen, W.W.: Fast Effective Rule Induction, From Machine Learning. In: Proceedings of the Twelfth International Conference, ML 1995 (1995)
21. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Wadsworth International Group, Belmont (1984)
22. Kasabov, N., Song, Q.: DENFIS: Dynamic, evolving neural-fuzzy inference systems and its application for time-series prediction. IEEE Transactions on Fuzzy Systems 10, 144–154 (2002)
23. Olmeda, I., Fernandez, E.: Hybrid classifiers for financial multicriteria decision making: The case of bankruptcy prediction. Comp. Economics 10, 317–335 (1997)
24. Canbas, S., Caubak, B., Kilic, S.B.: Prediction of commercial bank failure via multivariate statistical analysis of financial structures: The Turkish case. European Journal of Operational Research 166, 528–546 (2005)
25. Beynon, M.J., Peel, M.J.: Variable Precision Refought Set Theory and Data Discretisation: An Application to Corporate Failure Prediction. Omega 29, 561–576 (2001)
26. Rahimian, E., Singh, S., Thammachote, T., Virmani, R.: Bankruptcy prediction by neural network. In: Trippi, R.R., Turban, E. (eds.) Neural Networks in Finance and Investing, Irwin Professional Publishing, Burr Ridge (1996)
27. Asuncion, A., Newman, D.J.: UCI machine learning repository. University of California, School of Information and Computer Science, Irvine (2007)

A New Improved Self Adaptive Particle Swarm Optimization Technique for Economic Load Dispatch

Kamal Krishna Mandal[1,*], Bidishna Bhattacharya[2],
Bhimsen Tudu[1], and Niladri Chakraborty[1]

[1] Dept. of Power Engineering, Jadavpur University, Salt Lake Campus, Kolkata-700098
kkm567@yahoo.co.in
[2] Dept. of Electrical Engineering, Techno India, Salt Lake, Kolkata-700092

Abstract. This paper presents a new improved self adaptive particle swarm optimization technique to avoid premature convergence for economic load dispatch problem. Many evolutionary techniques such as particle swarm optimization (PSO), differential evolution (DE) have been applied to solve this problem and found to perform in a better way in comparison with conventional optimization methods. But often these methods converge to a sub-optimal solution prematurely. In this method, the inertia weight is made self adaptive depending on the population size and the fitness rank of the particle along with time variant acceleration coefficients. A thirteen-unit test system is considered to demonstrate the effectiveness of the proposed method. The results obtained by the proposed algorithm are compared with other classical as well as modern heuristic techniques. It is found that the proposed method can produced improved results.

1 Introduction

Economic load dispatch is one of important tasks in the power system operation and planning. The main objective of the economic load dispatch (ELD) is to schedule the committed generating unit outputs so as to meet the required load demand at minimum operating cost while satisfying all unit and system equality and inequality constraints. Thus, it is a large-scale highly nonlinear constrained optimization problem. Traditionally, a Lagrangian augmented function is first formulated [1] and the optimal conditions are obtained by partial derivation of this function. Different mathematical as well as classical optimization techniques methods have been applied to solve the ELD problem. These include dynamic programming (DP), quadratic programming (QP), linear programming, homogenous linear programming, nonlinear programming techniques and interior point method [2]–[4].

In the recent year several heuristic techniques such as genetic algorithms (GA) [5], evolutionary programming (EP) [6], artificial neural network (ANN) [7], simulated annealing (SA) [8], technique differential evolution [DE] algorithm [9],

* Corresponding author.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 215–223, 2012.

bio-geography based optimization technique [10]. A chaotic differential evolution based method was presented by Lu et al. [11] for solving dynamic ELD problem and presented promising results.

Particle swarm optimization (PSO) is one of the modern heuristic algorithms and has gained lots of attention in various power system optimization problems. Kennedy and Eberhart [12] originally developed the PSO concept based on the behavior of individuals (i.e. particles or agents) of a swarm or group. Gaing used PSO to solve economic dispatch problem considering generator constraints with line loss [13].

A new improved self adaptive particle swarm optimization technique (NISAPSO) with time-varying acceleration coefficients (TVAC) [14] is applied in this paper to address the problem of premature convergence in ELD problems. In this case, the inertia weight is made self adaptive in terms population size and fitness rank of the particle [15]. The proposed approach was applied on a simple test system to determine its effectiveness. The results have been compared with other methods. It is found that the proposed method is capable to avoid premature convergence and can produce better results.

2 Problem Formulation

The primary objective of ELD problem is to minimize the total fuel cost of the generating units and to meet the system demand under several operating constraints. The fuel cost curve for any unit is assumed to be approximated by segments of quadratic functions of the active power output of the generator for simplicity. Thus, the problem may be described as the minimization of the total fuel cost as defined by (1)

$$FC(P_g) = \sum_{i=1}^{n} \left(a_i P_{gi}^2 + b_i P_{gi} + c_i \right) \tag{1}$$

where, $FC(P_g)$ is total fuel cost of generation in the system (\$/hr), a_i , b_i , c_i are the fuel cost coefficients of the *i* th generating unit, P_{gi} is the power generated by the *i*th unit and *n* is the number of thermal units.

The cost is minimized with the following constraints.

(i) Generator upper and lower limit

The power output of each generating unit must be greater than or equal to the minimum power permitted and also be less than or equal to maximum power permitted on that specified unit. This can be defined as follows.

$$P_{gi}^{\min} \le P_{gi} \le P_{gi}^{\max} \tag{2}$$

where, $P_{gi}^{\min}$ is the minimum power generation by *i*th unit, $P_{gi}^{\max}$ is the maximum power generation by the *i*th unit.

(ii) Power balance

Total power generated by all the units must be equal to power demand plus losses and the same can be expressed as

$$\sum_{i=1}^{n} P_{gi} = P_D + P_L \tag{3}$$

Where, P_D is the total power demand and P_L is the total transmission loss. The transmission loss P_L can be calculated by using B matrix and is defined by (4)

$$P_L = \sum_{i=1}^{n}\sum_{j=1}^{n} P_i B_{ij} P_j \tag{4}$$

where, B_{ij} s are the elements of loss coefficient matrix.

2.1 Classical PSO

Particle Swarm Optimization (PSO) is one of the recent developments in the category of heuristic optimization technique. Kennedy and Eberhart [12] originally developed the PSO concept based on the behaviour of individuals (i.e. particles or agents) of a swarm or group.

Let in a physical *d*-dimensional search space, the position and velocity of the *i*th particle (i.e. *i*th individual in the population of particles) be represented as the vectors $X_i = (x_{i1}, x_{i2}, ..., x_{id})$ and $V_i = (v_{i1}, v_{i2}, ..., v_{id})$ respectively. The previous best position of the *i*th particle is recorded and represented as $pbest_i = (pbest_{i1}, pbest_{i2}, ..., pbest_{id})$. The index of the best particle among all the particles in the group is represented by the $gbest_d$. The modified velocity and position of each particle can be calculated using the current velocity and the distance from $pbest_{id}$ to $gbest_d$ as shown in the following formula:

$$V_{id}^{k+1} = w * V_{id}^{k} + c_1 * rand(\,) * \left(pbest_{id} - X_{id}^{k}\right) + c_2 * rand(\,) * \left(gbest_d - X_{id}^{k}\right) \quad i = 1, 2, \ldots\ldots\ldots\ldots, N_p, \quad d = 1, 2, \ldots\ldots\ldots\ldots, N_g \tag{5}$$

where N_p is the number of particles in a swarm or group, N_g is the number of members or elements in a particle, V_{id}^{k} is the velocity of individual *i* at iteration *k*, *w* is the weight parameter or swarm inertia, c_1 , c_2 are the acceleration constant and $rand(\,)$ is uniform random number in the range [0 1].

The updated velocity can be used to change the position of each particle in the swarm as depicted in (6) as:

$$X_{id}^{k+1} = X_{id}^{k} + V_{id}^{k+1} \tag{6}$$

In general, the inertia weight w the inertia weight w is set according to the following equation:

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \times iter \tag{7}$$

where $iter_{\max}$ is the maximum iteration number and $iter$ is the current iteration number.

2.2 Concept of Self Adaptive Inertia Weight (SAIW)

The inertia weight w is utilized to adjust the influence of the previous velocity on the current velocity, and also to balance between global and local exploration capabilities. The diversity in the population decreases during the end of the optimization procedure. Consequently, the velocities of the particles gradually decrease. To prevent this situation, the utilization of different inertia weights for different particles can play an important role. In Classical PSO, particle with better fitness value is closer to global optimum. For the particle with a better fitness, it is required to impose stronger local exploration ability in order to search its local surrounding region in a better way. On the other side, the particle with inferior fitness, it is required to impose stronger global exploration ability in order to search the entire solution space in a better way.

The swarm size also plays an important role in finding the optimal value. When the swarm size is very small, the PSO algorithm may not be able to search the entire solution space effectively due lack of searching agents. On the other side, very large swarm size may slow down the convergence rate.

In order to address the above issues, inertia weight can be adjusted according to the swarm size and fitness function. This can be implemented as follows [15].

$$w = \left[3 - \exp\left(-p/200\right) + \left(r/100\right)^2\right]^{-1} \tag{8}$$

where p is the population size and r is the fitness rank of the particle.

2.3 Concept of Time-Varying Acceleration Coefficients (TVAC)

It is observed from (5) that the search toward the optimum solution is heavily dependent on the two stochastic acceleration components (i.e. the cognitive component and the social component). Thus, it is very important to control these two components properly in order to get optimum solution efficiently and accurately. It is reported that a relatively higher value of the cognitive component, compared with the social component, results in excessive roaming of individuals through a larger search space. On the other hand, a relatively high value of the social component may lead particles to rush toward a local optimum prematurely [12].

For this, the concept of parameter automation strategy called time varying acceleration coefficients (TVAC) had been introduced [14]. The main purpose of this

concept is to enhance the global search capability during the early part of the optimization process and to promote the particles to converge toward the global optimum at the end of the search. The concept of time varying acceleration coefficients (TVAC) can be introduced mathematically as follows [14].

$$C_1 = \left(C_{1f} - C_{1i}\right)\frac{iter}{iter_{\max}} + C_{1i} \tag{9}$$

$$C_2 = \left(C_{2f} - C_{2i}\right)\frac{iter}{iter_{\max}} + C_{2i} \tag{10}$$

where C_{1i} , C_{1f} , C_{2i} , C_{2f} are constants representing initial and final values of cognitive and social acceleration factors respectively.

2.4 New Improved Self Adaptive Particle Swarm Optimization (NISAPSO) with TVAC

It is seen that the classical PSO is either based on a constant inertia weight factor or a variable inertia factor. It is reported that the particles in classical PSO may converge to a local minimum prematurely due to lack of diversity in the population, particularly for complex problems. The above situation may be avoided and can be implemented as follows.

$$\begin{aligned} V_{id}^{k+1} = w \times V_{id}^{k} &+ \left(\left(C_{1f} - C_{1i}\right)\frac{iter}{iter_{\max}} + C_{1i} \right) \times rand_1 \times \left(pbest_{id} - X_{id}^{k}\right) \\ &+ \left(\left(C_{2f} - C_{2i}\right)\frac{iter}{iter_{\max}} + C_{2i} \right) \times rand_2 \times \left(gbest_{d} - X_{id}^{k}\right) \end{aligned} \tag{11}$$

Here, *w* is varied according to (8) depending on the current population size and fitness of the particle.

3 Results and Discussions

The proposed algorithm based on NISAPSO was implemented using in house Matlab code on 3.0 MHz, 2.0 GB RAM PC. To demonstrate the effectiveness and feasibility of the proposed algorithm, it was applied on a test system consisting of thirteen generating units [16].

The system data including minimum and maximum generation limits and cost coefficients are shown in Table 1. The B-coefficients are taken from [16] and not reproduced here due to space limitation.

Table 1. Unit data for 13-Unit Systems

Unit	$P_{gi}^{\min}$	$P_{gi}^{\max}$	a_i	b_i	c_i
1	60	180	0.00324	7.74	240.00
2	60	180	0.00324	7.74	240.00
3	60	180	0.00324	7.74	240.00
4	60	180	0.00324	7.74	240.00
5	60	180	0.00324	7.74	240.00
6	60	180	0.00324	7.74	240.00
7	40	120	0.00284	8.60	126.00
8	40	120	0.00284	8.60	126.00
9	55	120	0.00284	8.60	126.00
10	55	120	0.00284	8.60	126.00
11	0.0	680	0.00028	8.10	550.00
12	0.0	360	0.00056	8.10	309.00
13	0.0	360	0.00056	8.10	307.00

Table 2. Results of thirteen generator sytem for Economic Load dispatch

Unit	Demand (MW)		
	975	1925	2575
P1 (MW)	88.3722	122.0626	157.1346
P2 (MW)	81.7153	113.5342	175.4975
P3 (MW)	97.4297	114.1215	161.5405
P4 (MW)	74.1713	129.9797	164.8965
P5 (MW)	79.0190	126.8792	172.4661
P6 (MW)	71.4999	127.8593	162.5760
P7 (MW)	55.7958	73.0511	112.2270
P8 (MW)	47.5934	62.5838	93.1694
P9 (MW)	69.1717	83.4575	66.1031
P10 (MW)	61.5209	77.1573	89.8232
P11 (MW)	156.4589	570.0735	680.0000
P12 (MW)	63.1526	269.3442	332.1420
P13 (MW)	35.7168	85.2248	278.9005
Total Generation (MW)	981.6176	1955.3287	2646.4764
Losses (MW)	6.6176	30.3287	71.4764
Fuel Cost ($./hr)	11183.00	19323.00	25234.00
CPU Time (Sec).	7.60	6.68	8.24
Iterations	500	500	500

The performance of PSO based algorithm is quite sensitive to the various parameter settings. Tuning of parameters is essential in all PSO based methods. Based on empirical studies on a number of mathematical benchmark functions [14], it has been reported the best range of variation as 2.5–0.5 for C_1 and 0.5–2.5 for C_2. We

experimented with the same range and the best results were obtained for 2.5 – 1.1 for C_1 and 1.1 – 2.5 for C_2 out of 50 trial runs. The optimization is done with a randomly initialized population of 40 swarms. The maximum iteration was set at 500.

Table 2 shows the results for optimized fuel cost, generation schedule, losses and CPU time for a demand of 975 MW, 1925MW and 2575 MW. The randomness of the proposed method has been verified by testing with same demand for several times.

The convergence characteristic of the proposed algorithm based on NISAPSO for a demand of 1925 MW is shown in Fig.1. Fig.1 also compares the convergence characteristics of NISAPSO with that of PSO. It is clearly seen that proposed method is capable to avoid premature convergence.

The results are also compared with classical as well as neural based methods [16] for the demand of 975 MW, 1925 MW, 2575 MW and are shown in Table 3. It is seen from Table 3 that the proposed method can produce superior results in terms of fuel cost and losses.

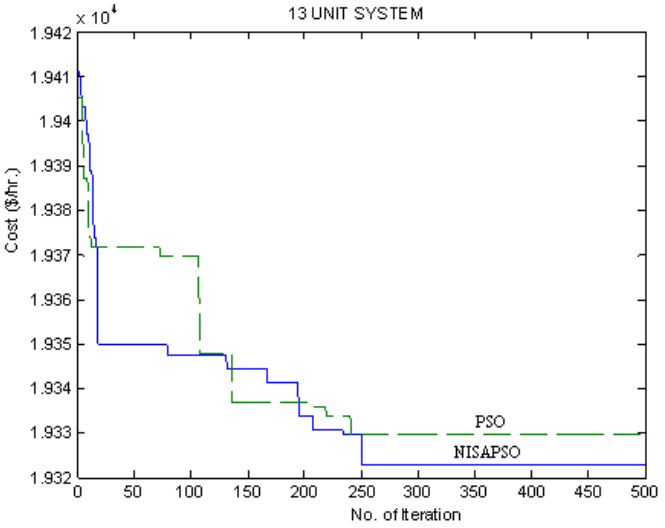


Fig. 1. Convergence characteristics for optimal fuel cost

Table 3. Comparison of results obtained by the proposed method

Demand (MW)		PNM [16]	[CM] [16]	Proposed NISAPSO
975	Fuel Cost ($./hr)	11201.95	11201.90	11183.00
	Losses (MW)	13.4011	13.4009	6.6176
1925	Fuel Cost ($./hr)	19506.92	19506.92	19323.00
	Losses (MW)	60.1509	60.1509	30.3287
2575	Fuel Cost ($./hr)	25514.85	25517.99	25234.00
	Losses (MW)	110.6711	110.6924	71.4764

4 Conclusion

Economic load dispatch (ELD) is one of the important issues in modern day power system operation. The basic objective is to reduce fuel cost while satisfying the several constraints. In this paper, a new improved self adaptive particle swarm optimization technique with time-varying acceleration coefficients have been successfully applied for solving economic load dispatch problem to avoid premature convergence. The inertia weight is made self adaptive in terms of population size and fitness rank of the particle. To evaluate the performance of the proposed algorithm, it has been applied on a test system. The results obtained by the proposed method are compared with other methods. It is found that the proposed method is capable to avoid premature convergence and can produce improved results.

Acknowledgements. We would like to acknowledge and thank Jadavpur University, Kolkata, India for providing all the necessary help to carry out this work. This work is supported by the program University with Potential for Excellence (UPE) – Phase II, sponsored by UGC, Govt. of India.

References

1. Happ, H.H.: Optimal power dispatch - a comprehensive survey. IEEE Trans., PAS-96, 841–854 (1997)
2. Lin, C.E., Viviani, G.L.: Hierarchical economic dispatch for piecewise quadratic cost functions. IEEE Trans. Power App. Syst. PAS-103(6), 1170–1175 (1984)
3. Yang, H.T., Chen, S.L.: Incorporating a multi-criteria decision procedure into the combined dynamic programming/production simulation algorithm for generation expansion planning. IEEE Trans. Power Syst. 4(1), 165–175 (1989)
4. Liang, Z.X., Glover, J.D.: A zoom feature for a programming solution to economic dispatch including transmission losses. IEEE Trans. Power Syst. 3, 544–550 (1992)
5. Walter, D.C., Sheble, G.B.: Genetic algorithm solution of economic dispatch with valve point loading. IEEE Trans. Power Syst. 8, 1325–1332 (1993)
6. Jayabarathi, T., Sadasivam, G., Ramachandran, V.: Evoluionary programming based economic dispatch of generators with prohibited operating zones. Electr. Power Syst. Res. 52, 261–266 (1999)
7. Kumar, J., Seblé, G.B.: Clamped state solution of artificial neural network for real-time economic dispatch. IEEE Trans. Power Syst. 10(2), 925–931 (1995)
8. Wong, K., Fong, C.: Simulated annealing based economic dispatch algorithm. IEEE Proceedings on Generation, Transmission and Distribution 140(6), 509–515 (1993)
9. Coelho, L.D.S., Mariani, V.C.: Improved differential evolution algorithms for handling economic dispatch optimization with generator constraints. Energy Conversion and Management 48, 1631–1639 (2007)
10. Bhattacharya, A., Chattopadhyay, P.K.: Solving complex economic load dispatch problems using biogeography-based optimization. Expert Systems with Applications 37, 3605–3615 (2010)

11. Lu, Y., Zhou, J., Qin, H., Wang, Y., Zhang, Y.: Chaotic differential evolution methods for dynamic economic dispatch with valve-point effects. Engineering Applications of Artificial Intelligence 24, 378–387 (2011)
12. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proc. IEEE Int. Conference on Neural Networks, vol. IV, pp. 1942–1948 (1995)
13. Gaing, Z.L.: Particle swarm optimization to solving the economic dispatch considering the generator constraints. IEEE Transaction on Power Systems 18(3), 1187–1195 (2003)
14. Ratnaweera, A., Halgamuge, S.K., Watson, H.C.: Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans. Evol. Comput. 8(3), 240–255 (2004)
15. Dong, C., Wang, G., Chen, Z.: The inertia weight self adapting in PSO. In: IEEE Proceedings of the 7th World Congress on Intelligent Control and Automation, Chongging, China, June 25-27, pp. 5313–5316 (2008)
16. Aravindhababu, P., Nayar, K.R.: Economic dispatch based on optimal lamda using radial basis function network. Electrical Power & Energy Stystems 24, 551–556 (2002)

Memetic Algorithm Based Task Scheduling Using Probabilistic Local Search

S. Padmavathi, S. MohitGolchha, and A. SeeniMohamed

Department of Computer Science & Engineering,
Thiagarajar College of Engineering,
Madurai, TamilNadu-625015, India
{spmcse,golchhamohit,seeni2010cse}@tce.edu

Abstract. This paper proposes a probabilistic local search based memetic algorithm for the task scheduling problem with the objective to minimize the maximum completion time, which is known to be NP-Hard problem. It has been proven to be NP-Complete for which optimal solutions can be found only after an exhaustive search. The main positive effect of the proposed approach is by choosing only good individuals as initial solutions for Local search thereby assigning an appropriate local search direction to each initial solution. The proposed probabilistic approach is compared with the non probabilistic memetic approach where tabu search act as local search. From these observations, it is found that the minimum local search probability will avoid the premature convergence of MA and also reduce the processing time rather than trapping into a local minima.

1 Introduction

Precedence-constrained parallel applications are one of the most typical application models used in scientific and engineering fields. Such applications can be deployed on homogeneous or heterogeneous cluster computing systems. These applications can be decomposed into a set of smaller tasks that generally have dependencies. Scheduling a set of dependent or independent tasks for parallel execution on a set of processors is an important and computationally complex problem. The objective of the task scheduling problem is to minimize the makespan (schedule length) i.e., the overall computation time of any application represented as Directed Acyclic Graph (DAG). Optimal scheduling of tasks of a DAG onto a set of processors is a strong NP-Hard problem. It has been proven to be NP-Complete for which optimal solutions can be found only after an exhaustive search.

In general, EAs are capable of exploring and exploiting promising regions of the search space. They can, however, take a relatively long time to locate the exact local optimum within the region of convergence. Memetic algorithms (MA) are recent extensions of EAs with the introduction of individual learning as a separate process of local refinement for accelerating search. In MAs, several studies (Hart et al 2004; Ishibuchi et al 2003) have been focused on how to achieve a balance between global and local search part.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 224–231, 2012.

Local search was applied to individuals in every generation. In some studies, (Deb and Goel 2001;Talbi et al 2001) local search was applied to individuals only in the final generation. While Deb and Goel (2001) used local search for decreasing the number of non-dominated solutions i.e for decreasing the diversity of the final solutions. Talbi et al (2001) intended to increase the diversity of final solutions by the application of local search. In many combinatorial optimization problems, Local search can be much more effectively executed than the genetic search. Task scheduling involves finding an optimal mapping of tasks to available machines in the cluster.

MA with Tabu search as local search is proposed which enriches the searching behavior and avoids the premature convergence. In this paper, a local search probability p_{LS} is introduced for decreasing the computation time spent by local search. Local search is not applied to all solutions in the current population but probabilistically applied to selected solutions with the probability p_{LS}. To test the performance of the proposed approach, highly communicating task graphs like Gaussian Elimination is generated and also tested with randomly generated DAGs.

The paper is organized as follows: introduction is followed by the problem modeling which is presented in Sect.2. Section 3. discusses the probabilistic local search based memetic algorithm framework. Section 4 introduces the proposed approach and implementation aspects. Section 5 presents the experimental results and discussions. Finally the conclusions and future research directions are presented in Section 6.

2 Problem Modeling

Parallel programs can be generally represented by a Directed Acyclic graph (DAG). A DAG, *G = (N, E, w, c)* consists of a set N of n nodes and E is a set of communication edges, corresponding to the dependency among tasks. The positive weight *w (n)* associated with node $n \in V$ represents its computation cost and the nonnegative weight *c* (e_{ij}) associated with edge $e_{ij} \in E$ represents its communication cost. The communication cost between two nodes assigned to the same processor is assumed to be zero. The task executions of a given application are assumed to be non-preemptive. In a given task graph, a task without any predecessor is called an *entry task* and task without any child is called an *exit task*. Here, it is assumed that there is one entry and exit task in DAG.

Priority is computed and assigned to each task based on the following attributes namely, Average Computation Cost (ACC), Data transfer cost (DTC) and rank of the predecessor task (RPT).The ACC of a task is the average computation cost on all the 'm' processors and it is computed by using Eqn.(1).

$$ACC(v_i) = \sum_{j=1}^{m} \frac{w_{i,j}}{m} \tag{1}$$

The DTC of a task v_i is the amount of communication cost incurred to transfer the data from task v_i to all its immediate successor task and it is computed at each level

$$DTC(v_i) = \sum_{j=1}^{n} C_{i,j} : i < j \tag{2}$$

using Eqn. (2) where 'n' is the number of nodes in the next level. The DTC value is zero for the exit task. The RPT of a task v_i is the highest rank of all its immediate

predecessortask and it computed using Eqn. (3) and $v_{1,}$ $v_{2,.....}$ v_h are the immediate predecessor of v_i.

$$RPT(v_i) = Max\{rank(v_1), rank(v_2), ...rank(v_h)\} \quad (3)$$

Where The RPT value for the entry task is zero. Here, the maximum rank of predecessor tasks of task v_i as one of the parameters to calculate the rank of the task v_i and the rank computation is given by Eqn (4). At each level, the task with highest rank

$$rank(v_i) = round\{ACC(v_i) + DTC(v_i) + RPT(v_i)\} \quad (4)$$

receives the highest priority followed by task with next highest rank value and so on. The task with minimum ACC value receives the higher priority. A task graph for Gaussian elimination for 3 x 3 matrix is shown in Fig.1 and its computation cost matrix is shown in Table 1.

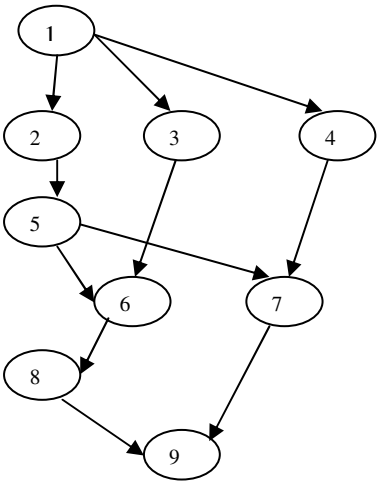


Fig. 1. Gaussian elimination task graph

Table 1. Computational cost matrix (W) for **Fig.1**

Task	*P1*	*P2*	*P3*
1	3	3	3
2	4	5	4
3	4	6	4
4	5	3	5
5	3	7	2
6	3	6	1
7	5	3	6
8	2	4	5
9	5	8	5

Let EST (n_i, p_j) and EFT (n_i, p_j) are the Earliest Start Time and Earliest Finish Time of task n_i on p_j, respectively. For the entry task v_{entry}, EST(v_{entry}, p_j) = 0, and for the other tasks in the graph, the EST and EFT values are computed recursively,

starting from the entry task, as shown in Eqns (6) and (7). In order to compute the EFT of a task n_i, all immediate predecessor tasks of n_i must have been scheduled.

$$EST(n_i, p_j) = \max (avail[j], \max(AFT(n_k + C_{i,k}))) \quad (6)$$

Where $n_k \in pred(n_i)$ i.e is the set of immediate predecessor tasks of task n_i and avail[j] is the earliest time at which processor p_j is ready for task execution.

$$EFT(n_i, p_j) = w_{ij} + EST(n_i, p_j) \quad (7)$$

The inner max block in the EST equation returns the ready time, i.e., the time when all the data needed by n_i has arrived at processor p_j. After a task n_k is scheduled on a processor p_j, the earliest start time and the earliest finish time of n_i on processor p_j is equal to the actual start time $AST(n_k)$ and the actual finish time $AFT(n_k)$ of task n_k, respectively. After all tasks in a graph are scheduled, the schedule length (i.e. the overall completion time) will be the actual finish time of the exit task, $n_{exit.}$

3 Probabilistic Local Search Based Memetic Algorithm

Potential algorithmic improvement in MA can be achieved by considering some important issues of MA (Nyuyen et al 2009; Ong et al 2006; Lozano et al 2004). One of the first issue pertinent to MA design is to consider how often the local search should be applied, i.e., local search frequency. To make local search more efficient, use the problem-specific local search schemes, choose an appropriate solution for a randomly specified local search direction and apply local search only to the selected solutions. We use the local search probability, p_{LS} for decreasing the number of solutions to which local search is applied. The proposed algorithm chooses with respect to the current local search direction.

4 The Proposed Approach

The proposed approach use the local search probability, p_{LS} to select the individuals for the next population thereby limiting the processing time of the task scheduling problem whose pseudocode is presented in Fig.2.

Step 1: Generate Chromosomes based on the population size and calculate their fitness value.
Step 2: Find the best fitness value and have it as the initial solution.
Step 3: Perform local search to a copy of the selected solution using the local search probability over the Chromosomes and find the best fit value. Initial solution is updated with this new best fit value
Step 4: **While (! stopping criteria)**
4.1 Cross over ()
4.2 Mutation ()
4.3 Perform step 3 again from the population obtained from step 4.2
4.4 **if** fitness value < best fit value **then**
4.4.1Update best fit value
end if
end while

Fig. 2. The proposed Memetic Algorithm

4.1 Genetic Representation

A schedule consists of the processor allocation and the start time of each node of the task graph. Each chromosome is represented as a group of genes i.e. task-processor pair (T_i, P_i) indicating that task T_i is assigned to the processor P_i shown in Fig 3. The position of genes in a chromosome represents the order in which tasks should be executed.

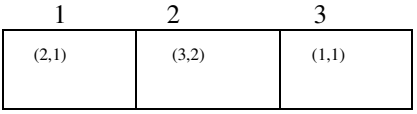


Fig. 3. Chromosomal representation

4.2 Initialization

Here the initial population is generated based on the priority calculation of the tasks at each level as shown in Table 2. As the objective of the task scheduling problem is to find the shortest possible schedule, the fitness of a chromosome is directly related to the length of the associated schedule. Here the fitness value is determined by the earliest finish time of the last task.

4.3 Selection

In this step, the chromosomes in the population are first ranked based on their fitness value for the best to the worst. Then they are selected to the pool.

4.4 Recombination

New chromosomes are created by combining two selected parent chromosomes and swaps second part of each chromosome after a randomly selected point. Single point and two point crossovers are alternatively performed and the crossover probability is also selected randomly.

4.5 Mutation Operator

This operator is applied with a lower probability (about 0.1 or less) than the crossover operator. Its main purpose is to avoid the convergence of the state search to a locally best solution. It takes each chromosome from the fittest ones and changes a randomly selected gene (T_i, P_i) to (T_i, P_j) which introduces diversity each time when it is applied, and consequently the population continues slowly to improve. Therefore the probability of crossover and partial-gene mutation is not fixed in the proposed algorithm.

4.6 Local Search

In the local search part, a neighbor is randomly generated from the neighborhood of the current solution. If the neighbor is better than the current solution, the current solution is replaced and local search continues for the new current solution in the same manner. If the quality of an offspring is very poor, the application of local search seems to be waste of the computation time. Thus local search should be applied to only good offspring.

4.7 Termination Criteria

When no improvement solution has been found over the last n iterations, the algorithm terminates. Typically this value is between 50 to 500 depending on the desired quality of the solution and the size of the problem.

5 Results and Discussions

The proposed approach has been implemented on 32 node HP Proliant cluster. Highly communication intensive application like Gaussian elimination task graph is also generated with matrix size varying from 3 to 15 and the population size is varied from 20 to 200. Although the memetic algorithm is a GA combined with the Tabu Search as a local search, it is not necessarily the case that the genetic parameters are the most ideal for a memetic algorithm. The tasks are selected for an initial pool according to the priority value as shown in Table 2 for the Gaussian Elimination task graph and are selected according to their fitness value.

Table 2. The DTC, ACC, RPT, rank and Priority values for the tasks in **Fig.1**

Level	Task	ACC	RPT	DTC	Rank	Priority
1	1	3	0	6	9	1
2	2	4.33	9	4	17	3
2	3	4.67	9	6	19	2
2	4	4.33	9	7	20	1
3	5	4.0	17	17	38	1
4	6	3.33	38	10	51	2
4	7	4.67	38	11	53	1
5	8	3.67	51	12	66	1
6	9	6	66	0	72	

The effectiveness of the proposed approach is evaluated by varying both the population size and number of iterations is investigated and the results are depicted in Fig.4 and Fig.5. The graph in Fig.5 shows that the processing time for non-probabilistic local search MA spent too much time in convergence whereas the probabilistic local search took less time to converge. The performance of the probabilistic local search based MA is compared with non-probabilistic local search-MA (NPLS-MA) for different population size and different local search probability.

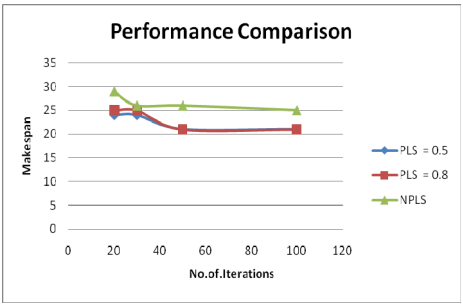


Fig. 4. No.of.Iterations vs. Makespan for 3 x 3 GE DAG for population size=50

Fig. 5. No.of.Iterations vs. Processing time for Random Task Graph

Table 3. The Performance of proposed approach for 3 x 3 Gaussian Elimination task graph

Population Size	Iterations	Makespan			Processing Time		
		p_{LS} =0.5	p_{LS} = 0.8	NPLS-MA	p_{LS} =0.5	p_{LS} = 0.8	NPLS-MA
30	20	26	26	29	0.00	0.00	0.00
	30	24	26	29	0.00	0.01	0.00
	50	24	26	26	0.01	0.02	0.01
	100	24	25	25	0.04	0.07	0.03
50	20	24	25	29	0.00	0.00	0.00
	30	24	25	26	0.00	0.00	0.01
	50	24	21	26	0.00	0.00	0.01
	100	21	21	25	0.04	0.02	0.03
100	20	22	22	26	0.00	0.00	0.00
	30	22	22	26	0.00	0.00	0.00
	50	22	20	26	0.01	0.00	0.0
	100	20	20	24	0.04	0.02	0.04
150	20	22	22	30	0.00	0.00	0.00
	30	22	22	30	0.00	0.01	0.00
	50	20	21	30	0.00	0.02	0.02
	100	20	21	24	0.05	0.06	0.04

From Table 3, the makespan of the proposed approach is converging very slowly when compared to its counterpart. The processing time of the proposed approach with $p_{LS} = 0.5$ is almost similar to NPLS-MA since only 50% of the selected population undergo local search. The processing time of the proposed approach with $p_{LS} = 0.8$, the search is not a random walk because different solutions are chosen as initial solutions for different local search directions. This may take too much of time in selecting the nearest neighbor and thereby increasing the processing time.

When the size of the population is increased, then the processing time of the probabilistic approach is very high when compared to its counterpart NPLS-MA. This is due to the fact that the local search is a random walk to search its neighborhood and doesn't degrade the current solution in the case of Pareto-dominance relation. From these observations, it is found that the minimum local search probabaility will avoid the premature convergence of MA and also reduce the processing time rather than trapping into a local minima.

6 Conclusion

In this approach, only good individuals are selected as initial solutions for local search and assigning appropriate local search directions to each initial solution. The probabilistic approach takes less time to find an optimal solution when the population size is very low and for minimum local search probability.

References

1. Deb, K., Goel, T.: A Hybrid Multi-objective Evolutionary Approach to Engineering Shape Design. In: Zitzler, E., Deb, K., Thiele, L., Coello Coello, C.A., Corne, D.W. (eds.) EMO 2001. LNCS, vol. 1993, pp. 385–399. Springer, Heidelberg (2001)
2. Ishibuchi, H., Yoshida, T., Murata, T.: Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. IEEE Trans. Evol. Comput. 7(2), 204–223 (2003)
3. Lozano, M., Herrera, F., Krasnogor, N., Molina, D.: Real-coded memetic algo-rithms with crossover hill-climbing. Evolutionary Computation 12(3), 273–302 (2004)
4. Nguyen, Q.H., Ong, Y.S., Lim, M.H.: A probabilistic memetic framework. IEEE Transactions on Evolutionary Computation 13(3), 604–623 (2009)
5. Ong, Y.S., Lim, M.H., Zhu, N., Wong, K.W.: Classification of adaptive memetic algorithms: a comparative study. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 36(1), 141–152 (2006)
6. Talbi, E., Rahoual, M., Mabed, M.H., Dhaenens, C.: A hybrid evolutionary approach for multicriteria optimization problems: Application to the flow shop. In: Proc. 1st Inst. Conf. Evolutionay Multi-Criterion Optimization, Zurich, Switzerland, March 7-9, pp. 416–428 (2001)

Neighborhood Search Based Artificial Bee Colony Algorithm for Numerical Function Optimization

Anguluri Rajasekhar[1], Swagatam Das[2], Bijaya Ketan Panigrahi[3], and Manas Kumar Mallick[4]

[1] Dept. of Electrical and Electronics Engineering, NIT Warangal, India
[2] Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata, India
[3] Dept. of Electrical Engineering, IIT Delhi, New Delhi, India
[4] Dept. of Computer Science & Engineering, ITER, Siksha 'O' Anusandhan University, Bhubaneswar, Odisha, India
{rajasekhar.anguluri,swagatam.das}@ieee.org, bkpanigrahi@ee.iitd.ac.in, manasmallick.iter@gmail.com

Abstract. In this paper we investigate about the Neighborhood search mechanisms to improve the performance of Artificial Bee Colony (ABC) on shifted and rotated benchmark functions, proposed in CEC 2005. Although basic version of ABC has been provided with adaptive search mechanism, it will not be able to tackle complex functions with much accuracy unless it was enriched with an efficient neighborhood search scheme. Experimental results have explicitly shown that Neighborhood search based ABC (NS-ABC) performed superiorly well over other variants of ABC.

Keywords: Artificial Bee Colony, Neighborhood search, Global Optimization, Differential Evolution.

1 Introduction

After the revolutionary performances of Evolutionary algorithms based on Darwin theory, swarm intelligent based algorithms came in to field of optimization with even better strategies and good methodologies to solve most difficult and complex problems with less computational burden [1]. Artificial Bee Colony algorithm also comes under the category of swarm intelligent methods and it is on par with that of previous methods like particle swarm optimization [2], Ant colony algorithm [3], Bacterial Foraging Optimization [4] etc. This powerful methodology with simple structure was first proposed by Karaboga [5-7] for numerical function optimization. Since its inception in 2005 ABC was used in many instances and also in very complex optimization problems including multi-objective problems as well [8-9].

The abundant increase in number of algorithms in past decade attracted researchers to develop most complex problem which are almost equivalent to real world optimization problems. CEC 2005 benchmark suite is one of the complex benchmark suite which was proposed in CEC 2005 competition [10]. As the complexity of problem increases algorithm leads to poor performance with basic methodology it has, though

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 232–239, 2012.

it may give promising solutions for normal problems. Artificial Bee Colony has no exception for this and to tackle these kinds of problems we propose a new variant known as Neighborhood search ABC based on Cauchy and Gaussian distributions. The proposed methodology was validated on 25 benchmark functions proposed in CEC 2005 benchmark functions considering the same environmental situations.

The rest of paper is organized as follows; in Section 2 the original ABC is presented. Section deals with NS-ABC method followed by Section 4 emphasizes over the results obtained. Finally in section 5 we unfold some of conclusions and future areas of research.

2 Artificial Bee Colony

According to Karaboga [5] Artificial Bee Colony algorithm falls under the category of bees simulating foraging behavior. It classifies the foraging artificial bees into three kinds, the *employed bees*, *onlooker bees* and *scout* bee. Each plays a vital role in developing ABC algorithm. The detailed optimization procedure of ABC algorithm can be found in the references cited [5-7]. The main idea behind this swarm based methodology is summarized below.

1. Foragers are employed to search the potential food sources which are in the vicinity of their hive.
2. Once a Forager meets the food source then it role changes to Employed bee and then it starts carry local exploitation, based on the profitability it loads nectar and also retains the position of food source.
3. Once Employed bee had done with local exploitation it returns to hive and then shares information with onlooker bees via waggle dances
4. There after onlooker bees chooses on of the employed bee information and then it starts to exploit the food sources. This process is repeated until all the employed bees and workers phase is completed.
5. The food source which is exhausted is abandoned and forager becomes scout and explores new food sources.

3 ABC with Neighborhood Search

Neighborhood search based ABC is a novel variant of ABC algorithm which embeds the mechanism of Neighborhood search. The major problem which hampers the performance of ABC is the exploitation mechanism in classical version. To overcome the problem and to enhance the searching strategy we introduce a neighborhood search based on Cauchy and Gaussian mutation strategies. Neighborhood search based methods showed a very prominent role in improving algorithms performances like Evolutionary Programming and Differential Evolution [11, 12]. Motivated by their performances we put forth a step to incorporate the similar mechanism in Employed and Onlooker phases of ABC algorithm. The steps involved are discussed in subsections.

3.1 Initialization Phase

The preliminary parameters algorithm (which is same to ABC also) are no. of . Food sources (FS) which is equal to number of employed bees or onlooker bees, *limit* is the parameter after which food source is determined to abandoned or not based on its usage.

The i_{th} food source of a population for a D-dimensional problem is represented by $X_i = \{x_{i1}, x_{i2}, ..., x_{iD}\}$, and then each food source is generated by

$$x_{ij} = lb_j + r * (ub_j - lb_j) \tag{1}$$

where $j = 1,2,3...., D$ and $i = 1,2,3,...., FS,$; r is the random number in range of [0 1]; lb_j and ub_j are upper and lower bounds for the dimension j, respectively.

3.2 Employed Bee Phase

Once a Forager discovers a food source it turns in to an employed bee and there after it tries to exploit the food source. In this NS-ABC this local exploitation is carried out using DE *'rand/1/bin'* mutation strategy, according to Eq (2)

$$v_i = x_{r1} + \begin{cases} (x_{r2} - x_{r3}) \cdot N(0.5,0.5), & \text{if } U(0,1) < 0.5 \\ (x_{r2} - x_{r3}) \cdot \delta & \text{otherwise} \end{cases} \tag{2}$$

where *r1, r2, r3* denote a randomly selected individual solutions, $N(0.5,0.5)$ denotes a Gaussian random number with mean 0.5 and standard deviation 0.5, δ denotes a Cauchy random variable with scale parameter t=1.

Conversely to ABC in which only on single dimension of trail solution is modified, in this NS-ABC the number of dimensions to be perturbed are determined by crossover operation with help of *CR* parameter and is given by

$$u_{i,j} = \begin{cases} v_{i,j} & \text{if } (rand_j(0,1) \le CR \,\|\, j == j_{r4}) \\ x_{i,j} & \text{otherwise} \end{cases} \tag{3}$$

where j_{r4} denotes a random dimension within in the i_{th} food source. There after a greedy mechanism is applied between u_i and x_i, then the better solution parameters are retained discarding the rest and this process is repeated till all the employed bees carries an exploitation task.

3.3 Onlooker Bee Phase

After the completion of employed bee phase, bees enter in to onlooker phase, at this stage onlooker bee evaluates the nectar information taken from all of employed bees and then selects a food source x_i depending on its probability value p_i which is given by:

$$p_i = \frac{fit_i}{\sum_{k=1}^{FS} fit_k} \tag{4}$$

FS is total number of food sources. Fitness value fit_i is calculated by using following equation

$$fit_i = \frac{1}{1 + f(X_i)} \tag{5}$$

Here $f(X_i)$ is the objective function to be minimized. The onlooker exploits its food source in the region X_i. To have a good exploitation mechanism in this phase the new food source is produced by *'current-to-best/1/bin'* like DE strategy as mutation operator and is given as follows

$$v_i = x_{r1} + \begin{cases} (x_{best} - x_i) + (x_{r2} - x_{r3}) \cdot N(0.5,0.5), & \text{if } U(0,1) < 0.5 \\ (x_{best} - x_i) + (x_{r2} - x_{r3}) \cdot \delta & \text{otherwise} \end{cases} \tag{6}$$

The crossover operation is performed similarly to that of employed bee phase and greedy mechanism is performed between obtained new v_i and x_i. This is repeated until all the onlooker bees complete the exploitation task.

3.4 Scout Bee Phase

Each bee will search for a better food source for a certain number of cycles (*limit*) and if the fitness value doesn't improve, then that particular bee becomes a *scout* bee. A food source is initialized to that *scout bee* randomly and the search process continues till it meets a termination criteria.

4 Experimental Results

To validate the performance of proposed NS-ABC we had considered a total of 25 benchmark functions proposed in CEC 2005 competition with considering the same experimental constraints. We had also compared NS-ABC with a new variant of ABC [13]. The experiments are carried out in an Intel P4 processor machine with 2GB RAM using MATLAB 7.9.

4.1 Parametric Setup of NS-ABC

The parameters involved in the NS-ABC are similar to that of ABC and Table 1 summarizes the list of parameters.

Table 1. Algorithmic Parameters of NS-ABC

Parameters	**Value**
Population size	20
Employed Bees (n_e)	50 % of colony
Onlooker Bees (n_o)	50 % of colony
CR	0.1
Limit	200

4.2 Results

To make a fair comparison we had considered the same experimental set up which were suggested in CEC 2005 competition. A test suite of 25 test functions of 30-dimensions are being considered for comparison in terms of mean and STD.

Table 2. Comparsion of NS-ABC & Modified ABC in terms of Mean and Std deviation

	F_1	F_2	F_3	F_4	F_5
NS-ABC	0.00e+00 (0.00e+00)	0.00e+00 (0.00e+00)	**1.55e+05 (2.14e+04)**	**1.02e-04 (6.28e-05)**	**3.40e+01 (3.31e-01)**
Modified ABC	0.00e+00 (0.00e+00)	0.00e+00 (0.00e+00)	2.20e+05 (2.53e+04)	2.20e+05 (2.53e+04)	6.02e+03 (7.16e+02)
	F_6	F_7	F_8	F_9	F_{10}
NS-ABC	**0.00e+00 (0.00e+00)**	**9.36e-03 (2.06e-03)**	**2.04e+01 (4.50e-01)**	**0.00e+00 (0.00e+00)**	**5.07e+01 (1.40e+00)**
Modified ABC	1.38e+02 (5.81e+01)	1.05e+02 (8.20e+01)	2.09e+01 (5.63e-02)	6.60e+01 (7.74e+00)	2.01e+02 (1.44e+01)
	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
NS-ABC	**2.26e+01 (1.10e+00)**	**4.23e+03 (1.50e+03)**	**2.72e+00 (3.45e-01)**	**1.33e+01 (3.85e-01)**	**1.00e+02 (2.08e+01)**
Modified ABC	3.56e+01 (8.84e-01)	9.55e+04 (1.75e+04)	1.07e+01 (9.32e-01)	1.36e+01 (1.33e+01)	2.88e+02 (3.25e+01)
	F_{16}	F_{17}	F_{18}	F_{19}	F_{20}
NS-ABC	**2.07e+02 (5.49e+00)**	**2.87e+02 (4.81e+00)**	**7.88e+02 (1.84e+01)**	**7.81e+02 (3.63e+01)**	**7.85e+02 (2.98e+01)**
Modified ABC	3.06e+02 (2.18e+01)	3.01e+02 (2.00e+01)	8.12e+02 (4.07e+01)	8.17e+02 (4.50e+01)	8.23e+02 (5.31e+01)
	F_{21}	F_{22}	F_{23}	F_{24}	F_{25}
NS-ABC	**5.06e+02 (1.73e+00)**	**8.00e+02 (3.94e+00)**	**8.12e+02 (2.32e+00)**	**2.00e+02 (1.02e-01)**	2.00e+02 (3.88e-01)
Modified ABC	6.42e+02 (1.41e+02)	9.04e+02 (6.01e+00)	8.20e+02 (8.13e+01)	2.01e+02 (5.45e-02)	**2.00e+02 (2.23e-03)**

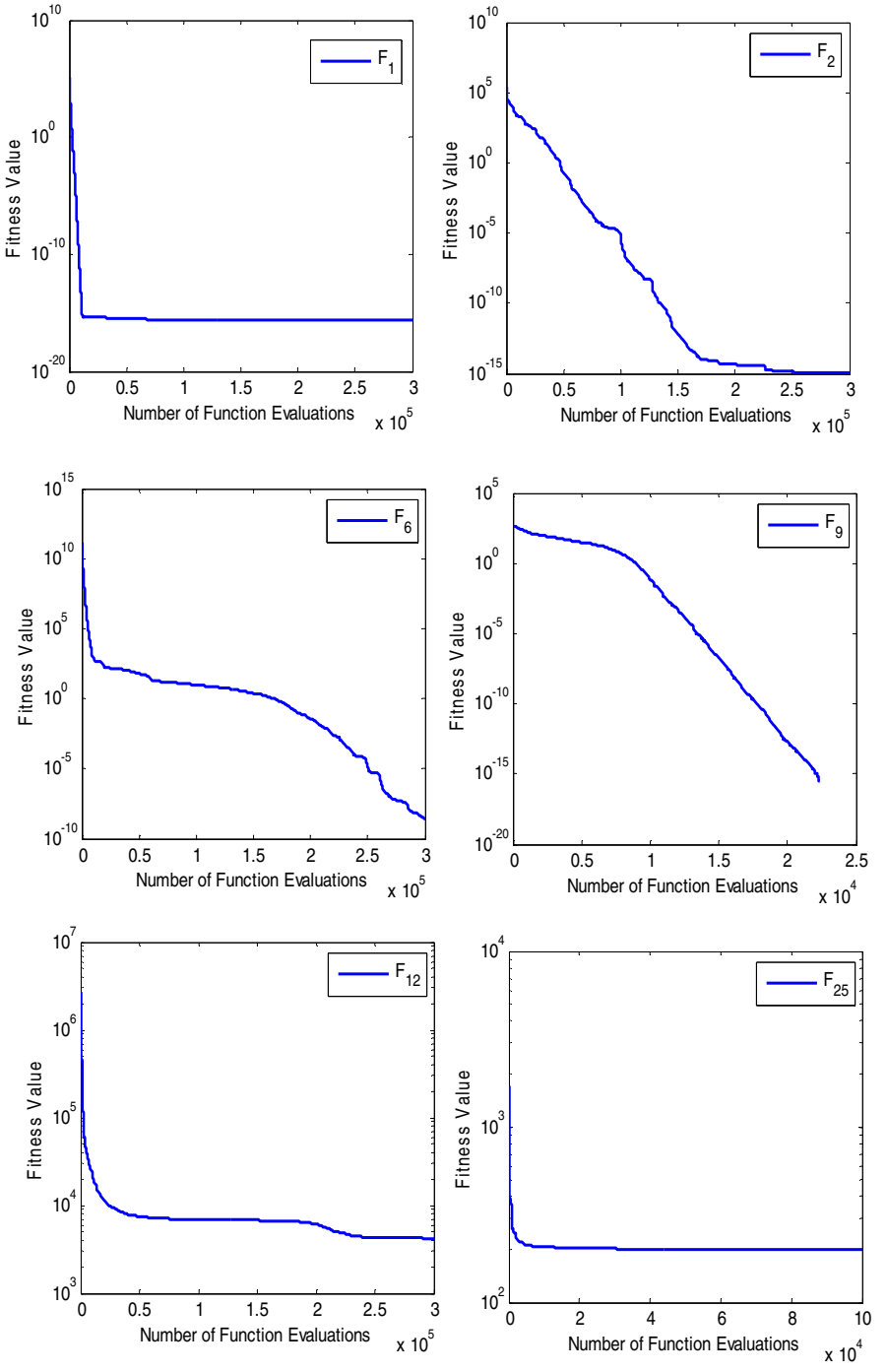


Fig. 1. Convergence of NS-ABC on selected functions

A total of 3.00e+05 NFEs and a total of 25 test runs are kept as termination criterion. Table 2 summarizes the obtained results of NS-ABC and also a new variant of ABC known as modified ABC [13]. While recording the results if the obtained value is less than 10^{-7}, this value is considered to be 0 and there is no practical significance if less than that. The best value is been highlighted in bold and also convergence plots are being provided in Fig 1.

From the Table 2 it is very clear that NS-ABC had outperformed modified ABC on 22 functions which clearly shows the potential of NS-ABC in solving complex functions.

5 Conclusions

A new strategy based on Neighborhood search strategy has been suggested of ABC algorithm for solving complex optimization problems. The proposed method had performed exceptionally well with a good convergence rate on more than 20 functions when compared with a new variant of ABC, which shows the superiority of proposed approach. Our future scope will be focusing on solving a Large scale optimization problems.

References

1. Kennedy, J., Eberhert, R.C., Shi, Y.: Swarm Intelligence. Morgan Kaufmann Series in Evolutionary Computation, San Fransisco (2001)
2. Kennedy, J., Eberhert, R.C.: Particle swarm optimization. In: IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948 (1995)
3. Dorigo, M., Gambardella, L.M.: Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans. on Evolutionary Computation 1(1), 53–66 (1997)
4. Passino, K.M.: Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Systems 22(3), 52–67 (2002)
5. Karaboga, D.: A idea based on Bee Swarm for Numerical Optimization, Technical Report, TR-06, Erciyes University Engineering Faculty, Computer Engineering Department (2005)
6. Karaboga, D., Basturk, B.: A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) algorithm. Journal of Global Optimization 39, 459–471 (2007)
7. Karaboga, D., Basturk, B.: On the performance of artificial bee colony (ABC) algorithm. Applied Soft Computing 8, 687–697 (2008)
8. Rajasekhar, A., Abraham, A., Pant, M.: Levy mutated Artificial Bee Colony algorithm for global optimization. In: IEEE International Conference on Systems, Man and Cybernetics (SMC), pp. 655–662 (2011)
9. Akbari, R., Hedayatzadeh, R., Ziarati, K., Hassanizadeh, B.: A multi-objective artificial bee colony algorithm. Swarm and Evolutionary Computation 2, 39–52 (2012)
10. Suganthan, P.N., Hansen, N., Liang, J.J., Deb, K., Chen, Y.-P., Auger, A., Tiwari, S.: Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization, Technical Report, Nanyang Technological University, Singapore (May 2005)

11. Yao, X., Liu, Y., Lin, G.: Evolutionary programming made faster. IEEE Trans on Evolutionary Computation 3(2), 82–102 (1999)
12. Yang, Z., Tang, K., Yao, X.: Large scale evolutionary optimization using cooperative coevolution. Information Sciences 179(15), 2985–2999 (2008)
13. Akay, B., Karaboga, D.: A modified Artificial Bee Colony algorithm for real-parameter optimization. Information Sciences 192, 20–142 (2012)

Path Generation and Obstacle Avoidance of an Autonomous Mobile Robot Using Intelligent Hybrid Controller

Prases Kumar Mohanty and Dayal R. Parhi

Robotics Laboratory,Department of Mechanical Engineering ,
National Institute of Technology, Rourkela
Odisha, India
pkmohanty30@gmail.com, dayalparhi@yahoo.com

Abstract. Intelligent soft computing techniques such as artificial neural network and fuzzy logic approaches are verified to be efficient and appropriate when implemented to a variety of systems. Recent years both techniques has been rising interest and as a result Neuro-Fuzzy techniques have been developed. It has also taken the advantages of neural network and fuzzy inference system. This paper presents the application of an adaptive neuro-fuzzy inference system (ANFIS) to path generation and obstacle avoidance of an autonomous mobile robot in an unknown static and dynamic environment. In this architecture different sensor based information such as front obstacle distance (FOD), left obstacle distance (LOD), right obstacle distance (ROD) and target angle (TA) are given as input to the adaptive controller and output from the controller is steering angle of the mobile robot. Outcome from the simulation results using MATLAB demonstrated that the ANFIS model could be used as a suitable and effective technique to navigate the mobile robot safely both in static and dynamic environment, find and reach to target objects.

Keywords: ANFIS, Mobile robot, navigation.

1 Introduction

Mobile robots are extensively used in various fields of engineering such as aerospace research, nuclear industry, mining industries and also in military operations. Path planning of an autonomous mobile robot in an unknown cluttered environment is the most important aspects in a mobile robots research area. But the mobile robot should have learning capability to navigate safely in complex, unknown environment. In path planning, the objective of the mobile robot is to travel safely from source to target without hitting with obstacles that present in environment. In this paper an intelligent hybrid technique i.e. ANFIS (Adaptive Neuro-Fuzzy Inference System) is implemented for the path planning of a mobile robot. This technique gives the advantages of both fuzzy logic and neural network. Many researchers have been developed several efficient hybrid techniques in the navigation of mobile robots.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 240–247, 2012.

Among these hybrid intelligent techniques, the adaptive Neuro-fuzzy system can learn only from its environment and then make decisions. A neural fuzzy system is based on fuzzy inference system, which is trained by a learning algorithm derived from neural network theory.

In literature survey, there can be found different approaches associated with motion control of the autonomous mobile robot. Sudhagar et al. [1] have presented ANFIS based navigation for truck like mobile robot.They have trained the truck with three inputs, five-layer neural network with back propagation learning algorithm for the control the movement of a mobile robot. Neuro-fuzzy motion planners for intelligent robots have been discussed by Tsoukalas et al. [2]. The robot is supported with sensors to acquire local environmental input and a neuro-fuzzy model processing predictable aspects of its environment. The method is verified through a robot learning to navigate from start point to goal point without colliding with obstacles present in its path. Navigation of multiple mobile robots using Neuro-fuzzy technique has been addressed by pradhan et al. [3]. In this design, output from the neural network given as input to the fuzzy controller to navigate the mobile robot successfully in the clutter environment. Experimental verifications also have been done with the simulation results to prove the validity of the developed technique. Navigation of mobile robots using adaptive neural-fuzzy system have discussed by Nefti et al. [4]. Different sensor based information they have given to the Sugeno–Takagi fuzzy controller and output from the controller is the robot orientation. Experimental results settle the importance of the methodology when dealing with navigation of a mobile robot in unknown or partially unknown environment. To determine collision-free path of mobile robot navigating in a dynamic environment using Neuro-fuzzy technique has been presented by Hui et al. [5]. The performances of Neuro-fuzzy approaches are compared other approaches (GA, Mamdani) and it is found that Neuro-fuzzy approaches are found to perform better than the other approaches. Control of mobile robot based on Neuro-fuzzy technique has been discussed by Godjevac and Steele [6]. In this paper they have shown how Neuro-fuzzy controllers can be achieved using a controller based on the Takagi-Sugeno design and a radial basis function neural network for its implementation. Neuro-fuzzy based mobile robot navigation has been presented by Rusu et al. [7]. In this paper they discussed a Neuro-fuzzy controller for sensor based navigation in indoor environments. A Neuro-Fuzzy Controller for mobile robot navigation has been addressed by Kim and Trivedi [8]. In this paper they have applied Neural integrated fuzzy controller to control the robot motion—steering angle, heading direction, and speed. Navigation of an autonomous mobile robot using neuro-fuzzy approach has been design by Crestani et al. [9]. They have developed a fuzzy-neural network based technique that considers the direction velocity of navigation as controllable terms.

The objective of the study in this paper is to open up a path planning layout for a mobile robot to navigate safely towards the destination while avoiding various obstacles present in the environment. Hybrid learning algorithm which combines the gradient descent method and least square estimate (LSE) is used to adjust the parameters in ANFIS.

2 Kinematic Analysis of Mobile Robot

The Kinematic analysis of differentially steered wheeled mobile robots in a two-dimensional plane can be done in one of two ways: either by Cartesian or polar coordinates. It is assumed that the mobile robot moves without slipping on a plane, i.e., there is a pure rolling contact between the wheels and the ground and also there is no lateral slip between the wheel and the plane. The modeling in Cartesian coordinates is the most common use and the discussion will be limited to modeling in Cartesian coordinates. The robot has two fixed standard wheels and one caster wheel and is differentially driven by skid steer motion. The two driving wheels are independently driven by two motors to acquire the motion and orientation. Both the wheels have same diameter '2r' (Fig.1). The driving wheels are separated by distance 'W'.The position of the robot in the 2-D plane at any instant is defined by the situation in Cartesian coordinates and the heading with respect to a global frame of reference. The tangential velocity of mobile robot is given by [11].

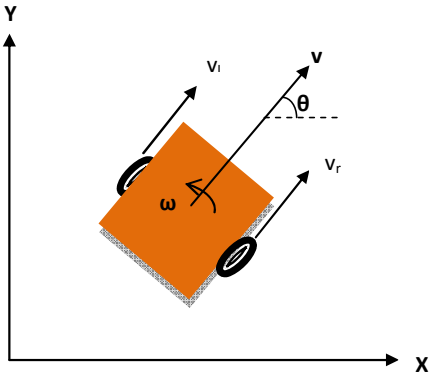


Fig. 1. Mobile Robot kinematic parameters

$$v_t = \frac{1}{2}(v_r + v_l) \tag{2.1}$$

$$\omega_t = \frac{1}{W}(v_r - v_l) \tag{2.2}$$

$$v_r = r\omega_r \quad \text{and} \quad v_l = r\omega_l \tag{2.3}$$

Where v is the linear velocity and ω is the angular velocity of the mobile robot. Suffixes r,l,t stand for right wheel, left wheel and tangential directions of motion respectively.

3 Architecture of the Adaptive Neuro-Fuzzy Inference System (ANFIS) for Current Analysis

Adaptive Network-Based Fuzzy Inference System (ANFIS) is one of hybrid intelligent neuro-fuzzy inference system and it functioning under Takagi-Sugeno-type fuzzy inference system, which was developed by Jang [1993]. ANFIS has a similar

structure to a multilayer feed forward neural network but the links in an ANFIS only indicate the flow direction of signals between nodes and no weights are associated with the links.

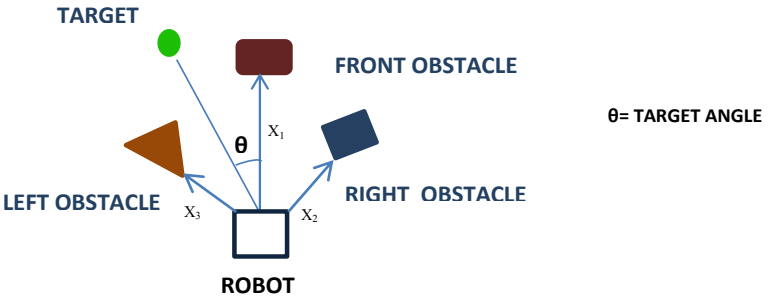


Fig. 2. Schematic Diagram for Robot Initial Position

As for the prediction of steering angle for mobile robot we assume the fuzzy inference system under consideration of four inputs i.e Front obstacle distance (X_1), Right obstacle distance (X_2), Left obstacle distance(X_3) , target angle(X_4) and each input variable has three membership functions(MF) A_1,A_2,andA_3,B_1,$B_2$and B_3,C_1,C_2andC_3, and D_1,D_2andD_3 respectively, then a Takagi-Sugeno-type fuzzy inference system if-then rules is set up as follows;

Rule: if X_1 is A_i and X_2 is B_i and X_3 is C_i and X_4 is D_i, then
f_n(steering Angle) = $p_nX_1+q_nX_2+r_nX_3+s_nX_4+u_n$

where,i=1,2,3 and p_n,q_n,r_n,s_n and u_n are the linear parameters of function fn. In ANFIS architecture nodes of the same layer have similar functions. The output signals from previous layers are the input to the next layer.

The function of each layer in ANFIS is discussed as follows;

Input Layer: In this layer nodes simply pass the incoming signal to layer-1. That is

$$\left.\begin{aligned} O_{0,FOD} &= X_1 \\ O_{0,ROD} &= X_2 \\ O_{0,LOD} &= X_3 \\ O_{0,TA} &= X_4 \end{aligned}\right\} \quad (3.1)$$

First Layer: This layer is the fuzzification layer. Neurons in this layer complete fuzzification process. Every node in this layer is an adaptive node and computing the membership function value. The output of nodes in this layer are presented as

$$\left.\begin{aligned} O_{1,i} &= \mu_{A_i}(X_1) \\ O_{1,i} &= \mu_{B_i}(X_2) \\ O_{1,i} &= \mu_{C_i}(X_3) \\ O_{1,i} &= \mu_{D_i}(X_4) \end{aligned}\right\} \quad (3.2)$$

i=1,2,3

Here $O_{1,i}$ is the bell shape membership grade of a fuzzy set S (A_i , B_i ,C_i and D_i) and it computes the degree to which the given inputs (X_1,X_2,$X_3$and X_4) satisfies the quantifier S. Membership functions defined as follows;

$$\mu_{A_i}(x) = \frac{1}{1+\left[\left(\frac{X_1-C_i}{a_i}\right)^2\right]^{b_i}} \quad 3.2(i)$$

$$\mu_{B_i}(x) = \frac{1}{1+\left[\left(\frac{X_2-C_i}{a_i}\right)^2\right]^{b_i}} \quad 3.2(ii)$$

$$\mu_{C_i}(x) = \frac{1}{1+\left[\left(\frac{X_3-C_i}{a_i}\right)^2\right]^{b_i}} \quad 3.2(iii)$$

$$\mu_{D_i}(x) = \frac{1}{1+\left[\left(\frac{X_4-C_i}{a_i}\right)^2\right]^{b_i}} \quad 3.2(iv)$$

a_i , b_i and c_i are parameters that control the Centre, width and slope of the Bell-shaped function of node 'i' respectively. These are also known as premise parameters.

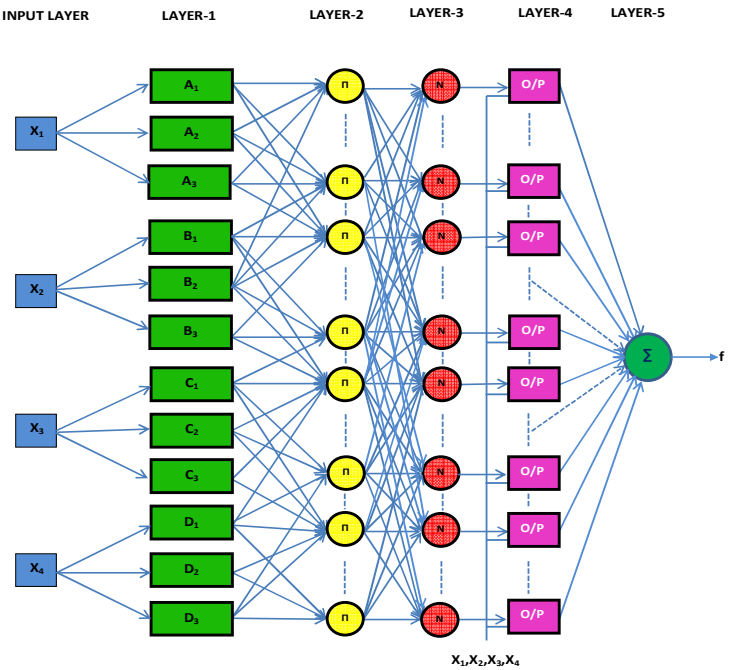


Fig. 3. Schematic Diagram for ANFIS Structure

Second Layer: It is also known as rule layer. Every node in this layer is a fixed node and labeled as π_n. Each node in this layer corresponds to a single Sugeno-Takagi fuzzy rule. A rule node receives inputs from the respective nodes of layer-2 and determines the firing strength of the each rule. Output from each node is the product of all incoming signals.

$$O_{2,n} = w_n = \mu_{A_i}(X_1) \cdot \mu_{B_i}(X_2) \cdot \mu_{C_i}(X_3) \cdot \mu_{D_i}(X_4) \tag{3.3}$$

Where 'W_n' represents the firing strength or the truth value, of n^{th} rule and n=1, 2, 3…81 is the number of Sugeno-Takagi fuzzy rules.

Third Layer: It is the normalization layer. Every node in this layer is a fixed node and labeled as N_n. Each node in this layer receives inputs from all nodes in the fuzzy rule layer and determines the normalized firing strength of a given rule. The normalized firing strength of the nth node of the n^{th} rule's firing strength to sum of all rules's firing strength.

$$O_{3,n} = \bar{w}_n = \frac{\bar{w}}{\sum_{n=1}^{81} w_n} \tag{3.4}$$

The number of nodes in this layer is the same the number of nodes in the previous layer that is 81 nodes. The output of this layer is called normalized firing strength.

Fourth Layer: Every node in this layer is an adaptive node. Each node in this layer is connected to the corresponding normalization node, and also receives initial inputs X_1, X_2, X_3 and X_4. A defuzzification node determines the weighted consequent value of a given rule define as,

$$O_{4,n} = \bar{w}_n f_n = \bar{w}_n[p_n(X_1) + q_n(X_2) + r_n(X_3) + s_n(X_4) + u_n] \tag{3.5}$$

Where $\bar{w}_n$ is a normalized firing strength from layer-2 and [p_n, q_n, r_n, s_n, u_n] are the parameters set of this node. These parameters are also called consequent parameters.

Fifth Layer: It is represented by a single summation node. This single node is a fixed node and labeled as $\sum$. This node determines the sum of outputs of all defuzzification nodes and gives the overall system output.

$$O_{5,1} = \sum_{n=1}^{81} \bar{w} f_n = \frac{\sum_{n=1}^{81} w_n f_n}{\sum_{n=1}^{81} w_n} \tag{3.6}$$

4 Simulation Results

The simulation results are obtained by using MATLAB software to show the performance and validity of the developed method in various environmental conditions. When a robot is closed to an obstacle, it avoids a hitting by moving away from it in opposite direction. When the data values from sensors are less than the threshold values then the obstacle avoidance behavior is activated. An advantage of ANFIS is to control velocity and direction of robot, if there are no obstacles on the robot path then it moves quickly towards the goal. Fig.4 shows the ability of the robot navigates safely and finds the target successfully in cluttered environment.Fig.5 shows avoidance behavior of a single mobile robot. In Fig.6. it can be noted that the robots stay well away from the obstacles as well as from each other. The simulation results also compared with fuzzy logic and neural network technique and it is verified that using the ANFIS controller the robot reached to the specified target in less time (Table-1).

Table 1.

Sl.No.	Figure	Fuzzy	NN	ANFIS
		Total travel time in Sec.		
1	4	13.6	12.1	11.2
2	5	10.9	10.1	8.9
3	6	17.5	16.4	14.9

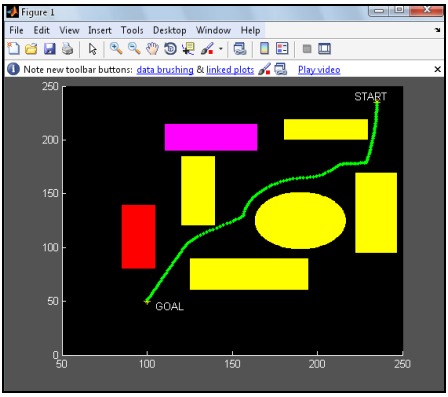


Fig. 4. Navigation path for one mobile robot to reach target using ANFIS controller

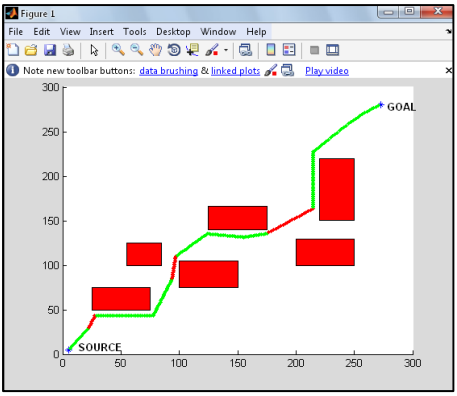


Fig. 5. Environment for one robot and one target using ANFIS controller

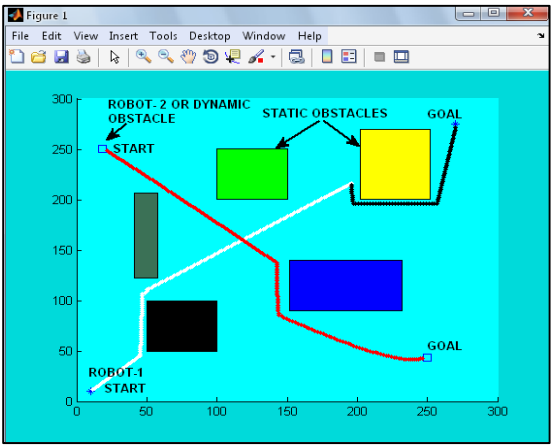


Fig. 6. Collision avoidance between two mobile robots using ANFIS

5 Concluding Remarks

In this paper, an adaptive neuro-fuzzy technique has been applied for navigation of an autonomous mobile robot in an unknown cluttered environment. The simulation experiments using Matlab in different environmental scenarios showing better performance to avoiding static as well as dynamic obstacles present on the robot path. The resulting architecture has also adaptable to any kinds of complex environment. The proposed method also compared with other intelligent techniques and the settlement in results show the efficiency of the technique. The future research is planned to implement the proposed design to multiple robots navigation instead of a single robot.

References

1. Sudhakar, K., Noorul, A., Selvaraj: Neuro-Fuzzy Based navigation for Truck like Mobile Robot. International Journal of Soft Computing 5, 633–637 (2007)
2. Tsoukalas, L.H., Houstis, E.N., Jones, G.V.: Neuro-fuzzy Motion Planners for Intelligent Robots. Journal of Intelligent and Robotic System 19, 339–356 (1997)
3. Pradhan, S.K., Parhi, D.R., Panda, A.K.: Neuro-fuzzy technique for navigation of multiple mobile robots. Fuzzy Optimum Decision Making 5, 255–288 (2006)
4. Nefti, S., Oussalah, M., Djouani, K., Pontnau, J.: Intelligent Adaptive Mobile Robot Navigation. Journal of Intelligent and Robotic System 30, 311–329 (2001)
5. Hui, N.B., Mahendar, V., Pratihar, D.K.: Time-optimal, collision-free navigation of a car-like mobile robot using neuro-fuzzy approaches. Fuzzy Sets and Systems 157, 2171–2204 (2006)
6. Godjevac, J., Steele, N.: Neuro-fuzzy control of a mobile robot. Neuro-computing 28, 127–142 (1999)
7. Rusu, P., Petriu, E.M., Whalen, T.E., Cornell, A., Spoelder, J.W.: Behavior-Based Neuro-Fuzzy Controller for Mobile Robot Navigation. IEEE Transactions on Instrumentation and Measurement 52, 1335–1340 (2003)
8. Ng, K.C., Trivedi, M.M.: A Neuro-Fuzzy Controller for Mobile Robot Navigation and Multi robot Convoying. IEEE Transactions on Systems, Man, and Cybernetics—part B: Cybernetics 28, 829–840 (1998)
9. Crestani, P., Fernando, R.J., Von, Z.A.: Hierarchical neuro-fuzzy approach to autonomous navigation. In: Proceedings of the International Joint Conference on Neural Networks, Honolulu, USA, pp. 2339–2344 (2002)
10. Jang, J.S.R.: Adaptive Network based Fuzzy Inference System. IEEE Transactions on Systems, SMC 23(3), 665–685 (1993)
11. Parhi, D.R., Singh, M.: Navigational Path Analysis of Mobile Robots using an Adaptive Neuro-fuzzy Inference System Controller in a Dynamic Environment. In: Proc. IMechE., vol. 224, pp. 1369–1381 (2009)
12. Parhi, D.R.: Navigation of Multiple Mobile Robots in an unknown Environment, Doctoral Thesis, Cardiff School of Engineering, University of Wales, UK (2000)

An Adaptive Memetic Algorithm for Multi-robot Path-Planning

Pratyusha Rakshit, Dhrubojyoti Banerjee, Amit Konar, and Ramadoss Janarthanan

ETCE Depratment, Jadavpur University, India
pratyushar1@gmail.com,
{dhrubo_jyoti_banerjee,konaramit}@yahoo.co.in,
srmjana_73@yahoo.com

Abstract. This paper provides a novel approach to design an adaptive memetic algorithm by utilizing the composite benefits of Differential Evolution for global search and Q-learning for local refinement. The performance of the proposed adaptive memetic algorithm has been studied on a real-time multi-robot path-planning problem. Experimental results obtained for both simulation and real frameworks indicate that the proposed algorithm based path-planning scheme outperforms real coded Genetic Algorithm, Particle Swarm Optimization and Differential Evolution, particularly its currently best version with respect to two standard metrics defined in the literature.

1 Introduction

Coined by Dawkins in 1976, the word "meme" refers to the basic unit of cultural transmission or imitation. Memetic Algorithms (MAs) are population-based meta-heuristic search algorithms that combine the composite benefits of natural and cultural evolution. MA captures the power of global search by its evolutionary component and local search by its cultural component.

The class of algorithms incorporating the adaptive selection of memes is referred to as Adaptive MA (AMA). Usually, the selection of the meme for an individual member of the population is done based on its ability to perform local improvement. The variant of AMA we would use in this paper is Roulette-Choice strategy based Hyperheuristic AMA [1]. In the Roulette-choice strategy, a meme M_e is selected with probability relative to the overall improvement. Given that g(.) is a choice function, then the probability of selection of M_e is $g(M_e)/\sum_{i=1}^{n} g(M_i)$ where n is the total number of memes considered.

The AMA to be proposed requires Differential Evolution (DE) for global search and a reinforcement learning algorithm for local refinement. The particular version of the DE (DE/current-to-best/1) used here has two parameters called scaling factors, which are adaptively selected from a meme pool. The scaling factors for all member of the population in a DE algorithm should not be equal for the best performance. A member with a good fitness should search in the local neighborhood, whereas a poor

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 248–258, 2012.

performing member should participate in the global search. A good member thus should have small scaling factors, while worse members should have relatively large scaling factors [2]. This is realized in the paper with the help of TDQL.

The TDQL works on the principle of reward and penalty. It employs a Q-table to store the reward/penalty given to an individual member of the population. Members are assigned suitable values of their scaling factors from a given meme pool before participation in the evolutionary process. After completion of the evolutionary process, members are rewarded based on their fitness, and the reward/penalty given to the member depending on the improvement/deterioration in fitness measures of the trial solution is stored in the Q-table. The process of evolution and Q-table updating thus synergistically helps each other, resulting in an overall improvement in the performance of the AMA.

The proposed AMA has successfully been employed in multi-robot path-planning. It refers to online trajectory planning from given starting positions to fixed goal positions for each robot without hitting teammates and obstacles [3], [4]. There exist two alternative approaches, centralized and distributed, to handle the problem. In a centralized approach, the next position of all the robots are determined from their current positions, by minimizing an objective function concerning total path of traversal by the robots, satisfying the necessary constraints on collision avoidance of individual robots with teammates and obstacles. In the distributed approach, the objective function with all the necessary constraints for the centralized problem is divided into n objective functions for n robots, where the i-th objective function refers to the distance objective and collision avoidance constraints for the i-th robot. The minimization problem in the centralized approach, which has a high order of computational complexity, now boils down to relatively simplified problem of minimization of n objective functions for n robots. In both the cases, the above problem is solved by iteratively identifying next positions of the robots until the goal position for all the robots are reached.

This paper considers an evolutionary path-planning with robots of definite size and circular cross-section and tested both in simulation and real environments, partitioned into square grids of equal size. The starting and the goal positions of each robot on the grid map are given, and the proposed AMA is used to locally plan the trajectory of motion of the robots with an aim to minimize the total path traversed by the robots without collision with obstacles. Performance metrics used in the existing literature [4] have been used here to compare the relative merits of the proposed AMA with respect to Genetic Algorithm (GA) based realization given in [3]. Experiments reveal that the proposed AMA based planner outperforms other realizations designed with Particle Swarm Optimization (PSO), DE/current-to-best/1 and SaDE.

The paper is divided into five sections. Section 2 provides an overview of the classical Q-learning. In section 3, we propose the AMA realized with DE and TDQL. Section 4 provides the formulation of the multi-robot motion planning problem and experiment with Khepera II mobile robots and computer simulation. Conclusions are given in section 5.

2 An Overview of the Classical Q-Learning

Let $S = \{s_1, s_2, ..., s_n\}$ be a set of states of an agent, $A = \{a_1, a_2, ..., a_n\}$ be a set of actions that the agent can select in each state $s_i \in S$, $r(s_i, a_j)$ and $Q(s_i, a_j)$ be the immediate and total reward that the agent acquires by execution of an action a_j at state s_i, $\delta(s_i, a_j)$ be the transition function that returns the next state s_k due to selection of action a_j at state s_i, i.e., $s_k = \delta(s_i, a_j)$, γ be the discounting factor used to penalize the future reward after a delay of k units by scaling it by a factor γ^k for positive integer k.

In Q-learning, the agent selects its next state from its current state by using a policy. The policy attempts to maximize the cumulative reward that the agent could attain in subsequent state-transitions from its next state. Let $\overset{*}{V}(s)$ be the total cumulative reward that the agent earns at state s, which is approximated as $\underset{a'}{Max}\, Q(s, a')$. Thus,

$$Q(s,a) = r(s,a) + \gamma \overset{*}{V} Q(\delta(s,a)) = r(s,a) + \gamma \underset{a'}{Max} Q(\delta(s,a), a') \tag{1}$$

In the classical Q-learning for deterministic state-transition, the algorithm begins with a randomly selected initial state. An action $a \in A$ is randomly selected, and the agent because of this action receives an immediate reward *r*, and moves to the new state using δ-transition rule, provided in a table. The Q-value of the previous state s due to selected action a is updated in a two-dimensional Q-table using (1). Now, the next state $s' = \delta(s,a)$ is considered as the initial state, and the steps of action selection, receiving immediate reward, transition to next state and Q-table updating are repeated forever.

Differential Q-learning is a modified version of Q learning. It has the ability to remember the effect of past Q value of a particular state-action pair while updating the corresponding Q value. The modified Q update equation is given by

$$Q(s,a) \leftarrow (1-\alpha) \times Q(s,a) + \alpha \times (r(s,a) + \gamma \underset{a'}{Max} Q(\delta(s,a), a')) \tag{2}$$

The learning rate α determines to the extent the newly acquired information will override the old information. A setting of $\alpha = 0$ makes the agent stop learning, while $\alpha = 1$ would make the agent consider only the most recent information. The discount factor γ determines the importance of future rewards. A factor of 0 will make the agent "opportunist" by only considering current rewards, while a factor approaching 1 will make it strive for a long-term high reward. If the discount factor is greater than or equal to 1, the Q values may diverge.

3 Proposed Adaptive Memetic Algorithm

The AMA to be proposed shortly includes a DE for global exploration and a TDQL for adaptive selection of memes. The process of adaptive selection of scaling factors F_1'

and F_2' as in (2) from the meme pool, followed by one step of DE and reward/penalty updating in the Q-table is continued until the condition for convergence of the AMA is satisfied. In this paper, F1' and F2' are set equal to F to save complexity to avoid maintenance of two Q-tables for each individual scaling factors.

The row indices of the Q-table represent states S_1, S_2, …, S_{NP} of the population obtained from the last iteration of the DE algorithm. A fitness function based rank evaluation of individual members is used to allocate the member to a specific state. Thus state S_k includes a member of rank k. The column indices of the Q-table correspond to uniformly quantized values of the scaling factors to be used in the evolutionary algorithm. Let the parameter under consideration be F with possible quantized values F_1, F_2, …, F_{10}. Then $Q(S_i, 10F_j)$ represents the total reward given to a member at state S_i for selecting $F=F_j$. The Roulette-Choice strategy is used to select a particular value of F from the meme pool $\{F_1, F_2, \ldots, F_{10}\}$ using the $Q(S_i, 10F_j)$, j=1, 2, …, 10 for the individual member located at state S_i. It must be noted that the factor 10 is used to get integer index of Q(.,.). Principles used in designing the AMA are introduced below.

1. Initialization: DE-TDQL starts with a population of NP D-dimensional parameter vectors within the prescribed minimum and maximum bounds: $\vec{X}_{\min} = \{x_{\min-1}, x_{\min-2}, \ldots, x_{\min-D}\}$ and $\vec{X}_{\max} = \{x_{\max-1}, x_{\max-2}, \ldots, x_{\max-D}\}$. Hence, we may initialize the j-th component of the i-th vector at generation G=0 as

$$x_{i,j}(G) = x_{j-\min} + rand_{i,j}(0,1) \times (x_{j-\max} - x_{j-\min}) \tag{3}$$

The entries for the Q-table are initialized as small values. If the maximum Q-value attainable is 100, then we initialize the Q-values of all cells in the Q-table as 1.

2. Adaptive Selection of Parameters of the DE: The probability of selection of $F=F_j$ from the meme pool $\{F_1, F_2, \ldots, F_{10}\}$ is given by

$$P(F_j) = Q(S_i, 10F_j) / \sum_{l=1}^{10} Q(S_i, 10F_l) \tag{4}$$

To maintain adaptation and learning in all Q's in each row, we select a particular F from the meme pool by a random selection. This random selection is realized by generating a random number r between (0, 1) and then we determine F_j, such that the cumulative probability of $F= F_1$ through F_{j-1} is less than a randomly generated number r, and the cumulative probability for $F= F_1$ through $F=F_j$ is greater than r. Symbolically, we need to hold:

$$\sum_{m=1}^{j-1} P(F=F_m) < r \le \sum_{m=1}^{j} P(F=F_m) \Rightarrow \frac{\sum_{m=1}^{j-1} Q(S_i, 10F_m)}{\sum_{l=1}^{10} Q(S_i, 10F_l)} < r \le \frac{\sum_{m=1}^{j} Q(S_i, 10F_m)}{\sum_{l=1}^{10} Q(S_i, 10F_l)} \tag{5}$$

4 Differential Evolution (DE/Current-to-Best/1)

4.1 Mutation

A donor vector $\vec{V}_i(G)$ corresponding to each population member or target vector $\vec{X}_i(G)$ is created by randomly selecting two other members $\vec{X}_{rand-1}(G)$ and $\vec{X}_{rand-2}(G)$ from the current population P_G, where

$$\vec{V}_i(G) = \vec{X}_i(G) + F_1'(\vec{X}_{best}(G) - \vec{X}_i(G)) + F_2'(\vec{X}_{rand-1}(G) - \vec{X}_{rrand-2}(G)) \quad (6)$$

and F'_1 and F'_2 are two scaling factors selected from the meme pool adaptively by step 2 before invoking the DE process in [0, 1].

4.2 Crossover

There are two types of crossover (recombination) schemes- binomial and exponential [2]. In the proposed realization we have used only binomial crossover. Here, a trial vector $\vec{U}_i(G)$ is generated for each pair of $\vec{V}_i(G)$ and $\vec{X}_i(G)$ by (5)

$$u_{i,j}(G) = \begin{cases} v_{i,j}(G) \; if \; rand_{ij} \le Cr \; or \; j = j_{rand} \; (j_{rand} \in [1, D]) \\ x_{i,j}(G) \; otherwise \end{cases} \quad (7)$$

where $rand_{i,j}(0,1) \in [0, 1]$ is a uniformly distributed random number lying in [0,1].

4.3 Selection

Here, for a given objective $f(\vec{x})$ to be minimized, we have

$$\begin{aligned} \vec{X}_i(G+1) &= \vec{U}_i(G) \quad if \; f(\vec{U}_i(G)) \le f(\vec{X}_i(G)) \\ &= \vec{X}_i(G) \quad if \; f(\vec{U}_i(G)) > f(\vec{X}_i(G)) \end{aligned} \quad (8)$$

5 Ranking of the Members and State Assignment

Let f_i be the fitness of the i-th member in the last iteration. A ranking policy is designed to compute normalized fitness $f_i / \sum_{j=1}^{NP} f_j, \forall i$, and then sort them in descending order. The r-th element of the sorted list has rank r, and this member is allocated to state S_r for all r=1 to NP.

6 Reward/Penalty Based Q-Table Updating

If the fitness of the member increases due to transition from S_i to S_k on selection of F_j, then $Q(S_i, F_j)$ will be updated following (9) with a positive reward function: reward(S_i, $10F_j$) = increase in fitness of the member,

$$Q(S_i,10F_j) = (1-\alpha)Q(S_i,10F_j) + \alpha(reward(S_i,10F_j) + \gamma \max_{F'} Q(S_k,10F')) \quad (9)$$

else $Q(S_i, 10F_j)$ will be evaluated by (9) with a negative reward= -K, of constant value, however, small.

7 Convergence

After each evolution, we repeat from step-2 until the termination condition is satisfied.

Pseudo Code of AMA

I. Set the generation number $t = 0$ and randomly initialize a population of NP individuals $P_t = \{\vec{X}_1(t), \vec{X}_2(t),..., \vec{X}_{NP}(t)\}$ with $\vec{X}_i(t) = \{x_{i,1}(t), x_{i,2}(t),..., x_{i,D}(t)\}$ for i=[1, NP] and each individual uniformly distributed in the range $[\vec{X}_{\min}, \vec{X}_{\max}]$ as given in (3) .Set α=0.25, γ=0.8. Evaluate $f(\vec{X}_i(t))$, for i= [1, NP]. Rank each vector according ascending order of cost function f(.). Let the ranked population be $R(0) = [r_1(0), r_2(0),..., r_{NP}(0)]$, where $r_i(0)$ denotes a target vector of rank i in t-th generation. Initialize $[Q(r_i(0), j)] = 1,\ j = [1,10]$ .for $r_i \in [1, NP]$ and $j \in [1,10]$ denotes the index of uniformly quantized scaling factor F, and t denotes t-th iteration.

II. **While** stopping criterion is not reached, **do**

Initialize $[reward(r_i(t), j)] = 0,\ r_i = [1, NP],\ j = [1,10]$.

For i=1 to NP **do**

II.a. **Roulette- Choice selection:** Randomly select a scaling factor F_{r_i} from 0.1 to 1.0 with an interval of 0.1 such that the probability of selection of a particular $F = F_j$ is $P(j) = Q(r_i(t), j) \Big/ \sum_{l=1}^{10} Q(r_i(t), l)$.

II.b. **Mutation:** Generate a donor vector $\vec{V}_i(t)$ corresponding to $\vec{X}_i(t)$ by (6).

II.c. **Crossover:** Generate trial vector $\vec{U}_i(t)$ for $\vec{X}_i(t)$ as in (7). Evaluate $f(\vec{U}_i(t))$.

II.d. **Selection:**

If $f(\vec{U}_i(t)) < f(\vec{X}_i(t))$ **Then do**

$reward(r_i(t), 10 \times F_{r_i}) = f(\vec{X}_i(t)) - f(\vec{U}_i(t)); \vec{X}_i(t+1) = \vec{U}_i(t);$

Evaluate $f(\vec{X}_i(t+1))$;

If $f(\vec{U}_i(t)) < f(\vec{X}_{best}(t))$ **Then** $\vec{X}_{best}(t) = \vec{U}_i(t)$; Evaluate $f(\vec{X}_{best}(t))$;
End If;

Else $reward(r_i(t), 10 \times F_{r_i}) = -K; \vec{X}_i(t+1) = \vec{X}_i(t);$

End If;
End For;

II.e. Determine the ranked population be $R(t+1) = [r_1(t+1), r_2(t+1), ..., r_{NP}(t+1)]$ in the (t+1)-th generation, where $r_i(t+1)$ denotes a target vector of rank i in (t+1)-th generation.

II.f. **Update Q-table:**
For i=1 to NP **do**
For F_j=0.1 to 1.0 **do**
If $reward(r_i(t), 10 \times F_j) \neq 0$ **Then**

$$Q(r_i(t), 10 \times F_j) = (1-\alpha) Q(r_i(t), 10 \times F_j) + \alpha [reward(r_i(t), 10 \times F_j) + \gamma \max_F Q(r_i(t+1), 10 \times F)];$$

End If;
End For;
End For;
Increase the counter value $t = t + 1$.
End While;

8 Realization of Multi-Robot Path-Planning Using DE-TDQL

A. Formulation

In the present context, we consider a 2-dimensional work-space, partitioned into equal sized square grids containing two or more mobile robot and obstacles with linear boundary. The grids are referred to by their distinct integer addresses. A potential robot path between a given starting and a goal position is constructed by joining two or more line segments. The line segments pass through a number of junctions, called intermediate nodes. The intermediate nodes are symbolized by their grid numbers. While planning a trajectory for a robot, other mobile robots are considered as moving obstacles. The path-planning problem for each robot is executed in steps until all robots reach their respective (predefined) goal positions.

Here, we represent a solution by a structure containing n fields, where the first (S) and the last fields (T) indicate the starting and the goal positions of the mobile robot. The second onwards successive (n-2) fields represent the intermediate nodes (Fig. 1 and 2).

We now propose an evaluation method to check the feasibility of a path. If all the line segments in a path are found to be free from intersection, the path length, defined by sum of the length of the line segments in the planned path, is assigned as its cost indicating the quality of the solution. Otherwise, the evaluation method assigns the cost by estimating the 'depths' of intersection of the constituent lines lying on the path with obstacles. A cost function [3] for a solution representative of a possible path for the i-th robot is given as

$$F_i = \sum_{j=1}^{N} \left(d_j + \beta_j C \right) \tag{10}$$

where N is the number of line segments in a path, d_j is the Euclidean distance of the two successive nodes forming the j-th line segment. The factor C is used to maintain uniformity in order of magnitude of the two summations. β_j is the coefficient denoting depth of collision, which is defined as

$$\beta_j = \begin{cases} 0 & if \;\; j-th \; line \;\; segment \;\; is \;\; feasible \\ \sum_{k=1}^{M} \alpha_k & if \;\; j-th \; line \;\; segment \;\; intersects \;\; with \;\; obstacles \end{cases} \tag{11}$$

Here, M is the number of obstacles the j-th line segment intersects. α_k is defined as the shortest moving distance for escaping the intersected obstacle [3]. The cost function F_i is to be minimized to determine the next position of robot i. The minimization of F_i is to be performed for all i in parallel. This has been taken care of by n DE-TDQLs each engaged to minimize one F_i for i =1 to n.

We now illustrate the measurement of α_k and β_j in Fig. 3. In Fig. 3(a), α_k is treated as the shortest distance to move the line out of the obstacle k. Fig. 3(b) elucidates a special example, where the line segment intersects two obstacles. It is evident that it is very difficult to determine the amount of shift of the line segment that will make it possible to move away from both obstacles. So the sum of α_1 and α_2 is used for calculation of β_j. For other complex configurations, the reader may consult the paper by Yang [3].

B. Experiments: The experiments were undertaken in two phases.

B.1. Experiments in Simulated Environment

Experiments were performed with n ($2 \leq n \leq 14$) similar soft-bots of circular cross section (radius=6 pixels) on a Pentium machine. For each robot the starting and the goal points are pre-defined prior to initiating the experiment. The experiments were performed with 2, 4, 6, 8 and 10 differently shaped obstacles. Extensive experiments were performed with 50 world maps of diverse configurations. We set no. of runs for each experiment, k =10.

B.2. Experiments in Real Environment on Khepera- II Platform

The experiment was undertaken with a world map of 8×6 grids of equal size and two Khepera-II mobile robots (diameter of 7 cm). Each robot is equipped with 8 infrared sensors, two motor driven side wheels and one caster wheel. The robots were used to sense obstacles around them in the world map and turn wheels by motor firing for controlled movement in prescribed directions. The robots were controlled by two Pentium-IV personal computers (PCs) through wired connections. A control program that determines the next position of a robot from its current position using DE-TDQL based optimization algorithm is run on the attached Pentium machine. The necessary commands for motor movements are transferred to the robots from their connected computers. Two sample runs of path-planning in the real environment is given in Fig. 4. It is observed that the robots follow the shortest paths avoiding collision with obstacles. The experiment was repeated 50 times on 10 different world maps of

different grid counts, each with five different obstacle-maps, and in all the 50 environments the robots could successfully trace the shortest paths. Results of the experiments performed are summarized in Table-1. Three performance metrics, namely 1) total number of steps taken to reach the goal, 2) ATPT, and 3) ATPD, defined in next sub-section, have been used here too to determine the relative merits of DE-TDQL over other algorithms. Table-1 confirms that DE-TDQL outperforms the remaining four algorithms with respect to all the three metrics.

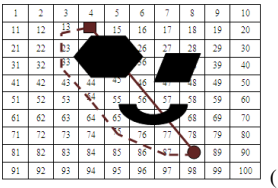


Fig. 1. The theoretical (solid line) and planned paths (dashed line) between given starting and goal positions

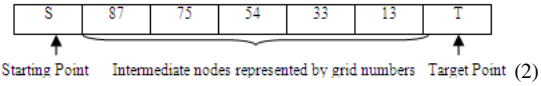


Fig. 2. An example of solution in the DE-TDQL-based multi-robot motion planning

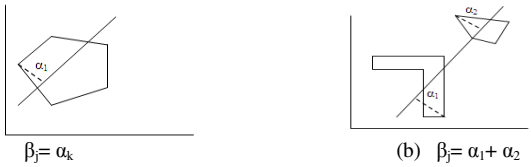


Fig. 3. Definition of the coefficient β_j

C. Performance Analysis

To determine a quantitative measure of the relative performance of different algorithms, we use two metrics suggested in [4].

Average Total Path Deviation (ATPD): Let P_{ik} be a path from the starting point S_i to the goal point G_i generated by the program for robot R_i in the k-th run. For n robots in the workspace the Average Total Path Deviation (ATPD) is $\sum_{i=1}^{n}(P_{i-ideal} - \sum_{j=1}^{k} P_{ij} / k)$ where $P_{i-ideal}$ is the theoretical shortest ideal path from given starting to goal position of i-th robot.

Average Uncovered Target Distance: Given a goal position G_i and the current position C_i of a robot on a 2-dimensional workspace uncovered target distance $UTD = \sum_{i=1}^{n} \|G_i - C_i\|$. Now, for k runs of the program, we evaluate the average of UTDs and call it the Average Uncovered Target Distance (AUTD).

In Fig. 6 we plot ATPT by varying no. of robots using 5 different algorithms, including GA, PSO, DE/current-to-best/1, SaDE and DE-TDQL. The second study on performance analysis was undertaken by plotting ATPD by generating paths by five

different evolutionary algorithms (as used in ATPT) with no. of robots as variable (Fig. 7). From these two figures, we observe that DE-TDQL outperforms the remaining four algorithms as both ATPT and ATPD remain the smallest for DE-TDQL irrespective to the no. of robots. The last analysis on performance was undertaken by comparing AUTD over the no. of planning steps (Fig. 8). It is apparent from Fig. 8 that DE-TDQL returns the smallest no. of steps required to reach the goal. In brief, the proposed DE-TDQL based path-planning outperforms all the four other algorithms with respect to all three metrics.

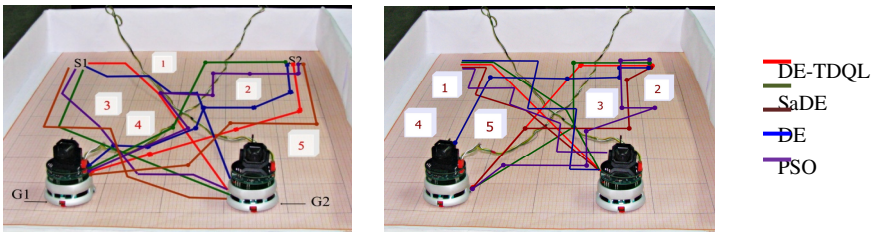


Fig. 4. Trajectories planned by execution of different algorithms in Khepera environment with five obstacles

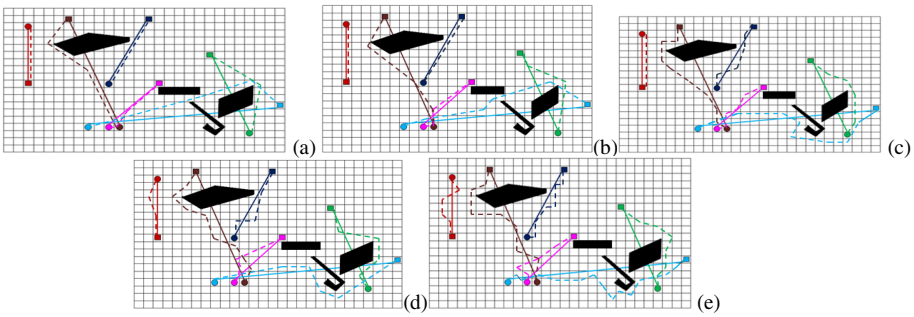
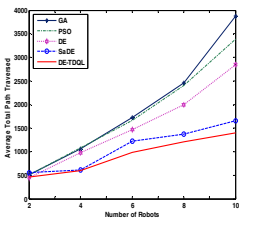
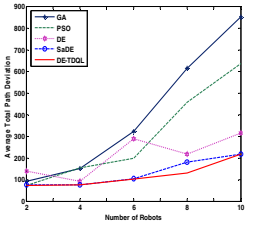


Fig. 5. Final configuration of the world map after execution of the (a) DE-TDQL (b) SaDE (c) DE (d) PSO and (e) GA based simulations with 6 robots and 2 obstacles requiring 23, 25, 29, 32 and 34 steps respectively



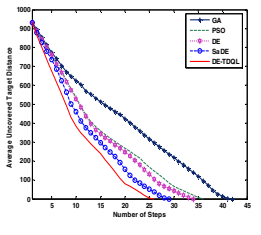
(6)

Fig. 6. Average total path traversed vs. number of robots with number of obstacles= 5



(7)

Fig. 7. Average total path deviation vs. number of robots with number of obstacles= 5



(8)

Fig. 8. Average uncovered target distance vs. number of steps with number of robots=5 and obstacles= 5

Table 1. Comparison of Number of Steps, ATPT and ATPD By the Robots

	World-map-I			World-map-II		
Algorithms	Number of Steps	ATPT (inch.)	ATPD (inch.)	Number of Steps	ATPT (inch.)	ATPD (inch.)
DE-TDQL	**10**	**42.2**	**7.2**	**12**	**43.2**	**8.2**
SaDE	12	44.9	9.9	15	46.1	11.1
DE	17	46.4	11.4	18	46.8	11.8
PSO	19	47.1	12.1	21	48.6	13.6
GA	23	50.0	15.0	23	50.3	15.3

9 Conclusion

The paper introduced a new technique for efficiently employing DE and Q-learning together to develop an adaptive memetic algorithm. A case study on multi-robot path-planning problem has been undertaken to demonstrate the importance of the proposed DE-TDQL algorithm. A formulation of the objective function for the problem has been given following [3], and the DE-TDQL algorithm is employed to minimize the objective function in order to determine the next position of all the robots from their current positions in the given world map. The experiments undertaken reveal that the DE-TDQL based AMA here too outperforms classical DE and PSO, real coded GA and SaDE algorithms with respect to two parameters AUTD and ATPD. The experiments performed in real environment with Khepera-II mobile robots give the same results as in simulated environment, thereby justifying the efficacy of the algorithm free from system-level implementation.

References

1. Cowling, P., Kendall, G., Soubeiga, E.: A Hyperheuristic Approach to Scheduling a Sales Summit. In: Burke, E., Erben, W. (eds.) PATAT 2000. LNCS, vol. 2079, pp. 176–190. Springer, Heidelberg (2001)
2. Storn, R., Price, K.V.: Differential Evolution–a simple and efficient heuristic for global optimization over continuous spaces. J. Global Optimization 11(4), 341–359 (1997)
3. Yang, S.X., Hu, Y., Meng, M.Q.H.: A Knowledge Based GA for Path Planning of Multiple Mobile Robots in Dynamic Environments. In: IEEE Conference on Robotics, Automation and Mechatronics, pp. 1–6 (June 2006)
4. Chakraborty, J., Saha, S.: Co-operative Multi-robot Path Planning Using Particle Swarm Optimization. In: Proc. of IEEE WIE National Symposium on Emerging Technologies (2007)
5. Lin, C.C., Chen, K.C., Chuang, W.J.: Motion Planning using a Memetic Evolution Algorithm for Swarm Robots. International Journal of Advanced Robotic Systems 9, 1–9 (2012)
6. Gayle, R., Moss, W., Lin, M.C., Manocha, D.: Multi-Robot Coordination using Generalized Social Potential Fields. In: IEEE International Conference on Robotics and Automation (2009)
7. Bhattacharjee, P., Rakshit, P., Goswami, I., Konar, A., Nagar, A.K.: Multi-Robot Path-Planning Using Artificial Bee Colony Optimization Algorithm. In: NaBIC 2011, pp. 219–224 (2011)

Power Loss Minimization by the Placement of DG in Distribution System Using GA

Dasarathan Sattianadan[1], M. Sudhakaran[2],
Subhransu Sekhar Dash[1], and K. Vijayakumar[1]

[1] SRM University, Chennai, India
[2] Pondicherry Engg. College, Pondicherry, India
sattia.nadan@gmail.com,
{karan_mahalingam,munu_dash_2k}@yahoo.com,
kvijay_srm@rediffmail.com

Abstract. Power loss minimization is an important aspect of distribution System where the load variation is more compared to other systems. There are different methods to minimize the power loss like DG placement, capacitor placement, load balancing etc. Among those methods DG placement was much more beneficial because it is directly related to real power loss. This paper is based on power loss minimization by the placement of distributed generators (DG) in the distribution system. The location of DG is found with the help of voltage stability index (VSI) and DG size is varied in small steps and corresponding power loss is calculated by running the power flow and the result obtained is verified by Genetic Algorithm. The simulation study is carried out on a 33 bus Distribution System by considering different load models.

Keywords: Distributed generation, Voltage Stability Index, Genetic Algorithm.

1 Introduction

In the modern world, day by day the load demand increases rapidly due to Industrial and Domestic needs. On the other hand the conventional energy sources are decreasing rapidly. In this case we need an alternative method to meet the load demand, distributed generation is meant for that. It has huge potential benefits about which this paper is concerned.

The distributed generation has been defined by many researchers [1,2], but in general distributed generation is nothing but a small generator which is connected at the consumer terminal. Placement of DG is an important factor because the improper location may leads to voltage instability. The Newton Rapson load flow method used in [3]. This method reduces the power loss and the cost factor very effectively, but the conventional method of load flow analysis was not applicable for distribution system because of its high R/X ratio, a large value of resistance and reactance of the line and radial structure of the distribution system.

Tuba Gozel used loss sensitivity factor for the determination of the optimal size and location of DG to minimize total power loss [4]. Andrew used the Linear

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 259–266, 2012.

Programming Technique for placement of DG with multiple constraints [5]. Mallikarjuna used Simulated Annealing for determining the optimal location and size of DG units in a microgrid, given the network configuration and heat and power requirements of various load points [6]. Krueasuk used PSO to find optimal location and size of DG [7]. Lalitha used fuzzy approach to find optimal DG localization [8].

Hughifam used multi-objective function to minimize cost of energy losses, Investment cost of DG and Operation and maintenance cost [9]. Ochoa minimized real power loss and simple phase short circuit level [10]. Celli used multi objective approach, based on the non-dominated sorting Genetic Algorithm has been adopted to solve the optimal placement of different types of generation simultaneously. He saved the energy in the form of greenhouse gas emission reduction[11]. Vinoth Kumar addressed minimizing the multi objective index using genetic algorithm for the optimal Placement of DG[12].

This paper minimizes the Power loss by the Placement of the optimal size of DG and is organized as follows: Section-2: Defines the Objective Function and the load flow analysis of the distribution system, Power flow problem is a crucial part of power system design procedures, and it is categorized into the transmission power flow and the distribution power flow. The distribution networks commonly have some special features such as: Unbalanced loads and unbalanced operation; being radial with sometimes weakly-meshed topology; and high resistance to reactance R/X ratios. Due to these features the conventional load flow like Gauss Sedial and Newton Rapson fail to solve. Hence we need a special method to solve the load flow on Distribution Systems, here network Topological based load flow has been considered for load flow Analysis [13,14]. Section-3: Candidate Bus Selection by using VSI and Load Modelling is discussed. Section-4: Explains Genetic Algorithm Technic. Section-5: Test Results and Discussion Section 6: The paper is concluded..

2 Objective Function

The objective of the present optimization problem is to minimize the Distribution network power loss

$$\text{Min.} f_1 = \sum_{b=1}^{N_b} (I_b)^2 \cdot R_b \tag{1}$$

Where,

N_b- Total number of branches in the given radial distribution system

b - Branch number

I_b - Branch current in branch b

R_b- Resistance of branch b

2.1 Load Flow Analysis for Radial Distribution System

The simple distribution system shown in Fig.1 will be used as an example. The power injections can be converted into the equivalent current injections using Eq. (2)

$$I_i = (P_i + Q_i / V_i)^* \tag{2}$$

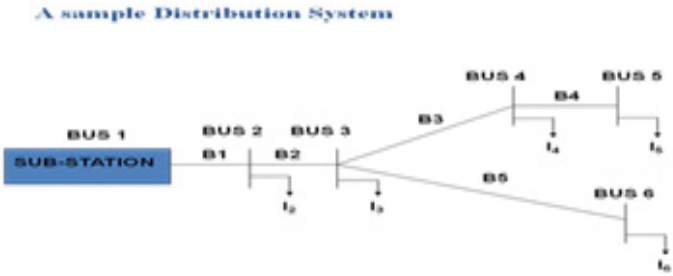


Fig. 1. A sample Distribution System

And a set of equations can be written by applying Kirchhoff's Current Law (KCL) to the distribution network. Then, the branch currents can be formulated as a function of the equivalent current injections. For example, the branch currents *B*5, *B*3 and *B*1 can be expressed as,

$$\begin{aligned} B_5 &= I_6 \\ B_3 &= I_4 + I_5 \\ B_1 &= I_2 + I_3 + I_4 + I_5 + I_6 \end{aligned} \tag{3}$$

The Bus-Injection to Branch-Current (***BIBC***) can be obtained by using the above equations. The Branch-Current to Bus Voltage (BCBV) matrix is responsible for the relations between the branch currents and bus voltages. The corresponding variation of the bus voltages, which is generated by the variation of the branch currents, can be found directly by using the BCBV

$$V_2 = V_1 - B_1 Z_{12} \tag{4}$$

$$V_3 = V_2 - B_2 Z_{23} \tag{5}$$

$$V_4 = V_3 - B_3 Z_{34} \tag{6}$$

By using equ (4) and equ (5), The voltage of Bus 4 can be rewritten as,

$$V_4 = V_1 - B_1 Z_{12} - B_2 Z_{23} - B_3 Z_{34} \tag{7}$$

2.2 Algorithm for Distribution System Load Flow:

A brief idea of how bus voltages can be obtained for a radial system is given below.

1. Input data.
2. Form the *BIBC* matrix.
3. Form the *BCBV* matrix.
4. Form the *DLF* matrix.
5. Iteration $k = 0$.

6. Iteration $k = k + 1$.
7. Solve the equations iteratively and update voltages $I_i^k = (P_i+Q_i /V_i)$
   $[\Delta V^{k+1}] = [DLF] [I^k]$

If $I_i^{k+1} - I_i^k$ >tolerance, go to step(6)
Else print result.

3 Candidate Bus Selection Using Voltage Stability Index

A system experiences a state of voltage instability when there is a progressive or an uncontrollable drop in voltage magnitude following a disturbance, increase in load demand or change in operating condition. It is usually identified by an index called voltage stability index of all the nodes in a radial distribution system [4].

$$VSI\ (n_2) = |V_1^4| - 4[P_2R_1 + Q_2X_1]\,|V_1^4| - 4[P_2X_1 - Q_2R_1]^2 \quad (8)$$

Nodes with minimum voltage instability, in different laterals of the distributed system are chosen as the candidate location for placement of distributed generators. Following steps are involved in the optimal siting of the distributed generator

a) Perform load flow to calculate the bus voltage magnitudes and total network power loss in the RDS.
b) Compute the Voltage Stability Index (VSI) using equ.(8)
c) Select the buses with the highest priority i.e. the lowest value of VSI (First,Second) and place DG.
d) Run the power flow program again and find losses of the power system.
e) Check the voltage profile limitation at each bus of the system.
f) Change the size of DG to small step and calculate loss by running load flow.
g) Find the bus which leads to the lowest power system losses.

3.1 Load Modeling

A balanced load that can be represented either as constant power, constant current or constant impedance load has been considered here. The general expression of load is given below

$$P(m) = P_n [a_1 + a_2V(m) + a_3V(m)^2] \quad (9)$$

$$Q(m) = Q_n [b_1 + b_2V(m) + b_3V(m)^2] \quad (10)$$

Where,

P_n, Q_n - Nominal real and reactive power respectively
V(m) – Voltage at node m

For all the loads, equation (9) and equation (10) are modeled as

$$a1+a2+a3 = 1.0 \quad (11)$$

$$b1+b2+b3 = 1.0 \quad (12)$$

For Constant Power (CP) load $a_1= b_1= 1$ and $a_i = b_i = 0$ for i=2, 3. For Constant Current (CI) load $a_2= b_2= 1$ and $a_i = b_i = 0$ for i= 1, 3. For Constant Impedance (CZ) load $a_3= b_3= 1$ and $a_i = b_i = 0$ for i=1, 2

4 Genetic Algorithm

(i) Fitness Function

Genetic Algorithm work with population of individuals. Each individual stands for solution. The quality of solution is determined by fitness function. The fitness function is used to select individuals from current generation to advance in to next generation. This process is continued until there is no chance for best individuals in the population

(ii) Selection

Some of individual selected based on fitness function. These individual go mate and pass the genetic code to the next generation.

(iii) Cross over

Selected individuals can be subjected to cross over with a probability P = 0.6-0.9 defined by the user. The sons have the properties and chacteristics of both their parents.

(iv) Mutation

The Cross over operator cannot explore the whole search space because there is no new information introduced. Mutation introduced the new element by changing 0 to 1(or) 1 to 0 which increases the search space.

In this paper the following GA parameters have been considered,

Population Size : 20

Scaling Function : RANK

Selection Function : Stochastic Uniform

Mutation Function : Constraint Dependent

Crossover Function: Scattered

Crossover Fraction: 0.8

Generations : 100

Initial Penalty : 10

Penalty Factor : 100

5 Test Result and Discussion

To analysis the effect of DG on the Distribution system, the 33Bus system has been considered here. The effect of DG with different load models has been shown in fig.2. From the fig it was clear that the optimal size of DG is 0.14MW for reduced loss and the optimal location has been selected by using VSI, Which is at location 18.

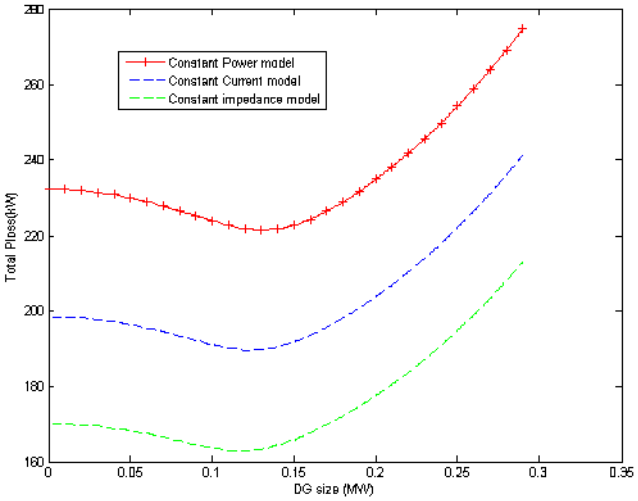


Fig. 2. Comparative Analysis of Load models using VSI- 33Bus System

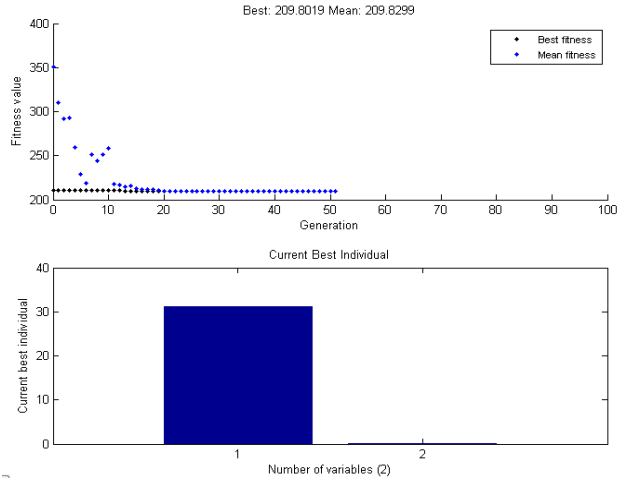


Fig. 3. Output of Genetic Algorithm- 33Bus System

The same test system with the Constant Power model has been tested with Matlab GA tool box and test result shown in fig.3. From the fig.3 it was found that the best location is 32 for minimizing power loss at 209.8kW the corresponding DG size is 0.2MW. Table.1 gives the information about the comparative analysis of power loss reduction with the consideration of different load models.

The effect on Bus Voltages for Constant Power model by the Placement of DG for the test system has been shown in fig.4. From the fig.4.it has been found that GA gives the better voltage profile improvement compare to VSI and Base case.

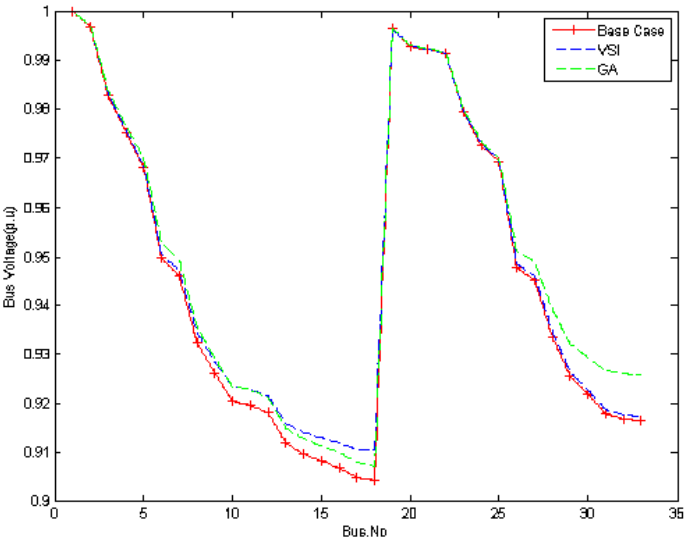


Fig. 4. Comparative Analysis of Bus Voltages for Constant Power model- 33Bus System

Table 1. Comparative Analysis of Power Loss Reduction

Test case	33 bus system					
Load Model	Constant Power model Base Case: Ploss = 232.28kW Qloss = 157.18kvar Min.Bus Voltage = 0.9043p.u		Constant Impedance model Base Case: Ploss =170.2kW Qloss = 114.5kvar Min.Bus Voltage = 0.9196p.u		Constant Current model Base Case: Ploss =198.54kW Qloss =133.95kvar Min.Bus Voltage = 0.9123p.u	
Approach	VSI	GA	VSI	GA	VSI	GA
Optimal location	18	32	18	32	18	32
Optimal Size of DG in MW	0.12	0.2	0.12	0.2	0.13	0.2
Ploss in kW	221.28	209.82	163.08	154.69	189.6	179.94
Qloss in kvar	149.14	141.80	109.26	103.91	127.44	121.16
Min.Bus Voltage(p.u)	0.9105	0.9077	0.9248	0.9222	0.9175	0.9153

6 Conclusion

In this paper the optimal location of DG is obtained using VSI and the size of the DG is obtained by trial and error method to minimize the real losses. GA. is used simultaneously to find the location and size both to minimize the losses. Even though the size of DG obtained by GA is high, for this size the loss obtained by VSI method is greater than the loss obtained by GA, so GA gives better performance when compared to other methods.

References

1. Ackermann, Anderson, G., Sooder, L.S.: Distributed generation: A generation. Electrical Power Systems Research 57 (2001)
2. El-Khattam, W., Salama, M.M.A.: Distributed generations technologies, definitions and benefits. Electrical Power Systems Research, 119–128 (2004)
3. Ghosh, S.: Optimal sizing and placement of distributed generation in a network system. Electrical Power and Energy Systems, 849–856 (2010)
4. Gozel, T., Hakan Hocaoglu, M.: An analytical method for the sizing and siting of distributed generators in radial systems. Electrical Power Systems Research, 912–918 (2009)
5. Keane, A., O'Malley, M.: Optimal Allocation of Embedded Generation on Distribution Networks. IEEE Transactions on Power Systems 20(3), 1640–1646 (2005)
6. Vallem, M.R., Mitra, J.: Siting and Sizing of Distributed Generation for Optimal Microgrid Architecture. In: IEEE Conference, pp. 611–616 (2005)
7. Wichit, K., Weerakorn, O.: Optimal Placement of Distributed Generation Using Particle Swarm Optimization. M. Tech Thesis. AIT, Thailand (2005)
8. Lalitha, M.P., Reddy, V.C., Usha, V., Reddy, N.S.: Application of fuzzy and PSO for DG placement for minimum loss in radial distribution system. ARPN Journal of Engineering and Applied Sciences 5(4), 32–37 (2010)
9. Haghifam, M.-R., Falaghi, H., Malik, O.P.: Risk-based distributed generation placement. IET Gener. Transm. Distrib. 2(2), 252–260 (2008)
10. Ochoa, L.F., Padilha, F.A., Harrison, G.P.: Evaluating distributed time-varying generation through a multiobjective index. IEEE Trans. Power Delivery 23(2), 1132–1138 (2008)
11. Celli, G., Ghiani, E., Mocci, S., Pilo, F.: A multiobjective evolutionary algorithm for the sizing and sitting of distributed generation. IEEE Trans. 20(2), 750–757 (2005)
12. Vinothkumar, K., Selvan, M.P.: Impact of DG Model and Load Model on Placement of Multiple DGs in Distribution System. In: IEEE Conference, pp. 508–513 (2010)
13. Teng, J.H.: A Network-Topology Based Three Phase Load Flow for Distribution Systems. Proceedings of National Science Council ROC (A) 24(4), 259–264 (2000)
14. Teng, J.H.: A Direct Approach for Distribution System Load Flow Solutions. IEEE Transaction on Power Delivery 18(3), 882–887 (2003)

Multipopulation Based Differential Evolution with Self Exploitation Strategy

Rupam Kundu, Rohan Mukherjee, and Shantanab Debchoudhury

Department of Electronics and Telecommunication Engineering,
Jadavpur University
{rupam2422,rohan.mukherjii,sdch10}@gmail.com

Abstract. In this article a multi-population based DE-variant has been proposed to tackle DOPs. The algorithm, denoted as MPBDE-SES uses a self exploitative scheme along with classical DE. Moreover it also uses Brownian and Quantum individuals. An aging mechanism has been incorporated to get rid of stagnation. Apart from this exclusion principle, repulsion scheme and a recombination based mutation strategy causes uniform distribution of the subpopulation over the entire search space which enhances the tracking ability of the algorithm. Performance of MPBDE-SES has been tested over the suite of benchmark problems used in Competition on Evolutionary Computation in Dynamic and Uncertain Environments, held under the 2009 IEEE Congress on Evolutionary Computation (CEC) and compared with six state-of-the-art EAs. The results obtained clearly and statistically outperform the other algorithms.

1 Introduction

Several problems that we face in the real world are mostly dynamic in nature. In Dynamic Optimization Problems (DOP's) the functional landscapes changes with time, and so the optimizer has to track the global optima adapting to the change in environment. For solving DOP's early convergence imposes limitation on the conventional EA's, rather a proper balance of exploration and exploitation can cause the conventional EA's respond effectively to dynamically changing environment and continuously track the moving optima. Differential Evolution has emerged one of the best real parameter optimizers in the field of Dynamic Optimization.

In this article we have presented a multi-population based DE-variant with Brownian and Quantum Individuals addressed as MPBDE-SES. The MPBDE-SES algorithm is a combination of a number of strategies for handling the dynamic nature of the functional landscapes considered, among which exploration has been its key feature besides exploitation to ensure faster convergence avoiding local optima. Different techniques have been adopted while switching between the changes in environment to recover from a already converged population for further explorations.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 267–275, 2012.

2 MPBDE-SES

In this section we discuss in details the various elements of the proposed algorithm. The strategies adopted are as follows- Classical DE, self exploiting interaction, exclusion rule, aging mechanism, Brownian and Quantum individual, a recombination based mutation strategy and a repulsion scheme.

2.1 Classical DE

Classical DE consists of a number of steps as:

a. **Mutation with Difference Vectors:** DE creates a *donor* vector corresponding to each population member or *target* vector in the current generation through mutation.
b. **Crossover:** Through crossover, the donor vector mixes its components with the target vector under this operation to form the trial vector.
c. **Selection:** Selection determines whether the target or the trial vector survives to the next generation.

Detailed analysis of DE algorithm can be found in [1, 2].

2.2 Self Exploitation Strategy

Considering two ethnic gene pools X and Y, suppose in gene pool X gene A is predominant while gene B is almost absent. On the other hand for gene pool Y gene B is predominant while gene A is almost absent. Now consider a situation when the two gene pools get intermixed the result is production of a new variety of gene pool with comparatively higher percentage of gene B than in X. At the same time there is enough probability that the high percentage of gene A is retained along with a better percentage of gene B. What we now have is an enriched version of gene pool with higher percentage of both A and B. The reverse comparison is true if we consider gene pool Y. Now if we look at such subsequent steps of intermixing of varied gene pools produced the result is a dynamic optimization-like scenario where interaction leads to an optimized result. The algorithm for the Interaction Scheme is given below:

```
For i=1:NP
 For  j=1:NP
    Calculate the better particle among the i-th and
    j-th particle
    Compute the vector from the worse particle towards
    the better particle
    Add a scaled version of it to the current position
    of the better particle.
    Accept it if it is better than the previous
    position.
  End for
 End for
```

2.3 Exclusion Principle

The exclusion principle ensures the distribution of populations around different basins of attraction. Suppose if two populations are initialized very close to each other, or if they are moving toward the same basin of attraction, it is unnecessary to allow both of them to follow the same path. The $\vec{l}_{best}$ of two subpopulations are noted and the Euclidean distance between them is calculated. If this distance between these $\vec{l}_{best}$ of any two subpopulations is less than a limiting value then the subpopulation whose best individual has a lower objective function value between the two is considered for re-initialization. For a search range R, a D-dimensional search space, NP number of populations, the limiting value for the distance is given by $exclusion\_radius <= R/(NP * D)$.

2.4 Aging Mechanism

While the searching procedure is going on there can be a possibility for the subpopulations to get trapped in some local optima, which causes a hindrance to the searching process. To mitigate this problem, we employ the aging mechanism, taking into account the consistent poor performance of that particular subpopulation that has gone trapped.

Pseudo Code for Aging Mechanism:

```
For i=1:NP
If (change in fitness of best element<1)
Age(i)=age(i)+1;
Else Age(i)=0;
End if
If age(i)> λ
Reinitialise that subpopulation
End if
End for
```

The aging parameter λ="10" is chosen so because if the value of λ is very small then we are eliminating the possibility for the subpopulation to recover from being trapped, and if it is very large unnecessary function evaluations are wasted for a subpopulation trapped in a local optima.

2.5 a Quantum Individual Generation Scheme

The steps for stochastically generating an individual, positioned inside a sphere of radius R and centered on the local best position $\vec{l}_{best,G}$ are as follows:

i) Generate a radius that depends on the present functional value $f(\vec{X}(t))$, given as

$$radius(t) = \min[1, \max\{0.6, |\log_{10}(f(\vec{X}(t)))|\}].$$

ii) Generate a vector taking each component randomly from a standard normal distribution with mean= 0 and variance=1. $\vec{R}_{i,G} = \{r_1, r_2, r_3, ..., r_D\}; r_i \in (0,1)$ and

$1 \le i \le D$ where $\mathcal{N}(\mu,\sigma)$ denotes a normal distribution with mean μ and standard deviation σ

iii) Compute the distance of this vector from the origin. The distance is given by Dist such that Dist= $\sqrt{\sum_{i=1}^{D} r^2}$

iv) New quantum individual's position will be given by

$$\vec{l}_{best,G} + radius(t) * (\vec{R} / Dist) * \vec{X}_{i,G}$$

2.5 b Brownian Individual Generation Scheme

New individuals are generated in a Gaussian hyper parallelepiped centered at the position $\vec{l}_{best,G}$.Brownian individual's position will be: $\vec{l}_{best,G}+\mathrm{N}(0,\sigma)$ .where σ is the standard deviation of the normal distribution from which the perturbation is randomly sampled.

2.6 Repulsion Scheme

After the end of one generation if the same sub population is employed to track the global optima in the next generation the searching ability of the sub population decreases due to the lack of diversity and uniform distribution of the individual members of the sub populations in the entire search space. So we introduce two schemes to get rid of the above problem namely,

i. Perform crossover operation between the current population and the initial population stored at the beginning in a separate archive with probability of selection CR as mentioned in 2.1b.
ii. **Intra-repulsion scheme.**

Intra-repulsion Scheme: In this scheme we evaluated Euclidean distance between every two particles of each subpopulation. If they are located within the repulsion radius, the particle with lower fitness value will be repulsed from the other. Change is accepted only if better result with respect to its previous position is encountered.
In MPBDE-SES we have taken the value of radius of repulsion as 1. The value chosen is a significantly moderate one so that the individuals are as far dispersed as possible so that proper mutation enables them to create a varied range of diversity that ensures in the changed environment a property that exploits and explores as well. The obtained vectors are restricted within bounds as mentioned in the problem.

Differential Evolution is unique in the sense that its functional value is monotonic increasing or monotonic decreasing in accordance to the optimization problem. At the end of each generation the functional value of best particle is re-evaluated. If a sharp deterioration is detected rather than an improvement, dynamic change is inferred.

3 Program Code

1. Initialise the population(subpopulation size[r]]*no. of subpopulation[*NP*])
2. Consider initial age of all the subpopulation=0
3. Compute exclusion radius =(maximum-minimum)/*NP*D*;
4. While(FES<no of _changes**d***10,000)
5. Calculate fitness of every particle of subpopulation.
6. While dynamic change not detected
   a) Evaluate best particle with respect to fitness for each subpopulation.
   b) Compute the distance between best particles of all subpopulations.
   c) If distance between any two subpopulations < rexcl
   d) Reinitialise the worse subpopulation among the two.
   e) Send the other subpopulation for **Self exploitation Strategy.**
   f) Update fitness value for all particles in the population and re-evaluate the best elements.
   g) Include diversity in every subpopulation according to **Diversity scheme**(Brownian and Quantum).
   h) **Aging mechanism** (2.4).
   i) Find out the globally best population.
   j) Re-evaluate fitness of best particle and compare to detect whether dynamic change has occurred or not as mentioned in (2.6).
7. End while
8. **Repulsion Scheme** (2.6).
9. End while

4 Experimental Setup and Results

All the experiments have been performed on machines having the following specifications: CPU: 2.26 GHz Intel Core i3 RAM: 4 GB and MATLAB 2009b edition. We have presented the performance of our algorithm in terms of mean and standard deviation (STD) of error values obtained over the suite of benchmark problems used in Competition on Evolutionary Computation in Dynamic and Uncertain Environments, held under the 2009 IEEE Congress on Evolutionary Computation (CEC)[2]. We have compared these results with five significant evolutionary DOP solvers DASA[3], jDE[4], dopt-aiNET[5], CPSO[6] and CESO[7]. In all the cases the $no\_of\_subpopulation = 10$, $subpopulation\_size = 10$.

The results obtained in all the test cases mentioned above by MPBDE-SES and 6 other significant evolutionary dynamic optimizers are tabulated below in Table 1.

Table 1. Error values achieved by MPBDE-SES and other algorithms on the benchmark problems generated by GDBG for test functions F1-F7

Test Functions	Algorithm	Error	T1	T2	T3	T4	T5	T6	T7
F1 (Number of peaks=10)	MPBDE-SES	Average	**2.70e-05**	**8.60e-03**	**1.29e-06**	**5.50e-03**	**9.30e-03**	**1.04e-01**	**1.32e-06**
		STD	**5.82e-05**	**2.12e-02**	**1.12e-06**	**7.00e-08**	**2.27e-02**	**2.54e-01**	**2.26e-05**
	DASA	Average	1.80 e–01	4.18	6.37	4.82 e–01	2.54	2.34	4.84
		STD	1.25	9.07	10.7	1.95	4.80	8.66	8.96
	jDE	Average	0.028813	3.5874	2.99962	0.015333	2.17757	1.1457	3.5017
		STD	0.442537	7.83849	7.12954	0.288388	4.38812	5.72962	7.89858
	DynDE	Average	0.07324	2.5567	5.4245	0.1263	1.5651	1.3115	4.1137
		STD	2.9567	8.4313	9.2485	0.9426	4.6461	6.2511	8.5249
	dopt-aiNET	Average	0.1353	5.8667	4.2545	5.3563	4.4356	9.9407	4.2110
		STD	1.0061	10.2772	8.1828	8.9414	5.5545	15.8214	8.6873
	CPSO	Average	0.03514	2.718	4.131	0.09444	1.869	1.056	4.54
		STD	0.4262	6.523	8.994	0.7855	4.491	4.805	9.119
	CESO	Average	0.072645	2.36549	5.17873	0.092156	1.45614	0.92819	3.93569
		STD	2.87253	7.93215	8.97652	0.815189	4.45321	5.92797	8.4234
F1 (Number of peaks=50)	MPBDE-SES	Average	**7.89e-06**	**2.16e-05**	**0.4276**	5.127	**9.80e-03**	**0.1429**	**0.1429**
		STD	**1.04e-05**	**4.63e-05**	**1.0474**	9.05	**0.0221**	**0.3485**	**2.7061**
	DASA	Average	4.42 e–01	4.86	8.42	5.09 e–01	1.18	2.07	7.84
		STD	1.39	7.0	9.56	1.09	2.18	5.97	9.05
	jDE	Average	0.172355	4.08618	4.29209	0.0877388	0.948359	1.76542	4.36913
		STD	0.763932	6.4546	6.74538	0.24613	1.76552	5.82652	6.9321
	DynDE	Average	0.3286	4.6547	6.4641	0.1412	1.01621	0.9859	6.2513
		STD	1.5224	6.3453	9.3523	0.5914	2.6489	4.8631	9.06517
	dopt-aiNET	Average	0.3644	4.7485	5.2531	2.6565	2.8641	6.8330	4.4172
		STD	0.9275	6.7580	6.6830	5.9773	4.1579	11.8790	6.4528
	CPSO	Average	0.2624	3.279	6.319	0.125	0.8481	1.482	6.646
		STD	0.9362	5.303	7.442	0.3859	1.779	4.393	7.94
	CESO	Average	0.57932	4.35869	6.26213	0.136004	0.981202	0.949049	6.14005
		STD	1.425123	6.21243	9.16489	0.525679	2.596313	4.78245	8.48542
F2	MPBDE-SES	Average	**0.4918**	**3.23e-04**	**9.2866**	6.1032	**1.6791**	**1.98**	**1.1129**
		STD	**1.2044**	**6.14e-04**	**12.902**	2.66379	**4.113**	**4.887**	**4.5886**
	DASA	Average	3.30	25.6	18.9	1.45	4.96	2.11	3.87
		STD	8.78	83.2	67.8	3.83	1.12	5.29	8.12
	jDE	Average	0.963039	43.0004	50.1906	0.793141	67.0523	3.36653	13.2524
		STD	3.08329	114.944	124.015	2.53425	130.146	12.9738	45.7797
	DynDE	Average	1.3627	13.0179	11.9214	0.7842	20.7842	2.1845	2.4235
		STD	5.03159	48.2532	45.7054	2.2248	64.5341	3.9643	7.1031
	dopt-aiNET	Average	0.984	8.1209	17.9979	1.0652	101.3840	6.5192	3.7385
		STD	0.0291	14.3832	62.2259	2.8269	134.5180	13.8172	7.9542
	CPSO	Average	1.247	10.1	10.27	0.5664	25.14	1.987	3.651
		STD	4.178	35.06	33.45	2.137	64.25	5.217	6.927
	CESO	Average	1.34512	12.1431	11.5219	0.75431	20.5651	2.06489	2.32687
		STD	4.66213	47.2519	43.6323	2.16255	62.7372	3.88546	7.00123
F3	MPBDE-SES	Average	640.041	676.594	630.7124	693.1722	619.673	709.9972	639.54
		STD	37.6728	52.6616	22.3028	52.6616	22.9609	74.3324	4.4161
	DASA	Average	1.57 e+01	824	688	4.35 e+02	6.97	6.26 e+02	4.33 e+02
		STD	6.71 e+01	204	298	4.41 e+02	3.15	4.60 e+02	3.80 e+02
	jDE	Average	11.3927	558.497	572.105	65.7409	475.768	243.27	153.673
		STD	58.1106	384.621	386.09	208.925	379.89	384.98	286.379
	DynDE	Average	21.2512	792.4579	635.6154	341.7015	749.2651	519.5245	415.3212
		STD	73.6549	255.6163	342.7753	419.8116	280.9181	438.2467	390.3450
	dopt-aiNET	Average	810.8300	1078.750	1073.4300	1031.5300	1023.90	1186.9000	1061.8300
		STD	66.1085	64.1245	64.9950	274.7490	57.8713	292.2960	110.0980
	CPSO	Average	137.5	855.1	765.9	430.6	859.7	753	653.7
		STD	221.6	161	235.8	432.2	121.5	361.7	334
	CESO	Average	19.7312	791.1642	634.5213	339.5279	747.5267	516.1254	414.9842
		STD	73.6549	254.1579	341.2314	416.6123	279.3412	436.6415	388.3141

Table 1. (*continued*)

F4	MPBDE-SES	Average	**0.485**	**13.96**	**5.790**	1.882	**29.53**	**0.036**	**2.446**
		STD	**1.188**	**20.24**	**9.577**	3.600	**65.15**	**0.061**	**7.696**
	DASA	Average	5.60	65.6	53.6	1.85	1.08e+02	2.98	2.74e+01
		STD	2.65e+01	160	140	4.22	1.78e+02	7.59	9.00e+01
	jDE	Average	1.48568	49.5044	51.9448	1.50584	69.4395	2.35478	11.7425
		STD	4.47652	135.248	141.78	4.10062	144.041	5.78252	39.4469
	DynDE	Average	1.86162	39.5923	23.4921	0.8691	44.6713	1.5624	6.5213
		STD	5.75312	98.6312	94.5314	3.1723	121.7162	6.2149	26.5951
	dopt-aiNET	Average	1.4227	122.4410	98.6688	4.2632	304.5660	12.6334	52.9010
		STD	4.5459	201.6270	196.6950	9.7255	203.2430	55.8386	130.5930
	CPSO	Average	2.677	37.15	36.67	0.7926	67.17	4.881	7.792
		STD	7.055	99.43	97.18	2.775	130.3	15.39	19.21
	CESO	Average	1.86413	28.5925	23.4561	0.84958	44.4145	1.45423	6.4312
		STD	5.6079	98.6229	92.5379	2.95216	119.9201	6.17315	25.6572
F5	MPBDE-SES	Average	**1.50E-04**	**6.56E-06**	**1.92E-06**	**5.55E-06**	**2.3**	**3.60E-11**	**4.66E-06**
		STD	**3.60E-04**	**1.50E-05**	**8.05E-07**	**9.18E-06**	**5.6731**	**8.20E-11**	**8.25E-05**
	DASA	Average	9.55e−01	0.99	0.949	3.92e−01	2.30	4.67e−01	1.11
		STD	3.43	4.05	3.31	1.61	6.36	1.73	3.76
	jDE	Average	0.159877	0.33392	0.3579	0.108105	0.409275	0.229676	0.434294
		STD	1.02554	1.64364	1.83299	0.826746	1.90991	0.935494	2.22792
	DynDE	Average	2.9929	2.9481	2.9125	1.3796	8.4378	2.3049	0.5214
		STD	6.8831	4.7179	5.3886	2.4199	12.1132	3.6182	0.7135
	dopt-aiNET	Average	40.8943	34.4531	34.9420	120.6370	943.2230	480.3370	219.4660
		STD	221.2120	119.8960	115.0250	293.5420	633.3180	610.8020	427.8170
	CPSO	Average	1.855	2.879	3.403	1.095	7.986	4.053	6.527
		STD	5.181	6.787	6.448	4.865	13.81	8.371	22.8
	CESO	Average	2.80304	2.82617	2.81789	1.28678	8.11117	2.22074	3.45078
		STD	6.2714	4.4479	5.25534	2.3017	11.7631	3.53149	11.71356
F6	MPBDE-SES	Average	**0.6872**	**2.0597**	**5.2601**	**1.62E-07**	59.182	54.4473	**4.7113**
		STD	**1.6833**	**5.0452**	**8.649**	**3.97E-07**	29.1442	21.259	**13.94**
	DASA	Average	8.87	37	26.7	9.74	3.79e+01	1.33e+01	1.17e+01
		STD	1.33e+01	122	98.4	2.20e+01	1.18e+02	5.74e+01	3.67e+01
	jDE	Average	6.22948	10.3083	10.954	6.78734	14.9455	7.8028	10.736
		STD	10.4373	13.2307	23.2974	10.1702	45.208	10.9555	14.7267
	DynDE	Average	6.0471	20.2205	19.3782	8.8731	43.3514	12.1798	13.3644
		STD	11.0458	62.2093	67.3585	26.6683	136.9062	25.2617	21.0744
	dopt-aiNET	Average	20.4434	391.1960	456.4410	83.9698	845.8620	482.2070	372.4740
		STD	79.3230	395.4350	405.0380	220.1770	251.2080	434.4210	394.6680
	CPSO	Average	6.725	21.57	27.13	9.27	71.57	23.67	32.58
		STD	9.974	63.51	83.98	24.23	160.3	51.55	76.9
	CESO	Average	5.99334	20.0895	18.5418	7.9072	43.1644	11.8575	13.2205
		STD	10.3175	60.9701	65.6901	24.394	135.0895	24.3906	20.3943

5 Figures

To show the variation in convergence characteristics for different functions we have given the sample convergence graphs for functions F1(number of peaks=10), F1(number of peaks 50), F2, F3, F4,F5, F6 with change type T7 over 300000 Fes. . As mentioned earlier the relative value r(t) is calculated as $f(\vec{X}_{best}(t))/f(\vec{X}^*(t))$ for function F1 and for other functions , r(t) is calculated as $f(\vec{X}^*(t))/f(\vec{X}_{best}(t))$. Here we have basically plotted the ratio r(t) for the above mentioned functions.

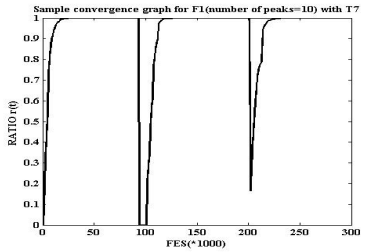


Fig.1. F1(number of peaks=10)with T7

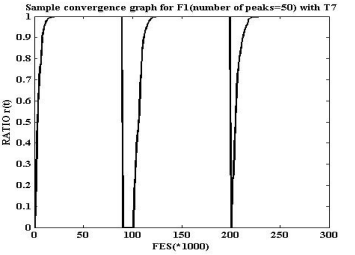


Fig.1.F1(number of peaks=50)with T7

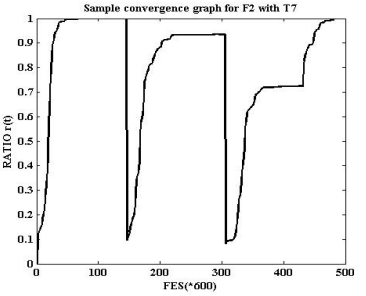


Fig.1. F2 with T7

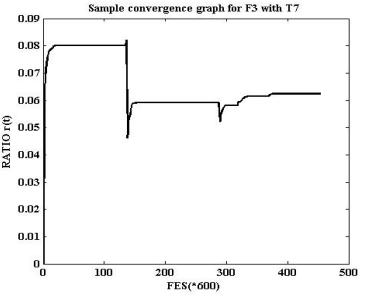


Fig.1. F3with T7

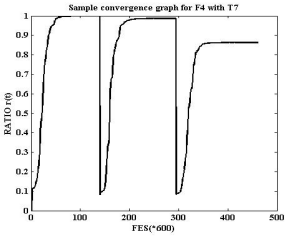


Fig.1. F4with T7

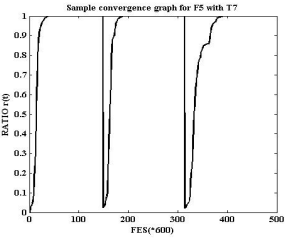


Fig.1. F5with T7

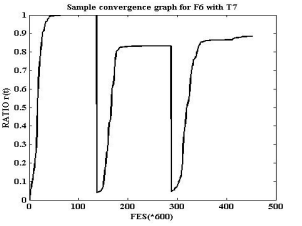


Fig.1. F6with T7

6 Conclusion

The main features of MPBDE-SES algorithm described in this paper is summarized below:

i)It uses classical DE to generate new generation individulals.
ii)Apart from classical DE it uses an interaction scheme to enhance faster convergence.
iii)In addition to classical DE, it uses Quantum and Brownian individual generation rule to generate quantum and brownian individual to increase the diversity and exploration ability of the algorithm.

iv)The algorithm uses an Exclusion Principle to ensure uniform distribution of subpopulations over the entire search space.
v)An aging mechanism has been included to get rid of stagnation.
vi) The statistical summary of the simulation results also shows that MPBDE-SES is far better than other algorithms in terms of performance in dynamic landscape.So we can conclude MPBDE-SES is very good optimizer for DOPs.

References

[1] Storn, R., Price, K.: Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11(4), 341–359 (1997)
[2] Das, S., Suganthan, P.N.: Differential evolution – a survey of the state-of-the-art. IEEE Transactions on Evolutionary Computation 15(1), 4–31 (2011)
[3] Li, C., Yang, S.: A clustering particle swarm optimizer for dynamic optimization. In: Proc. 2009 Congr. Evol. Comput., pp. 439–446 (2009)
[4] Yang, S., Li, C.: A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments. IEEE Trans. on Evolutionary Computation 14(6) (December 2010)
[5] Liu, L., Wang, D., Yang, S.: Compound Particle Swarm Optimization in Dynamic Environments. In: Giacobini, M., Brabazon, A., Cagnoni, S., Di Caro, G.A., Drechsler, R., Ekárt, A., Esparcia-Alcázar, A.I., Farooq, M., Fink, A., McCormack, J., O'Neill, M., Romero, J., Rothlauf, F., Squillero, G., Uyar, A.Ş., Yang, S. (eds.) EvoWorkshops 2008. LNCS, vol. 4974, pp. 616–625. Springer, Heidelberg (2008)
[6] Liu, L., Wang, D., Yang, S.: Particle swarm optimization with composite particles in dynamic environments. IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics 40(6) (December 2010)
[7] Mendes, R., Mohais, A.S.: DynDE: a differential evolution for dynamic optimization problems. In: Proc. of IEEE Congress on Evolutionary Computation, vol. 2, pp. 2808–2815 (2005)

Clustered Parent Centric Normal Cross-Over for Multimodal Optimization

Rohan Mukherjee[1], Rupam Kundu[1], and Swagatam Das[2]

[1] Department of Electronics and Telecommunication Engineering, Jadavpur University
[2] Electronics and Communications Sciences Unit, Indian Statistical Institute,
Kolkata – 700 108, India
{rohan.mukherjii,rupam2422}@gmail.com,
swagatam.das@isical.ac.in

Abstract. Genetic algorithms are mainly modeled on basic four steps of parent selection, crossover, offspring evaluation and replacement. It is not possible to model this for direct application to multimodal landscapes. In this paper we propose a novel algorithm in which GA named Parent Centric Normal Crossover is modified and works on a clustered population to tackle multimodal problems. We suggest a dynamic clustering scheme to maintain stable yet variable number of clusters of variable size which can tackle multimodal landscapes, and a Crossover Rate operator in GA for controlled convergence to tackle complex multimodal functions. The algorithm has been tested over widely used benchmarks from single dimension to complex composite functions and compared with other State of the art EAs. The results clearly prove C-SPC-PNX to be a robust multimodal optimization technique.

1 Introduction

Optimization technique that involves simultaneous detecting of the multiple global and local peaks in a landscape is multimodal optimization[16]. Evolutionary strategies like classical DE delivers effective performance in unimodal landscapes but fails to simultaneously optimize for multiple peaks. So modifications in DE to make it functionally effective in multi-modal environment have been a challenge in the recent times. Presently strong algorithms with special techniques such as 'niching'[2] are used to maintain multiple groups in single population to locate multiple optimas.The niching technique includes 'crowding'[3], 'speciation'[6], 'fitness sharing'[4], , 'distance'[17] and 'restricted tournament selection'[5]. But multimodal optimization still demands robust yet less complex techniques which will ensure an effective search with higher accuracy.

Through this article we propose a new technique for multimodal optimization inspired from GA for an efficient multimodal search procedure with data point clustering, local search based mutation, selective crossover and a probabilistic selection. Data point clustering ensures selection of proper locality of each solution such that efficient local search within the territory of each optimum is achieved. If this

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 276–284, 2012.

is properly achieved each multimodal peak will have different cluster set to converge on it. Thus clustering form the basic pillar for initiating multimodal search.
A modified local search technique by GA as proposed searches the locality by its local search based scheme for mutation. A selective crossover selects some part of the new population and keeps others from the old for a better Genetic Mutation. A probabilistic crowding keeps the solutions near its locality thus omitting any chance of escaping to other locality and makes decision based on fitness and distance both. An efficient local search technique found in this GA thus makes the clustered population fit for multimodal landscapes.

2 Clustered SPC-PNX

2.1 Fuzzy c-Means Clustering

Fuzzy c-means[12] is a data clustering technique in which each member has a definite membership function to belong to a certain group or cluster. Thus every data point belongs to every data clusters but with a different degree of belonging or membership to that cluster. A data point belonging to that cluster will have a high degree of belonging while a particle that is far away from the cluster centroid will a feeble membership to that cluster. The details can be found in [12].

Membership function is set for all data points for all clusters. A data point gets selected into that cluster which it has highest membership into. Thus some clusters may not find any constituent data point at all. Some cluster may have high density of points whereas some clusters may have very few. Thus though a maximum number of possible clusters is declared at the start of the algorithm the number of clusters at any iteration may take a value below the maximum. These clusters handle different cluster size and hence a dynamic clustering is achieved to suit any problem accurately.

2.2 Parent Centric Normal Crossover

SPC-PNX(Parent Centric Normal Crossover)[13] used in this paper, is a steady-state real-parameter GA(Genetic Algorithm) variant. The process involves between two randomly selected parents from the population resulting in λ offsprings, and crossover operation governs this process. Subsequently the offsprings produced are tested in terms of their objective functional value, which are thereby combined with the parents in order to keep the population size same and independent on the process of replacement. A single GA framework is devised on the following four basic steps namely successive selection, crossover, fitness evaluation and replacement.

Step1: For successive selection two individuals are randomly chosen from the pool of individuals. The uniformity in selection and non-consideration of fitness keeps scope for proper exploration and exploitation.

Step2: A parent centric crossover operator PNX is used in the second stage of the algorithm. The crossover operation takes place without any attached bias to any

direction, resulting in an explorative coverage of the fitness landscape. The spread of population depends on the between the parents and thus variation of accounts to the self-adaptive nature of the crossover operator. The process of selection of λ offspring does occur in random selection of either of two crossover methods, for all dimensions. These two operators which occur with equal probability are:

$$y_j^{(1)} = N(x_j^{(1)}, | x_j^{(2)} - x_j^{(1)} | / \eta) \tag{1}$$

$$y_j^{(2)} = N(x_j^{(2)}, | x_j^{(2)} - x_j^{(1)} | / \eta) \tag{2}$$

In addition to this we have added a crossover rate operator CR. By this we have

$$\begin{aligned} &y_j^{(k)} = N(x_j^{(k)}, | x_j^{(i)} - x_j^{(k)} | / \eta) \text{ if rand<CR or j==jrand} \\ &y_j^{(k)} = x_j^{(k)} \text{ otherwise} \end{aligned} \tag{3}$$

Where jrand is an integer in [1,d], $N(\mu,\sigma)$ is a random number drawn from a Normal distribution with mean μ and standard deviation σ, $x_j^{(i)}$ is the j-th component of the i-th parent and η is a tunable parameter. An increased value of η decreases the density of particles around the parents. The produced offsprings are limited within appropriate bounds.

Step3: The fitness values of the offsprings are calculated from the objective function under study.

Step4: Replacement occurs through a scaled probabilistic crowding scheme. A traditional SPC-PNX model uses the closest of randomly selected group of NREP individuals to compare with the offspring by scaled probabilistic selection[5] of the offsprings with the closest of NREP individuals. This provides an excellent platform for thorough exploration of the search space, as the fittest particle is not always selected, thus preventing any premature convergence. Thus efficient crowding schemes like this can be used as a prospective replacement procedure. The mathematical formulae used to determine the probability of offspring and closest particle among the two parents are:

$$p(\overrightarrow{x^{ofp}}) = \frac{f_{best} - f(\overrightarrow{x^{ofp}})}{2 f_{best} - f(\overrightarrow{x^{ofp}}) + f(\overrightarrow{x^{cst}})} \tag{4}$$

$$p(\overrightarrow{x^{cst}}) = \frac{f_{best} - f(\overrightarrow{x^{cst}})}{2 f_{best} - f(\overrightarrow{x^{ofp}}) + f(\overrightarrow{x^{cst}})} \tag{5}$$

where f_{best} is the function value of the best individual among the offspring and the parents.

2.3 Fuzzy C-Means Clustering with SPC-PNX

The proposed algorithm first uses the data points to be clustered based on distance by fuzzy-cmeans which form sets of clusters. Some clusters may remain empty if the membership function of each particle to lie in that cluster is outperformed by other clusters. Moreover each cluster can have any number of members in it. Thus dynamic clustering is possible.

The algorithm can be summarized as:

1. Initialize the population of NP number of trial solutions.
2. Specify NC_{max} number of maximum clusters possible.
3. While termination criterion not satisfied
4. Form data clustering by FCM to form NC(<=NC_{max}) cluster sets.
5. For i=1:NC
   Select parents from the population .
   Perform crossover for producing λ offsprings.
   Evaluate fitness of offsprings.
   Perform probabilistic tournament between offspring and closest of the parents.
   End For
6. Stop if termination criterion is satisfied. Otherwise go to *Step 2*.

3 Performance Measure

The performance of the algorithm on the given test functions (Section 4C Table 1) are tested on the basis of two criterion. All results are averaged over 25 independent runs and compared with other state of arts as : CDE[3], ShDE[4], SDE[6], FER-PSO[7], SPSO[11], r2pso[7], r3pso[7], r2pso-lhc[7], r3pso-lhc[7], SCMA-ES[8].

I) Average number of peaks found for each environment [10].

II) Ability to detect local optima's.[14].

4 Experimental Results

In this section we have presented our test results for the proposed algorithm over commonly used Niching Functions. The functions used for our experiment is given in Table 1 while the parameter settings for the experiment are given in Table 2. The simulated results are shown in Table 3, Table 4, and Table 5. The results show that CSPCPNX performs better than other SOAs over both small dimensional and complex Composite function problems and is efficient in simultaneous detection of both local and global extremums to lend it to diverse multimodal environments.

A. Experimental Setup

All the experiments have been performed on machines having the following specifications: CPU: 2.26 GHz Intel Core i3 RAM: 4 GB and MATLAB 2010a edition.

B. Final Population Plot

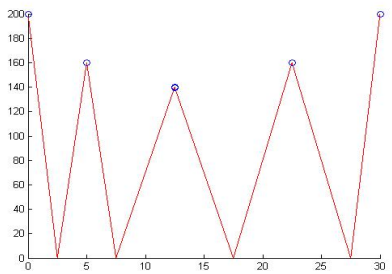


Fig. 1. Final Population for E1-F3

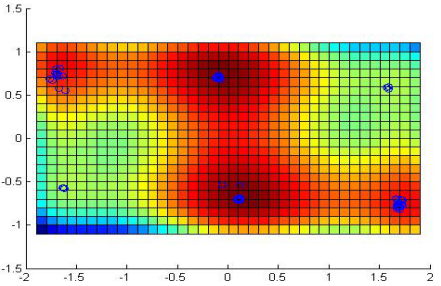


Fig. 2. Final Population for E1-F9

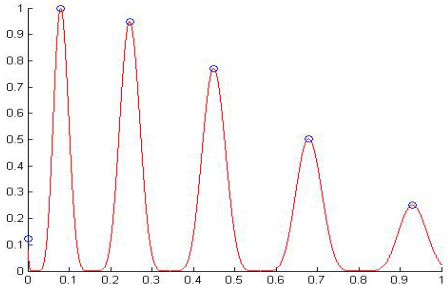


Fig. 3. Final Population for E1-F7

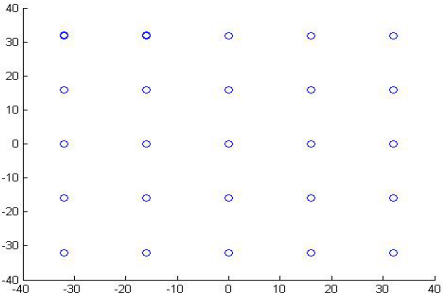


Fig. 4. Final Population for E1-F10

C. Test Functions

Table 1. Test Functions for Experiment (CF: Composition Function)[14]

Test Function Set 1		**Test Function Set 2 [15]**	
Test Function name	**Number of Global peaks/Dimension**	**Test Function name**	**Number of Global peaks/Dimension**
E1-F1: Two-peak trap	1/1	E1-F15: CF 1	8/10
E1-F1: Central two-peak trap	1/1	E1-F16: CF 2	6/10
E1-F3: Five-uneven-peak trap	2/1	E1-F17: CF 3	6/10
E1-F4: Equal Maxima	5/1	E1-F18: CF 4	6/10
E1-F5: Decreasing Maxima	1/1	E1-F19: CF 5	6/10
E1-F6: Uneven Maxima	5/1	E1-F20: CF 6	6/10
E1-F7: Uneven Decreasing Maxima	1/1	E1-F21: CF 7	6/10
E1-F8: Himmelblau's function	4/2	E1-F22: CF 8	6/10
E1-F9: Six-hump camel back	2/2	E1-F23: CF 9	6/10
E1-F10: shekel's foxholes	1/ 2	E1-F24: CF 10	6/10
E1-F11: 2D inverted Shubert function	18/2	E1-F25: CF 11	8/10
E1-F12: 1D inverted Vincent function	6/1	E1-F26: CF 12	8/10
E1-F13: 2D inverted Vincent function	36/2	E1-F27: CF 13	10/10
E1-F14: 3D inverted Vincent function	216/3	E1-F28: CF 14	10/10
		E1-F29: CF 15	10/10

D. Population Size, Maximum Number of Evaluations and Parameter Setting

The required level of accuracy, the allowed maximum number of functional evaluations and population size for function set1 are shown below in Table 2. Maximum number of clusters can be set at approximately [Population Size/10] but as far as optimization for a certain level of accuracy within a fixed number of functional evaluations is concerned, for best results maximum number of clusters can be set as shown in table. It can be tuned by running algorithm once or twice by an approximate knowledge on number of peaks in the landscape. SPC-PNX parameters η and CR are set at 3 and 0.8 respectively. For set 2 a population size of 600 is used with maximum functional evaluation fixed at 300,000[14].

Table 2. Parameter Settings

Function no.	ε	NC_{max}	Population size	Function Evaluations
E1-F1	0.05	5	50	10000
E1-F1	0.05	5	50	10000
E1-F3	0.05	5	50	10000
E1-F4	0.000001	5	50	10000
E1-F5	0.000001	5	50	10000
E1-F6	0.000001	5	50	10000
E1-F7	0.000001	5	50	10000
E1-F8	0.0005	5	50	10000
E1-F9	0.000001	5	50	10000
E1-F10	0.000001	5	50	10000
E1-F11	0.05	30	250	100000
E1-F12	0.0001	10	100	20000
E1-F13	0.001	40	500	200000
E1-F14	0.001	80	1000	400000

Table 3. Average of optima found in locating both global and local peaks

Fnc	CDE	ShDE	SDE	FER-PSO	SPSO	r2pso	r3pso	r2pso-lhc	r3pso-lhc	SCMA-ES	C-SPC- PNX
E1-F1	**2(2.5)**	**2 (2.5)**	1.84 (5)	1.48 (9)	1.44 (11)	1.72 (6)	1.48 (9)	1.48 (9)	1.52 (7)	**2(2.5)**	**2(2.5)**
E1-F2	**2(2)**	1.84 (5)	1.68 (8)	1.88 (4)	1.72 (7)	1.36 (10)	1.24 (11)	1.52 (9)	1.76 (6)	**2(2)**	**2(2)**
E1-F3	4.44 (2)	3.6 (3)	3.04 (5)	0.64 (10)	3.08 (4)	0.8 (9)	0.4 (11)	3(6)	2.16 (7)	2(8)	**5(1)**
E1-F5	4.28 (4)	3.12 (5)	1.52 (7)	1(9)	**5 (1.5)**	1(9)	1(9)	4.52 (3)	2.8(6)	0.48 (11)	**5(1.5)**
E1-F10	12.5 (8)	24.96 (2)	1.32 (10)	5.16 (9)	24.9 (3)	24.4 (6)	24.3 (7)	24.8 (4)	24.6 (5)	0.88 (11)	**25(1)**
Total Rank	18.5	17.5	35	41	26.5	40	47	31	31	34.5	**8**

Table 4. Average number of peaks found for test function set 1

Fnc	CDE	ShDE	SDE	FER-PSO	SPSO	r2pso	r3pso	r2pso-lhc	r3pso-lhc	SCMA-ES	C-SPC-PNX
E1-F1	**1(3)**	**1(3)**	**1(3)**	0.72(8)	0.48 (11)	0.76(7)	0.84(6)	0.56 (10)	0.6(9)	**1(3)**	**1(3)**
E1-F2	**1(3.5)**	**1(3.5)**	**1(3.5)**	**1(3.5)**	0.44 (10.5)	0.88(8)	0.96(7)	0.44 (10.5)	0.56(9)	**1(3.5)**	**1(3.5)**
E1-F3	**2(2.5)**	**2(2.5)**	1.96(5)	0.8(6)	0.24 (11)	0.48 (9.5)	0.6 (7.5)	0.48 (9.5)	0.6 (7.5)	**2(2.5)**	**2(2.5)**
E1-F4	3.84(9)	3.28 (10)	4.72 (8)	4.84(7)	4.88 (5.5)	4.92 (3.5)	4.88 (5.5)	**5(1.5)**	4.92 (3.5)	0.04 (11)	**5(1.5)**
E1-F5	0.72 (10)	0.44 (11)	**1(5)**	**1(5)**	**1(5)**	**1(5)**	**1(5)**	**1(5)**	**1(5)**	**1(5)**	**1(5)**
E1-F6	3.96 (9)	3.28 (10)	4.6 (8)	**5(1.5)**	4.92 (3.5)	4.88 (5.5)	4.72 (7)	4.92 (3.5)	4.88 (5.5)	0(11)	**5(1.5)**
E1-F7	0.6 (10)	0.4(11)	**1(4.5)**	**1(4.5)**	**1(4.5)**	**1(4.5)**	**1(4.5)**	**1(4.5)**	**1(4.5)**	0.96(9)	**1(4.5)**
E1-F8	0.32 (10)	0.16 (11)	3.72 (2)	3.68 (3)	0.84 (9)	2.92 (7)	2.76 (8)	3(6)	3.12 (5)	3.44 (4)	**4(1)**
E1-F9	0.04 (9.5)	0.04 (9.5)	**2(2)**	1.96 (4)	0.08 (11)	1.44 (8)	1.56 (5.5)	1.56 (5.5)	1.48 (7)	**2(2)**	**2(2)**
E1-F10	0.52 (9)	0.96(3)	0.32 (10)	**1(1.5)**	0.56 (8)	0.88 (4)	0.76 (5)	0.72 (6)	0.6(6)	0.04 (11)	**1(1.5)**
E1-F11	**17.7(1)**	16.56 (3)	12.4 (5)	17.4 (2)	8.52 (10)	15.2 (8)	15.6 (7)	15.1 (9)	16.2 (6)	2.16 (11)	16.32(4)
E1-F12	5.56(4)	5.6 (2.5)	4.88 (10)	5.36 (6.5)	5.6 (2.5)	5.52 (5)	5.16 (9)	5.36 (6.5)	5.28 (8)	1.52 (11)	**5.8(1)**
E1-F13	33.8(2)	**35.92** (1)	22.8 (7)	23.6 (5)	25.7 (4)	21.8 (10)	22.2 (9)	22.5 (8)	23.1 (6)	1.4 (11)	28(3)
E1-F14	152(2)	**197.8** (1)	50.6 (6)	68.6 (5)	70.1 (4)	40.6 (10)	45.4 (7)	42.2 (9)	43.3 (8)	0.04 (11)	146(3)
Total Rank	84.5	82	77	62.5	99.5	94.5	92.5	94.5	90	95	**37**

Table 5. Average number of peaks found for test function set 2

Fnc	CDE	ShDE	SDE	FER-PSO	SPSO	r2pso	r3pso	r2pso-lhc	r3pso-lhc	SCMA-ES	C-SPC-PNX
E1-CF1	0(8)	0(8)	1.79 (3)	1.08 (4)	0(8)	0(8)	0(8)	0(8)	0(8)	2(2)	**4(1)**
E1-CF2	1.2 (4.5)	1.1 (6)	1.2 (4.5)	2(2)	0(9)	0(9)	0(9)	0(9)	0(9)	1.9 (3)	**3(1)**
E1-CF3	0.7 (6)	1.11 (5)	1.5 (4)	2.5 (3)	0(9)	0(9)	0(9)	0(9)	0(9)	2.7 (2)	**4(1)**
E1-CF4	0(7)	0(7)	0(7)	0(7)	0(7)	0(7)	0(7)	0(7)	0(7)	0.2 (2)	**3(1)**
E1-CF5	1.1 (6)	1.3 (4.5)	1.3 (4.5)	2(2)	0(9)	0(9)	0(9)	0(9)	0(9)	1.9 (3)	**3(1)**
E1-CF6	0(8)	0(8)	1.4 (3)	1.2 (4)	0(8)	0(8)	0(8)	0(8)	0(8)	2.6 (2)	**3(1)**
E1-CF7	0(8)	0(8)	1(2.5)	0.5 (4)	0(8)	0(8)	0(8)	0(8)	0(8)	1(2.5)	**3(1)**
E1-CF8	0(8)	0(8)	1.4 (4)	1.5 (3)	0(8)	0(8)	0(8)	0(8)	0(8)	2.3 (2)	**3(1)**
E1-CF9	0(8)	0(8)	1.8 (2)	1.5 (4)	0(8)	0(8)	0(8)	0(8)	0(8)	1.7 (3)	**3(1)**
E1-CF10	0(8)	0(8)	1.1 (3.5)	1.1 (3.5)	0(8)	0(8)	0(8)	0(8)	0(8)	1.2 (2)	**1.5 (1)**
E1-CF11	0 (7.5)	0 (7.5)	1.3 (2)	0 (7.5)	0 (7.5)	0 (7.5)	0 (7.5)	0 (7.5)	0 (7.5)	0.7 (3)	**3(1)**

Table 5. (*continued*)

E1-CF12	0(8)	0(8)	1.6 (3.5)	1.6 (3.5)	0(8)	0(8)	0(8)	0(8)	0(8)	1.7 (2)	**2.3(1)**
E1-CF13	0(8)	0(8)	0.9 (3)	0.3 (4)	0(8)	0(8)	0(8)	0(8)	0(8)	1.4 (2)	**2.5(1)**
E1-CF14	0(8)	0(8)	**1(2.5)**	**1(2.5)**	0(8)	0(8)	0(8)	0(8)	0(8)	**1(2.5)**	**1(2.5)**
E1-CF15	0(8)	0(8)	1.6 (3)	1.2 (4)	0(8)	0(8)	0(8)	0(8)	0(8)	2(2)	**3(1)**
Total Rank	112	110	52	56	121.5	121.5	121.5	121.5	119.5	35	**16.5**

5 Conclusion

GA is a widely used optimization technique. Various modifications have been applied to conventional GA strategies improvising it. We integrated a modified GA with clustering to handle typical multi modal landscapes. Clearly the algorithm with its clustering and probabilistic selection from the parents is efficient in local search. We showed that it can efficiently track global optimums over a specified level of accuracy, is apt to handle complex landscapes and can simultaneously detect global and local peaks. Experimental Results clearly show that C-SPC-PNX outperform other state-of-the-arts which clearly prove it as a strong multimodal optimization strategy.

References

[1] Mahfoud, S.W.: A comparison of parallel and sequential niching methods. In: Proceedings of 6th International Conference on Genetic Algorithms, Pittsburg, USA, July 15-19, pp. 136–143 (1995) ISBN 155860-370-0

[2] Singh, G., Deb, K.: Comparison of multimodal optimization algorithms based on evolutionary algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1305–1312. ACM Press, Seattle (2006)

[3] Thomsen, R.: Multimodal optimization using Crowding-based differential evolution. In: Proceedings of the IEEE 2004 Congress on Evolutionary Computation, pp. 1382–1389 (2004)

[4] Goldberg, D.E., Richardson, J.: Genetic algorithms with sharing for multimodal function optimization. In: Proceedings of the Second International Conference on Genetic Algorithms, pp. 41–49 (1987)

[5] Harik, G.R.: Finding multimodal solutions using restricted tournament selection. In: Proceedings of the 6th International Conference on Genetic Algorithms, San Francisco, pp. 24–31 (1995)

[6] Li, X.: Efficient differential evolution using speciation for multimodal function optimization. In: Proceedings of the Conference on Genetic and Evolutionary Computation, Washington, DC, USA, pp. 873–880 (2005)

[7] Li, X.: Niching without niching parameters: particle swarm optimization using a ring topology. IEEE Transactions on Evolutionary Computation 14 (February 2010)

[8] Shir, O.M., Emmerich, M., Back, T.: Adaptive niche radii and niche shapes approaches for niching with the CMA-ES. Evolutionary Computation 18, 97–126 (2010)
[9] Deb, K.: Genetic algorithms in multimodal function optimization, the Clearing house for Genetic Algorithms. M.S thesis and Rep. 89002, Univ. Alabama, Tuscaloosa (1989)
[10] Stoean, C., Preuss, M., Stoean, R., Dumitrescu, D.: Multimodal optimization by means of a topological species conservation algorithm. IEEE Trans. Evol. Comput. 14(6), 842–864 (2010)
[11] Daniel, P., Li, X.: Locating and Tracking Multiple Dynamic Optima by a Particle Swarm Model Using Speciation. IEEE Transactions on Evolutionary Computation 10(4), 440–458 (2006)
[12] Sreenivasarao, V., Vidyavathi, S.: Comparative Analysis of Fuzzy C-Mean and Modified Fuzzy Possibilistic C-Mean Algorithms in Data Mining. International Journal of Computer Science and Technology, IJCST 1(1), 104–106 (2010)
[13] Ballester, P.J., Stephenson, J., Carter, J.N., Gallagher, K.: Real-Parameter Optimization Performance Study on the CEC-2005 benchmark with SPC-PNX. In: Congress on Evolutionary Computation, CEC (2005), doi:10.1109/CEC.2005.1554724
[14] Qu, B.Y., Suganthan, P.N.: Differential Evolution with Neighborhood Mutation for Multimodal Optimization. Accepted by Transactions on Evolutionary Computation, doi:10.1109/TEVC.2011.2161873
[15] Qu, B.Y., Suganthan, P.N.: Novel Multimodal Problems and Differential Evolution with Ensemble of Restricted Tournament Selection. In: IEEE Congress on Evolutionary Computation, Barcelona, Spain (July 2010)
[16] Das, S., Maity, S., Qu, B.-Y., Suganthan, P.N.: Real-parameter evolutionary multimodal optimization — A survey of the state-of-the-art. Swarm and Evolutionary Computation 1(2), 71–88 (2011)
[17] Qu, B.-Y., Suganthan, P.N., Das, S.: A Distance-Based Locally Informed Particle Swarm Model for Multi-modal Optimization. Accepted by Trans. on Evolutionary Computation, doi:10.1109/TEVC.2012.2203138

Stock Indices Prediction Using Radial Basis Function Neural Network

Minakhi Rout[1,*], Babita Majhi[2], Usha Manasi Mohapatra[1], and Rosalin Mahapatra[1]

[1] Dept. of CSE /CA/IT, ITER, Siksha O Anusandhan (Deemed to be) University, Bhubaneswar, India
{minakhi.rout,ushamanasi,rosalinmahapatra}@gmail.com
[2] Dept. of CSIT, G.G. Vishwavidyalaya, Central University, Bilaspur, India
babita.majhi@gmail.com

Abstract. Aim of the paper is to efficiently predict the stock market data for future days ahead using Radial Basis Function (RBF) neural network. DJIA and S&P 500 stock indices have been taken to simulate the RBF model and also comparison has been done with results obtained from Functional Link Artificial Neural Network(FLANN) and Multilayer Perceptron, (MLP). From the simulation result it is observed that the proposed model is giving better results than other two neural network models interms of prediction accuracy.

Keywords: stock market prediction, multilayer perceptron (MLP), functional link artificial neural network (FLANN) and radial basis function neural network (RBF).

1 Introduction

Stock Market prediction is carried to determine the appropriate time to buy, hold or sell. As it affects economic condition of a country, it always becomes a popular field of research. As more and more money is being invested by the investors in the stock market, the risk associated with it is also increases. Thus prediction of stock prices is very essential. But it is not an easy task to predict stock market prices because the stock market indices are nonlinear, dynamic, nonparametric, complicated and chaotic in nature. In addition, stock market's movements are affected by many macro economical factors such as political events, general economic conditions, firms' policies, investors' expectations, institutional investors' choices and movement of other stock market and psychology of investors. Stock market forecasting research offers many challenges and opportunities, with the forecasting of individual stocks or indexes focusing on forecasting future market prices. Stock prediction is a difficult task due to the nature of the stock data which is noisy and time varying. People are interested in predicting the stock trend, index or price. Therefore, many different methods and techniques have been reported. These methods include artificial neural network (ANN), linear and multi-linear regression (LR, MLR), genetic algorithm

* Corresponding author.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 285–293, 2012.

(GA), support vector machine (SVM) etc. A fusion model combining Hidden Markov Model(HMM), Artificial Neural Networks(ANN) and Genetic Algorithms(GA) to forecast financial market behavior is proposed in [1]. The comparison shows the forecasting ability of the fusion model is as good as that of ARIMA model. A three-layer multi-input Legendre neural network with a random time strength function is introduced in [2] for stock market forecasting. Hybridization of fractal feature selection method and support vector machine to predict the direction of daily stock price index is reported[3]. Fractal feature selection method is suitable for solving the nonlinear problem and it can exactly spot how many important features to choose. To evaluate the prediction accuracy of this method, the paper compares its performance with other five commonly used feature selection methods. [4] investigates the effectiveness of a hybrid approach based on the artificial neural networks (ANNs) for time series properties. Adaptive time delay neural networks (ATNNs) and the time delay neural networks (TDNNs), with the genetic algorithms (GAs) is used in detecting temporal patterns for stock market prediction tasks. The results show that the accuracy of the integrated approach proposed in this study is higher than that of the standard ATNN, TDNN and the recurrent neural network (RNN). The adaptive linear combiner based models optimized by using adaptive bacterial foraging (ABFO) and BFO algorithms is suggested for stock indices forecasting[5]. The short and long term prediction performance of these models are evaluated with test data and the results obtained are compared with those obtained from the genetic algorithm (GA) and particle swarm optimization (PSO) based models. Two data mining approaches, apriori algorithm and k-means has been used for forecasting Taiwan stock market[6]. By doing so, this research finds that different possible portfolio of stock categories investment can be implemented in the Taiwan stock market. The effectiveness of neural network models is studied in [7] which are known to be dynamic and effective in stock-market predictions. The models analysed are multi-layer perceptron (MLP), dynamic artificial neural network (DAN2) and the hybrid neural networks which use generalized autoregressive conditional heteroscedasticity (GARCH) to extract new input variables. The comparison of three different feature selection methods, Principal Component Analysis (PCA), Genetic Algorithms (GA), and decision trees (CART) has been done[8] for stock prediction. The simulation results show that combining multiple feature selection methods can provide better prediction performances than using single feature selection methods. The IBCO–BP model [9]for prediction of stock indices is developed. Its structure of the model is basically an adaptive BP ANN whose weights are updated using IBCO tool. Performance comparison with the BP model simulated indicates that the developed model offers less computational complexity, better prediction accuracy, and less training time. The fuzzy neural network optimized by differential evolution [10] is applied to the problem of stock market forecasting. Integration of nonlinear independent component analysis(NLICA) and neural network has been proposed in [11] for Nikkei 225 closing index and Shanghai B-share closing index forecasting. It concluded that the proposed NLICA-BPN model can be a good alternative for Asian stock market index forecasting since it can generate good forecasting performance.

From the literature survey it is observed that mainly neural network models are used for forecasting of the different stock markets as it is a nonlinear model. The radial basis function network is a feed forward neural network with a single hidden layer which computes the distance between input pattern and the center [12]. The RBF network is suitable for solving function approximation, system identification, pattern classification and exchange rate prediction because of its simple topological structure and their ability to learn in an explicit manner [13- 15]. In this paper RBF neural network is used for forecasting if two important stock indices of USA, S&P500 and DJIA.

The organization of the paper proceeds as follows. Section 2 explain the development of RBF based stock market forecasting model. Section3 describes the data collection and process for feature extraction. This section also provides the formulae of computing the technical indicators. Simulation study of the proposed model and discussion on results are carried out in Section 4. Finally the conclusion of the investigation is provided in Section 5.

2 Development of RBF Neural Network Based Stock Market Forecasting Model

A radial basis function network [12] is an artificial neural network that uses radial basis functions as activation functions. The basic architecture of RBF network based forecasting model is shown in Fig. 1. Here $x(n)$ is the input to the network and ϕ represents the radial basis function that perform the nonlinear mapping and M represents the total number of hidden units. Each node has a center vector c_k and spread parameter σ_k, where $k = 1, 2, ..., M$. The radial basis functions are represented by $\phi(\|x, c\|)$, where $\|\bullet\|$ represents the Euclidean norm. The radial basis functions which are generally used in various applications are Gaussian functions as given in (1)

$$\phi(r) = \exp\left(-\frac{r^2}{2\sigma^2}\right) \tag{1}$$

The estimated output of the network for n th pattern is

$$y(n) = \sum_{k=1}^{M} w_k(m)\phi(\bar{x}(n), c_k(m), \sigma_k(m)) \tag{2}$$

where

$$\phi(x(n), c_k(m), \sigma_k(m)) = \exp\left(-\frac{\|\bar{x}(n) - c_k(m)\|^2}{\sigma_k^2(m)}\right) \tag{3}$$

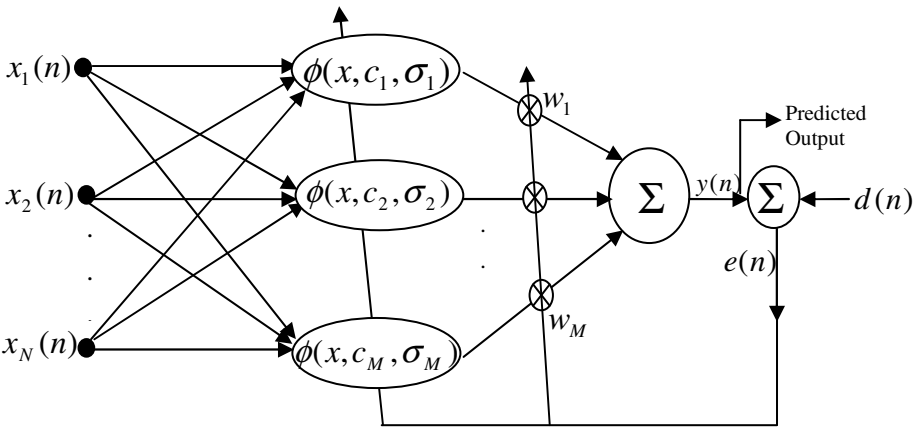


Fig. 1. Schematic diagram of RBFNN based forecasting model

The error for the n th pattern is obtained by

$$e(n) = d(n) - \sum_{k=1}^{M} w_k(m)\phi(x(n), c_k(m), \sigma_k(m))) \tag{4}$$

where $d(n)$ is the target value. As the Gaussian function is chosen as the radial basis function (4) can be rewritten as

$$e(n) = d(n) - \sum_{k=1}^{M} w_k(m) \exp\left(- \frac{\left\| x(n) - c_k(m) \right\|^2}{\sigma_k^2(m)} \right) \tag{5}$$

The cost function is defined as

$$\xi = \frac{1}{2} \sum_{n=1}^{n_1} e^2(n) \tag{6}$$

where n_1 is the number of training patterns. It is required to adjust the free parameters such as weight, center and spread so as to minimize ξ. According to the gradient descent algorithm the free parameters at m th epoch are updated using

$$w_k(m+1) = w_k(m) - \mu_m \frac{\partial \xi}{\partial w_k(m)} \tag{7}$$

$$c_k(m+1) = c_k(m) - \mu_c \frac{\partial \xi}{\partial c_k(m)} \tag{8}$$

$$\sigma_k(m+1) = \sigma_k(m) - \mu_\sigma \frac{\partial \xi}{\partial \sigma_k(m)} \tag{9}$$

where μ_w, μ_c and μ_σ are learning parameters within the range of 0 to 1and $k = 1, 2,....., M$. Finally the update equations are defined as

$$w_k(m+1) = w_k(m) + \sum_{n=1}^{n_1} \mu_w e(n) \phi(x(n), c_k(m), \sigma_k(m)) \quad (10)$$

$$c_k(m+1) = c_k(m) + \sum_{n=1}^{n_1} \mu_c \frac{e(n) w_k(m)}{\sigma_k^2(n)} \phi(x(n), c_k(m), \sigma_k(m)) [x(n) - c_k(m)] \quad (11)$$

$$\sigma_k(m+1) = \sigma_k(m) + \sum_{n=1}^{n_1} \mu_\sigma \frac{e(n) w_k(m)}{\sigma_k^3(m)} \phi(x(n), c_k(m), \sigma_k(m)) [\|x(n) - c_k(m)\|^2] \quad (12)$$

3 Data Collection and Preprocessing of the Data

The data set for stock market prediction experiments have been collected for Standard's and Poor's 500 (S&P 500), USA and Dow Jones Industrial Average (DJIA), USA. The experimental data used here consists of technical indicators and daily closing price of the indices. The total number of samples for the stock indices is 3228 trading days, from 3rd January 1994 to 23rd October 2006. Each sample

Table 1. Details of technical indicators and their formulae

Technical indicators used	Formula
Exponential moving average (EMA) (3 numbers)	$(P \times A) + (\text{Previous } EMA \times (1 - A))$ $A = 2/(N+1)$ P – current price, A – smoothing factor, N – time period (EMA10, EMA20 and EMA30 are calculated using the given formula)
Accumulation/distribution oscillator (ADO)	$\frac{(CP - LP) - (HP - CP)}{(HP - LP) \times (Period's\ volume)}$ CP – closing price, HP – highest price, LP – lowest price
Stochastic indicator (STI)	$\%K = \frac{(Today's\ Close - Lowest\ low\ in\ K-period)}{Highest\ high\ in\ K-period - Lowest\ low\ in\ K-period} \times 100$ % D = SMA of % K for the period
Relative strength index (RSI)	$RSI = 100 - \frac{100}{1 + (U/D)}$ U – total gain/n, D – total losses/n, n – number of RSI period (RSI9 and RSI14 are calculated using the given formula)
Price rate of change (PROC)	$\frac{(Today's\ Close - Close\ price\ X-period\ ago)}{Close\ price\ X-period\ ago} \times 100$
Closing price acceleration (CPACC)	$\frac{(Close\ Price - Close\ price\ X-period\ ago)}{Close\ price\ X-period\ ago} \times 100$
High price acceleration (HPACC)	$\frac{(High\ Price - High\ price\ N-period\ ago)}{High\ price\ N-period\ ago} \times 100$

consists of the closing price, opening price, lowest price, highest price and the total volume of stocks traded for the day. Ten technical indicators [5] are selected as feature subsets by the review of domain experts and prior research. These are computed from the raw data as indicated in the Table 1.

The data are splitted into two sets – training and testing sets. The training set consists of 2510 patterns and the rest is set aside for testing. All the inputs are normalized to values between -1 and +1.

4 Simulation Results and Discussion

Training of the forecasting models are carried out using RBFNN as described in Section 2 and the optimized weight values of the model are obtained through simulation. The detail procedure of proposed model is as follows :

Step-1 The features extracted from past stock data are given as the input to the model as shown in Fig.1.

Step-2 The output of hidden layer is calculated using (3) .

Step-3 Finally the output of hidden layer is weighted and summed to obtain the output of the model using (2).

Step-4 Error is calculated by using (4)

Step-4 After application of all training inputs to the RBF predictor, the parameters of the model are updated using
(10) - (12).

Step-5 The above process will repeated until minimum MSE is reached.

Step-6 Then testing is carried out using rest of the input in same manner as described above and MAPE is obtained using (13).

The performance of the models is evaluated by comparing with known test data. To compare the performance of the proposed models, other two neural network models, FLANN and MLP are also simulated. and the learning characteristics and test results of those are also presented. The experiments carried out in this study to test the performance of the model for predicting the closing price of the index for 1 day, 7 and 30 days in advance. The mean absolute percentage error (MAPE) is used to test the performance of the prediction model when the test data are used. The MAPE is defined as

$$MAPE = \frac{1}{N}\sum_{n=1}^{N}\frac{\left|y(n)-\hat{y}(n)\right|}{y(n)}\times 100 \tag{13}$$

where N = total no. of test patterns, $y(n)$ = actual value and $\hat{y}(n)$ = predicted value.

The structure chosen for MLP and RBF network is 10:3:1.The structure of FLANN is 10:50:1. In FLANN each input is expanded to five terms using trigonmetric expansion. The value of learning parameter μ is chosen suitably between 0 to 1 for all the networks. The comparison of converegence characteristics of RBFNN along with other two neural network models for 1 and 7 days ahead are obtained for S&P500 and DJIA and shown in Figs.2 and 3 respectively.The comparison of actual value and predicted value during training and testing for DJIA stock index for 7days ahead prediction using RBFNN are depicted in Figs.4 and 5 respectively. Similarly for S&P500 the comparison of actual and predicted value during training and testing for 1 day ahead prediction using RBFNN are presented in Figs.6 and 7 respectively. The MAPE value for two data sets obtaind upon optimized selection of rules and parameters for RBF, MLP and FLANN networks for 1day,7 and 30 days are given in Tables 2.

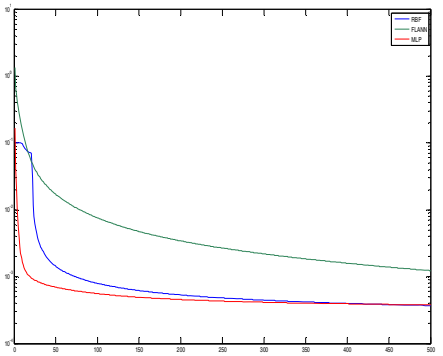


Fig. 2. Comparison of convergence characteristics of S&P500 for 1day ahead prediction using three different ANN models

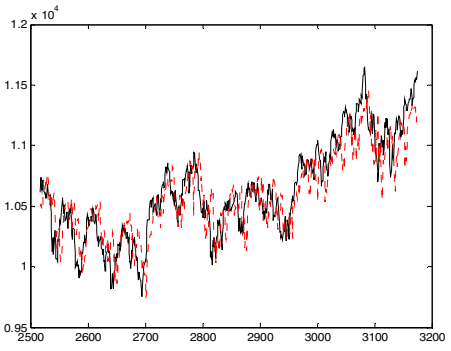


Fig. 5. Comparison of actual and predicted value of DJIA for 7 days ahead prediction using RBF model during testing

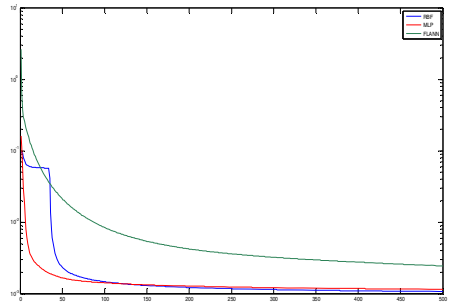


Fig. 3. Comparison of convergence characteristics of DJIA for 7days ahead prediction using three different ANN models

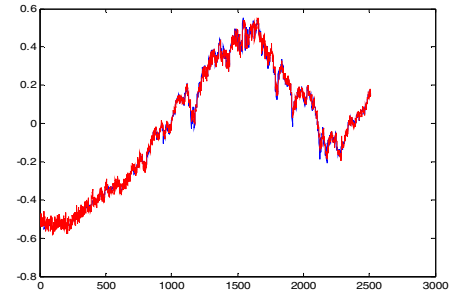


Fig. 6. Comparison of actual and predicted value of S&P500 for 1 day ahead prediction using RBF model during traing

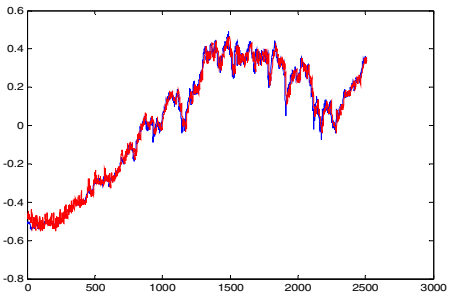


Fig. 4. Comparison of actual and predicted value of DJIA for 7 days ahead prediction using RBF model during traing

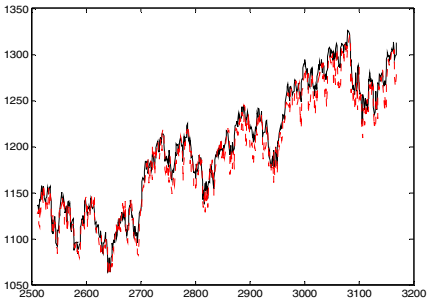


Fig. 7. Comparison of actual and predicted value of DJIA for 1 day ahead prediction using RBF model during testing

Table 2. Comparison of MAPE value using different prediction models

Days ahead	DJIA			S&P500		
	RBF	FLANN	MLP	RBF	FLANN	MLP
1	**0.7891**	1.2609	0.8996	**0.9471**	1.4313	0.9966
7	**1.4892**	2.0872	1.7683	**1.6693**	2.4258	2.5530
30	**2.1005**	3.3136	2.8589	**2.2471**	3.2947	2.5373

5 Conclusion

The proposed forecsating model predict different stock indices using technical indicators derived from the past stock indices. To demonstarte the performance of the proposed models simulation study is carried out using known stock indices and their prediction performance is compared with other two neural network models. The comparison indicates that the proposed RBF based prediction model gives better prediction accuracy in comparison to MLP and FLANN.

References

1. Hassan, M.R., Nath, B., Kirley, M.: A fusion model of HMM, ANN and GA for stock market forecasting. Expert Systems with Applications 33, 171–180 (2007)
2. Liu, F., Wang, J.: Fluctuation prediction of stock market index by Legendre neural network with random time strength function. Neurocomputing 83, 12–21 (2012)
3. Ni, L.-P., Ni, Z.-W., Gao, Y.-Z.: Stock trend prediction based on fractal feature selection and support vector machine. Expert Systems with Applications 38, 5569–5576 (2011)

4. Kim, H.-J., Shin, K.-S.: A hybrid approach based on neural networks and genetic algorithms for detecting temporal patterns in stock markets. Applied Soft Computing 7, 569–576 (2007)
5. Majhi, R., Panda, G., Majhi, B., Sahoo, G.: Efficient prediction of stock market indices using adaptive bacterial foraging optimization (ABFO) and BFO based techniques. Expert Systems with Applications 36, 10097–10104 (2009)
6. Liao, S.-H., Ho, H.-H., Lin, H.-W.: Mining stock category association and cluster on Taiwan stock market. Expert Systems with Applications 35, 19–29 (2008)
7. Guresen, E., Kayakutlu, G., Daim, T.U.: Using artificial neural network models in stock market index prediction. Expert Systems with Applications 38, 10389–10397 (2011)
8. Tsai, C.-F., Hsiao, Y.-C.: Combining multiple feature selection methods for stock prediction: Union, intersection, and multi-intersection approaches. Decision Support Systems 50, 258–269 (2010)
9. Zhang, Y., Wu, L.: Stock market prediction of S&P 500 via combination of improved BCO approach and BP neural network. Expert Systems with Applications 36, 8849–8854 (2009)
10. Enke, D., Grauer, M., Mehdiyev, N.: Stock Market Prediction with Multiple Regression, Fuzzy Type-2 Clustering and Neural Networks. Procedia Computer Science 6, 201–206 (2011)
11. Dai, W., Wu, J.-Y., Lu, C.-J.: Combining nonlinear independent component analysis and neural network for the prediction of Asian stock market indexes. Expert Systems with Applications 39, 4444–4452 (2012)
12. Elanayar, S.V.T., Shin, Y.C.: Radial basis function neural network for approximation and estimation of nonlinear stochastic dynamic systems. IEEE Trans. Neural Network 5, 594–603 (1994)
13. Samantray, S.R., Dash, P.K., Panda, G.: Fault classification and location using HS-transform and radial basis function neural network. Electric Power Syst. Research 76, 897–905 (2006)
14. Oyang, Y.J., Hwang, S.C., Ou, Y.-Y., Chen, C.Y., Chen, Z.W.: Data classification with radial basis function networks based on a novel kernel density estimation algorithm. IEEE Trans. Neural Netw. 16(1), 225–236 (2005)
15. Rout, M., Majhi, B., Mohapatra, U.M.: Efficient Long Range Prediction of Exchange Rates using Radial Basis Function Neural Network Models. In: Proc. of IEEE ICAESM 2012, pp. 530–535 (2012)

Gene Subset Selection for Cancer Classification Using Statsitical and Rough Set Approach

Asit Kumar Das[1] and Soumen Kumar Pati[2]

[1] Department of Computer Science and Technology,
Bengal Engineering and Science University, Shibpur, Howrah - 711103
[2] Department of Computer Science/Information Technology,
St. Thomas' College of Engineering and Technology, 4, D.H. Road, Kolkata - 700023
asitdas72@rediffmail.com, Soumen_pati@rediffmail.com

Abstract. Microarray technique is very useful for measuring expression levels of thousands or more of genes simultaneously. One of challenges in classification of cancer using high-dimensional gene expression data is to select minimal number of relevant genes which can maximize classification accuracy. Because of the distinct characteristics inherent to specific cancerous gene expression profiles, selecting the most informative cancer-related genes from high volume microarray gene expression data is an important and challenging bioinformatics research topic. In the paper, first some important genes are identified based on their rank computed statistically and then rough set theory is applied on reduced gene set for selecting genes with high class-discrimination capability. The method constructs relative discernibility matrix to find out the core genes which are essentially required to distinguish the normal and tumor samples and iteratively adds high ranked noncore genes one at a time to core genes for maximizing classification accuracy. The method is applied on some well known cancerous datasets to show the goodness of the method.

Keywords: Microarray cancer data, gene selection, Rough Set Theory, Discernibility matrix, Cancer classification.

1 Introduction

Now-a-days, an increasing number of applications in different fields produce massive volumes of very high dimensional objects under a variety of experimental constrains. In gene microarray dataset, it is common to encounter large sets of observations (genes), represented by hundreds or more of dimensions (samples). Microarray technology allows to simultaneously analyzing large number of genes and thus can give important insights about cell's meaning. The availability of massive volume of experimental dataset based on cancer research studies has given momentum in recent years to a large effort in developing mathematical, statistical, and computational techniques to guess biological models from experimental dataset. In most of the bioinformatics dataset, the number of genes is significantly larger than the number of samples (high gene-to-sample ratio dataset). This is typical of cancer classification [1]

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 294–302, 2012.

tasks where a logical investigation of the correlation of expression patterns of thousands of genes to specific phenotypic changes is expected to provide an enhanced catalog of cancer. In this context, the number of expressed gene probes (up to several thousands) and the number of observations to the number of tumor samples (typically on the order of hundreds) is typically interrelated.

In DNA microarray data analysis generally biologists measure the expression levels of genes in the tissue samples from patients, and find explanations about how the genes of patients relate to the types of cancers they had. Many genes could strongly be correlated to a particular type of cancer, however, biologists prefer to focal point on a small subset of genes that dominates the outcomes before performing in-depth analysis and expensive experiments with a high dimensional dataset.

The important gene selection [2] is primarily very fundamental in cancer research topic as the number of selected genes irrelevant to classification accuracy may be degrade, and hence, accurate prediction can be achieved only by performing gene selection reasonably, that is, identifying most informative genes from a large number of candidates [2]. Finding good subset of genes by exhaustive search of all possible combination of genes is an NP- complete problem. Therefore, different strategies, like Rough Set Theory [3], Heuristic Search [4] and so on are used to accomplish this task. Heuristic searches find a good solution efficiently by sacrificing completeness of the search. Rough set theory, a mathematical model, is popularly employed by researchers as it does not require any additional information like probability, fuzzy or Dempster-Shafer strategies.

In the paper, a novel gene selection technique has been proposed for selecting relevant genes to predict normal and cancerous samples. The method can be broken down into following steps:

i. The gene dataset is clustered and validated using [5] to achieve an optimal set of clusters and assign same class label (decision value) to all the genes within a cluster in such a way that two genes of different clusters have different class label. Thus decision attribute with k distinct values is generated for the dataset.

ii. The dataset is discretized using ChiMerge [6] algorithm and score of each sample with respect to generated decision attribute is computed using division operation of relational algebra [7]. Lower the score implies more important sample of the gene data and vice versa. The minimal subset of samples is formed using [8] by considering the samples with score below the average score.

iii. Importance factor of each gene with respect to discrete values of different predefined class label of microarray training-dataset is computed and form initial gene subset with high importance factor. Then traditional k-means clustering algorithm [9] with $k = 2$ is applied on each selected gene and miss-classification accuracy is measured to select optimal number of genes with less miss-classification accuracy and form modified gene subset.

iv. Core, the most important genes are found out from modified gene subset by constructing relative indiscernibility matrix [3]. Noncore genes from modified gene subset are added to core genes based on their importance factor one by one to increase the classification accuracy of a predefined set of classifiers and finally, the genes providing the highest classification accuracy is considered as the selected gene subset for cancer classification.

The proposed method yields better classification accuracy compare to others for some experimental datasets, which shows its effectiveness.

The article is organized into four sections. Section 2 describes the proposed gene subset selection for cancer classification. The experimental results and performance evaluation of the method are described in Section 3 and conclusions are drawn in Section 4.

2 Gene Subset Selection

The major problem in interpreting microarray data comes from their innate nature of 'high dimensional large sample size as well as large number of genes'. Therefore, robust and accurate gene selection methods are required to identify differentially expressed group of genes across different samples, e.g. between cancerous and normal cells. Gene selection is necessary to find out genes, responsible for complex disease which take part in disease network and provide information about disease related genes. Successful gene selection will help to classify different cancer types, lead to a better understanding of genetic signatures in cancers and improve treatment strategies.

2.1 Dimension Reduction

As all samples are not relevant to characterize the genes, a relevance analysis of samples is necessary to select only the important ones. Here, score is computed for each sample using division operation of relational algebra [7] for which decision attribute is required. As the gene datasets are unlabeled, they need to be transformed to labeled data. So, the dataset is clustered and validated using [5] to obtain optimal clusters of genes and labeled each gene in such a way that genes in same cluster have same class value and genes in different clusters have different class value. Thus, a decision attribute is generated which is used to compute score of each samples. Score computation by division (÷) operation of relational algebra [7] needs the discrete values of the samples. So, before score computation, the datasets are also discretized by ChiMerge [6] algorithm.

Let the labeled gene expression discretized dataset $\boldsymbol{DS = (U, C, D)}$, where $\boldsymbol{U}$ is the universe of discourse (set of all genes), $\boldsymbol{C = \{C_1, C_2, \ldots, C_n\}}$ is the number of samples and $\boldsymbol{D}$ is the decision attribute with k distinct values generated using clustering algorithm [5]. Now, the relational algebra operation division (÷), defined in 'Definition 1', is used to compute the score of each sample C_i using score function $\boldsymbol{S(C_i)}$ defined in equation (1).

$$S(C_i) = \left| \Pi_{C_i \cup D}(DS) \div \Pi_D(DS) \right| \tag{1}$$

Where $i = 1, 2, \ldots, n$. Minimum score of C_i implies that there is maximum number of genes having sample values similar to C_i, which can uniquely take the decisions. Thus, a sample with the minimum score is of the maximum importance and so lower score implies higher possibility of becoming a member of reduced sample set, selected using [8].

Definition 1: Relational algebra operation division (÷) is a binary operation applied on two relations R_1 (P) and R_2 (Q) and produce another relation R ($P - Q$) where $Q \subset P$ where P, Q are set of samples of R_1, R_2 respectively. So, R (i.e., $R_1 \div R_2$) contains set of all genes g such that for any gene g_1 and g_2 of R_1 and R_2 respectively, following conditions are hold.

- $g\ [P - Q] = g_1\ [P - Q]$
- $g_1\ [P - Q] = g_2\ [Q]$

2.2 Initial Gene Subset Selection

As the samples of genes are collected from both normal and cancerous patients, so the samples are divided into two disjoint classes say, normal class d_1 and cancerous class d_2. Now for each gene g_i, the maximum frequencies of discrete sample values are computed in each class; let they are P_{li} and P_{ri} for class d_1 and d_2 respectively. If the maximum frequencies of P_{li} and P_{ri} occur for same discrete value, then obviously, the gene g_i is not important for cancer classification, as both the normal and cancerous samples are almost similar. Otherwise, the sample values of normal and cancerous samples are distinct for gene g_i and so the gene is considered as an important gene with importance factor (PF_i) computed using (2).

$$PF_i = \frac{P_{l_i} + P_{r_i}}{m} \tag{2}$$

Where, $i = 1, 2,\ldots,n$ and m is the total number of samples.

So, higher the importance factor more relevant the gene is and vice versa. Thus, top n_1 ($<< n_2$) genes with importance factor greater than the average are selected as initial important genes and kept in set *IRED*.

2.3 Modified Gene Subset Selection

Distance based similarity/dissimilarity measurement among genes may not be effective for gene data analysis in a high dimensional space. Also, elegant gene selection decreases the workload and simplifies the subsequent design process to a great extent. Here, misclassification index for each of n_1 genes in *IRED* are computed and the genes with index value less than the average are considered as modified gene subset *MRED*. Let *MRED* contains n_2 ($<< n_1$) genes. Modified gene subset is computed as follows:

- The samples of each of n_1 genes in *IRED* is partitioned into two disjoint clusters using K-means clustering algorithm [9] with k = 2, as the genes contain normal and cancerous samples.
- The samples of each gene are already partitioned into two classes, normal class d_1 and cancerous class d_2.
- If for gene g_i, m_{1i} is the number of samples of d_1 class clustered as samples of d_2 class, m_{2i} is the number of samples of d_2 class clustered as samples of d_1 class and

m is the total number of samples then the misclassification index for gene g_i is computed using (3), for $i = 1, 2, \ldots, n_1$ genes.

$$ME_i = \frac{m_{1i} + m_{2i}}{m} \tag{3}$$

2.4 Final Gene Subset Selection

The method uses the concept of rough set theory (RST) to compute the minimal subset of genes called reduct which can, by itself, fully characterize the knowledge in the gene dataset as the whole set of genes (U) does and preserves partition of data with respect to cancer classification. But the reduct computation is an NP-complete problem and it is not unique. Whatever may be the number of reducts, all contain some genes which are absolutely necessary for classification, such genes are known as core genes. Selection of them is independent of any evaluation measure and classification algorithm, they are in nature the core of all good subset of genes. After core selection, a heuristic approach based on classification accuracy is applied for a single reduct computation, which gives final gene subset used for cancer classification. Let the reduced gene dataset is represented as RDS = (S, $MRED$, D), where $S = \{s_1, s_2, \ldots, s_m\}$ is the set of selected samples, $MRED = \{g_1, g_2, \ldots, g_{n2}\}$ is the modified set of genes and D is the decision attribute with two distinct values, d_1 and d_2 for normal and cancerous samples respectively.

2.4.1 Core Gene Selection

The sample set S is partitioned into two subsets, normal sample set S_1 and cancerous sample set S_2 so that S_1 contains p number of samples and S_2 contains q number of samples. Core genes are selected computing the discernibility matrix [4] for the dataset RDS. Each cell of the matrix contains genes that distinguish two associated samples. The discernibility matrix $M = (m_{ij})$ is a $p \times q$ matrix, in which the element m_{ij} for a normal sample and cancerous sample pair (s_i, s_j) is defined using (4).

$$m_{ij} = \{\cup\, g_k \in MRED \mid s_i(g_k) \neq s_j(g_k)\} \tag{4}$$

Where, $i = 1, 2, \ldots, p$; $j = 1, 2, \ldots, q$ and $s_i(g_k)$ is the value of sample s_i of gene g_k in $MRED$.

So, each entry in matrix M contains some genes which can differentiate the associated normal and cancerous samples. Here, the frequency of the genes in the matrix is calculated and the highest frequency gene is considered as the core gene as it can distinguish the maximum number of normal and cancerous samples. If many genes have same frequency, then all such genes are considered as core or the most important gene set CR.

2.4.2 Minimal Gene Subset Selection

Core genes have already been identified using discernibility matrix, a concept of Rough Set Theory. So, the modified gene set $MRED$ is partitioned into two disjoint

subsets *CR*, containing core genes and the remaining gene set *RS*, containing non-core genes. To obtain the minimal gene subset which can fully characterized the gene dataset, following steps are performed:

- Firstly, gene set *CR* is applied on some well known classifiers and the accuracies are recorded individually.
- Next, a gene with highest important factor (computed using equation (2)) is taken from *RS* and add to *CR*, provided the average classification accuracy of the classifiers increases.
- Iteratively a gene from *RS* is selected and *CR* is modified, which finally gives us the minimal gene subset *RED*.

In general, the time complexity of iterative approach is not good enough for searching the appropriate genes from *RS*. But in the method, it contains very few genes as the original gene set is already reduced by the previous methods; as a result, it reduces the overhead time of utilization of the optimization techniques such as PSO, GA, Differential evaluator etc. The overall gene selection algorithm is given bellow.

```
Algorithm: Gene_Subset_Selection(MRED, RED)
Input: Reduced gene set MRED = {g1, g2,…,gn2} with sample
       set S = {s1, s2, …, sm}.
Output: Minimal gene subset RED for gene classification.
Begin
    Let, S is partitioned into normal sample set S1 and
         cancerous sample set S2
    Let, S1 contains p number of samples and S2 contains q
         number of samples.
    Compute discernibility matrix (M)pXq using (4)
    Compute core CR of genes with maximum frequency in M
    Compute noncore gene subset RS = MRED – CR
    Compute importance factor of all genes in RS by (2)
    For next highest important factor gene gi • RS {
       For predefined j classifiers
          Compute accuracy ACj on gene dataset CR
       Compute average accuracy avg_AC
       New_CR = CR ∪ {gi} /* gi is of max imp. Factor */
       For same j classifiers
          Compute accuracy New_ACj on gene dataset New_CR
       Compute average accuracy New_avg_AC
       If(New_avg_AC > avg_AC)
          CR = New_CR
    } /* end of loop i */
    RED = CR
End
```

3 Experimental Results and Performance Evaluation

Experimental studies presented here provide an evidence of effectiveness of proposed gene subset selection technique. Experiments were carried out on gene dataset publicly available at http://www-genome.wi.mit.edu/mpr and http://carrier.gnf.org/welsh/prostate as training and test dataset, summarized in Table 1. The training dataset is used for forming optimal gene subset and test dataset is used for measuring the effectiveness of the method by computing classification accuracy and kappa statistics.

Table 1. Summary of Gene expression (training/testing) dataset

Dataset	No. of Genes	Class Name	No. of Training Samples (class1/class2)	No. of Test Samples (class1/class2)
Prostate cancer	12600	Tumor/Normal	102(52/50)	34(25/9)
Lung cancer	12533	MPM/ADCA	32(16/16)	149(15/134)

The proposed statistical approaches such as *dimension reduction (DIM)*, *initial gene subset selection (IGS)* and *modified gene subset selection (MGS)* methods are used sequentially that select the important samples and genes listed in Table 2.

Table 2. Reduced gene Dataset by reduction of samples and genes

Dataset	No. of samples selected by DIM	No. of genes selected by IGS	No. of genes selected by MGS
Prostate cancer	29 out of 102	1225 out of 12600	443 out of 1225
Lung cancer	12 out of 32	3790 out of 12533	1134 out of 3790

Then, the rough set approach is applied on reduced datasets that select core genes as well as the minimal gene subset (called reduct) based on the predefined set of classifiers {"Naïve Bayes", "Multilayer Perceptron", "Bagging", "J48"}. For Prostate cancer dataset, core gene is {*37639_at*} and reduct is {37639_at, 39939_at}, for which classification accuracy and Kappa Statistic is measured based on test samples, as listed in Table 3, to show the efficiency of the method.

Table 3. Minimal gene subset selection for Prostate cancer dataset

Gene Name	Naïve Bayes		Multilayer Perceptron		Bagging		J48	
	Accuracy (%)	Kappa Statistic	Accuracy (%)	Kappa Statistic	Accuracy (%)	Kappa Statistic	Accuracy (%)	Kappa Statistic
{37639_at}	97.06	0.9280	97.05	0.9270	97.05	0.9270	97.05	0.9217
{37639_at, 39939_at}	100	1	100	1	100	1	97.06	0.9280

Similarly, for Lung cancer dataset, core is {*37205_at*, *32046_at*} and then one noncore gene is added iteratively and gives the reduct {37205_at, 32046_at, 39795_at, 575_s_at}, as shown in Table 4.

Table 4. Minimal gene subset selection for Lung cancer dataset

Gene Name	Naïve Bayes		Multilayer Perceptron		Bagging		J48	
	Accuracy (%)	Kappa Statistic	Accuracy (%)	Kappa Statistic	Accuracy (%)	Kappa Statistic	Accuracy (%)	Kappa Statistic
{37205_at, 32046_at}	95.30	0.7480	95.30	0.7480	94.63	0.6848	93.96	0.6760
{37205_at, 32046_at, 39795_at}	97.32	0.860	97.99	0.8854	95.30	0.7326	94.63	0.7201
{37205_at, 32046_at, 39795_at, 575_s_at}	97.32	0.860	99.33	0.9618	98.66	0.9259	97.99	0.8854

The proposed method is implemented in Mat lab 7.8.1 version and all classification performances are measured by Weaka-3-6-5 Data Mining tool.

4 Discussions and Conclusion

Systematic and unbiased approach to cancer classification is of great importance to cancer treatment and drug discovery. Biologists focus on a small subset of genes that dominates the outcomes before conducting in depth analysis and expensive experiments with a larger set of genes. Therefore, automated discovery of this small and good gene subset is highly desirable. In the paper, a novel statistical and rough set approach has been proposed to select only important informative genes, which classify the cancer dataset effectively and efficiently.

References

1. Wang, X., Gotoh, O.: Microarray-Based Cancer Prediction Using Soft Computing Approach. Cancer Informatics 7, 123–139 (2009)
2. Su, Y., Murali, T.M., Pavlovic, V., Schaffer, M., Kasif, S.: RankGene: identification of diagnostic genes based on expression data. Bioinformatics 19, 1578–1579 (2003)
3. Pal, S.K., Skowron, A. (eds.): Rough Fuzzy Hybridization: A new trend in decision-making, pp. 3–98. Springer, Berlin (1999)
4. Zhong, N., Dong, J., Ohsuga, S.: Using rough sets with heuristics for feature selection. J. Intell. Inf. Syst., 199–214 (2001)

5. Pati, S.K., Das, A.K.: Cluster Analysis of Microarray Data Based on Similarity Measurement. International Journal of Bioinformatics Research 3(2), 207–213 (2011) ISSN: 0975-3087
6. Kerber, R.: ChiMerge: Discretization of Numeric Attributes. In: Proceedings of AAAI 1992, Ninth Int'l Conf. Artificial Intelligence, pp. 123–128. AAAI-Press (1992)
7. Das, A.K., Sil, J.: An Efficient Classifier Design Integrating Rough Set and Set Oriented Database Operations. Applied Soft Computing, Elsevier Science Direct 11, 2279–2285 (2011)
8. Pati, S.K., Das, A.K.: Optimal Samples Selection from Gene Expression Microarray Data Using Relational Algebra and Clustering Technique. In: Satapathy, S.C., Avadhani, P.S., Abraham, A. (eds.) Proceedings of the InConINDIA 2012. AISC, vol. 132, pp. 507–514. Springer, Heidelberg (2012)
9. Bradley, P.S., Bennett, K.P., Demiriz, A.: Constrained k-means clustering (Technical Report MSR-TR-2000-65), Microsoft Research, Redmond, WA (2000)

Support Vector Machine for Large Databases as Classifier

Rahul Kumar Sevakula and Nishchal K. Verma

Department of Electrical Engineering,
Indian Institute of Technology Kanpur,
Kanpur, India
{srahulk,nishchal}@iitk.ac.in

Abstract. Support Vector Machine (SVM) has been successful in multiple areas and is widely accepted as the best off the shelf algorithm for classification. A standard SVM has $O(n^3)$ time and $O(n^3)$ space complexities, hence making it limited in its usability for large database. We know that in real world scenario, most of the databases where Data Mining is used are large. This paper reviews various algorithms and techniques that have been brought forth since 1995 by researchers for implementing SVMs in a practical manner for large databases.

Keywords: SVM, Large database, Classification, Regression, Review.

1 Introduction

Support Vector Machines has a strong theoretical background in stochastic process, geometry and functional analysis. It is because of its good mixture of these three that it has been so successful at a practical level. The approach of SVM is quite simple. In a linearly separable case, the algorithm tries to find out the hyper plane which not only succeeds in classifying all the training points correctly, but also gives maximum margin between the boundary vectors (also known as support vectors). In Generalized Learning Theory by Vapnik, he proves that, given that the training samples are sufficiently large (beyond a certain threshold), then for most of the cases, provided VC dimension is not very high, ERM i.e. Empirical Risk Minimization (minimizing error for training samples) leads to SRM i.e. Structural Risk Minimization (minimization of error for test samples). Later it was further found that larger margin assists in getting lower VC dimension. This basically means that the classifier becomes generalized and can classify unknown samples correctly with good accuracy.

Soft Margin Classifier i.e. classifier for cases where classes are not completely separable, can also be worked upon effectively using SVMs by adding penalty with a cost for misclassified points in the original objective function.

The optimization problem for SVM is converted to that of a Convex Optimization problem and is solved using Lagrange Multipliers. Taking advantage of the dual form of the problem, our objective function becomes simpler to solve and also gives some

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 303–313, 2012.

special properties. One such useful property is that in the dual form, both objective function as well as solution for the hyper plane, training data is never accessed alone as individual points, but only in the form of dot products. Thus Kernel tricks can be employed which allows SVM to classify in a transformed feature space which generally will be a high dimension space, without actually mapping the data to the high dimension space. Thus SVM combined with the Kernel trick become extremely powerful being able to find best non-linear hyper plane in the original input space.

By changing the original objective function slightly, the problem can be converted to that of a regression problem. SVM has also been successful in regression.

Some of the applications where SVMs are already being used are in Text classification, Speaker Recognition, Speech Recognition, Image Recognition, Health Monitoring Devices etc.

SVMs from theoretical point of view are excellent, but when it comes to practical application, they are limited mainly because of having large time and space complexities of $O(n^3)$ and $O(n^3)$ respectively. The purpose is to check the state of art techniques that have been brought forth till now for making SVM efficient while being applied for large databases. For understanding this, obviously our first step would be understand the theory and implementation of SVMs thoroughly so that we can understand what part of the SVM implementation is being optimized from the perspective of large databases. Section 2 will briefly describe the mathematics involved in SVM ending with various objective conditions and their constraints. Section 3 will discuss the practical implementation on how to find the solution for the various objective functions and also discuss ideas on what can be optimized. Section 4 will discuss some novel algorithms that were introduced in the last one and half decade.

2 Support Vector Machines

Since most of the real world problems need a soft margin classifier rather than hard margin classifier, we will directly begin with soft margin problem.

$$\begin{aligned} &\min_{\xi,w,b} < w.w > + C\sum_{i=1}^{l} \xi_i^{\,2} \\ &\textit{subject to} \quad y_i(< w.x_i > + b) \geq \xi_i, \quad i = 1,.....,l \\ &\xi_i \geq 0, \iota = 1,....,\lambda \end{aligned} \tag{1}$$

As this turns out to be a convex optimization problem, after applying KKT conditions, we can get the dual objective function which is relatively easier to solve and also gives some wonderful properties like the kernel trick, as discussed before. After solving we get the following dual problem.

$$\max W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum y_i y_j \alpha_i \alpha_j K(x_i, x_j) + \frac{1}{C} \delta_{ij}$$

$$\text{subject to} \quad \sum_{i=1}^{l} y_i \alpha_i = 0 \qquad (2)$$

$$\alpha_i \geq 0, \; i = 1, ..., l$$

where, α is the vector containing all Lagrange multipliers.

3 Naive Way of Implementation

Given the objective function with constraints, how we find the solution for the Lagrange multipliers, is given in this section.

3.1 Gradient Ascent Method

We take a random vector of α and then we update it in the following way.

$$\delta\alpha = \eta \frac{\partial W(\alpha)}{\partial \alpha}$$

where, η is called the learning rate or step size which can be either fixed as well as variable. Note that $\frac{\partial W(\alpha)}{\partial \alpha}$ gives the direction of steepest ascent at that point and $\delta\alpha$ needs to be subtracted from α. Note that if we take the entire vector of α, this partial derivative is very expensive to calculate, as its dimension is very huge.

3.2 Stopping Criteria

Generally three basic criteria are used to know when we have to stop solving the dual objective function. They are

- Monitoring the growth of the dual objective function. Training can be stopped when the fractional rate of increase of the objective function $W(\alpha)$ falls below a given tolerance.
- Monitoring the KKT conditions for the primal problem as they are necessary and sufficient conditions for convergence. Given below are the conditions.

$$\alpha \geq 0 \;,$$

$$y_i f(x_i) \begin{cases} \geq 1 & for\ point\ with\ \alpha_i = 0 \\ = 1 - \alpha_i / C & for\ point\ with\ \alpha_i > 0 \end{cases} \qquad (3)$$

- To check the gap between the primal and dual objective functions (known as feasibility gap). This is so because this gap vanishes only at optimal points. So if we observe that the difference between the primal and dual objective functions is below a tolerance level, then we can stop solving the quadratic equation further. Given below is a proven result which shows our point

$$L(w^*, \alpha, \beta) \leq L(w^*, \alpha^*, \beta) \leq L(w, \alpha^*, \beta^*) \tag{4}$$

where, w^* is optimal solution in primal problem and α^* is optimal solution in the dual problem. All three are equal only at the optimal solution.

3.3 Ideas for Making SVM Faster

Once we know the working of SVM on how it tries to find out the support vectors from the training set (which are generally much lesser in number as compared to the entire training set), we can get some immediate ideas on how it can be made faster for larger databases. They are:

- Since very small number of samples in the training dataset actually contribute to the solution of optimal hyper plane, it would be very good if we can remove most of the unnecessary points without compromising on the solution, or else divide the problem into smaller SVMs which are faster.
- The objective function whose maximum is to be found is very tedious if we go by traditional convex optimization methods, hence we can get SVM implemented faster if we have techniques to converge at the solution faster, especially from the perspective of large databases.
- Kernel trick is greatly responsible for making SVMs popular. Repeated kernel computation is also expensive. So it would be good idea to store the kernel matrix fully or partially as it save time of redundant computation. This comes even handier by knowing the fact that some kernels like RBF kernel are not so easy to compute. Storing of the kernel matrix is known as Kernel caching.

4 Various Algorithms

The above three principles generally form the basis of almost all algorithms that have come till now for making the SVM training faster. Now we look into the various algorithms that are either very famous or have brought about some novel idea in the implementation method. The algorithms have been broadly classified into two categories, i.e. algorithms which mainly aim on reducing the size of training dataset and those which aim on faster convergence. Kernel caching is generally a part of most of the algorithms. Although there are some papers who focus on how to use the limited memory for storing Kernel matrix efficiently, we will not to get into details of such methods.

4.1 Reducing the Training Dataset

One of the good properties in solving SVM is that for finding the margin, very few points of the entire data are used to form the final hyper plane as most of the multipliers, are equal to zero.

Clustering Approach. R Koggalage et. al [2] worked upon clustering the entire training data. Then out of the clusters formed, those which are crisp are identified. Crisp clusters are those which have training samples from only one of the two classes. From crisp clusters, points within a certain radius from cluster center are removed and are removed from the set of training samples as they most probably cannot be support vectors. Considering that data is not so noisy, good number of the formed clusters are crisp in nature. Thus training data is reduced greatly.

Edge Detection with Clustering Approach. In images, edge is detected when the neighborhood pixels are very much different from the current pixel value. Since support vectors are points close to the margin, we use this idea of finding the points that are possibly support vectors. If the point has any of its neighbors (found by k nearest neighbors) from the other class as compared to itself, it can be a support vector. Hence all other points apart from these points which are on the edge are not much required. Still to make sure over fitting does not happen, the training data which are not on the edge, are collected and clustered into k clusters, and only the cluster centers are considered while solving the objective function. Thus the training data used is drastically reduced and so is the time for getting the solution which doesn't compromise much on accuracy of results [3].

Fast SVM Training Algorithm with Decomposition on Very Large Databases. Jian-xiong Dong et.al [4] presented amazing work in their paper on very large databases with large number of classes. One example of such large database would be recognition of written Chinese characters with more than 6000 classes, which obviously will need huge training dataset. Since implementing SVM directly on the large dataset would take too much time, they divided the entire databases into smaller randomly made subsets. An SVM model is implemented for each of the data subset, from which we get the support vectors. All the support vectors of all models are used for working out the final SVM model for the entire dataset. Hence training dataset of each SVM becomes very small, even for that of the final model. Since the support vectors of the subsets are only the probable support vectors of the final model this algorithm is valid and does not compromise on accuracy. Note that since several SVM models will be built initially which are independent to each other model, the computation can be done in parallel thus making it even faster for multi core/cpu systems. Apart from this they use Kernel caching and fast decomposition algorithms like that of Sequential Minimal Optimization (explained later), etc.

A Parallel Mixture of SVMs for Very Large Scale Problems (*SVMTorch*). Ronan et. al.[a] based on this paper[5] and [6] have developed the package *SVMTorch*. This package which is made in C++ is quite famous and has been used a lot for various experiments. They use the concept of parallel mixture of several small SVM instead of one SVM for classification. This is because training several smaller SVM is much

less expensive as compared to one SVM for a large database. So here also the training database is randomly divided into M subsets where M is generally between 10 and 50. The decisions of several SVM are then weighted using a ANN model such that it decreases the following objective function:

$$C = \sum_{i=1}^{N} \left[h\left(\sum_{m=1}^{M} w_m(x) s_m(x_i) \right) - y_i \right] \tag{5}$$

In the experiments that they performed, they showed that this method not only improved the speed remarkably, but also improved the accuracy. In [6], their discussion is on making Support Vector Regression faster through faster decomposition algorithms (like SMO).

4.2 Faster Convergence

As we had discussed before that since the dimension of α is huge, solving the dual objective function by taking partial derivative and thus updating all α_i is very tedious. The basic principle in almost all algorithms for faster decomposition is to efficiently select a working set. Working set is the part of α which will be updated whereas all others apart from the working set will be frozen, that is they will not be updated. Note that for all changes in α, the conditions of dual problem need to be satisfied as given in (2), which are.

$$\sum_{i=1}^{l} \alpha_i y_i = 0 \quad \text{And} \quad \alpha_i \geq 0 \tag{6}$$

SMO (Sequential Minimal Optimization). [7] The least number of α_i to be updated such that the above conditions be satisfied will have to be 2. Such extreme approach is taken by Platt's SMO where α_i of only two points are updated from the large number of training samples. Though the number of iterations is larger, each iteration is less expensive, thus giving an overall boost in speed. Choice of the two points is determined by a heuristic, while optimization of the two multipliers is performed analytically. Two such heuristics are:

- Take those points which violate KKT conditions. Hence their stopping criteria would be when all points fulfill the KKT conditions.
- Take those points which causes largest change to the dual objective function or some other defined function.

SVM Light. This also is a well-known package that is widely used. It is based on paper by Joachim [8] in 1998. He performs three things to make it faster:

- Decomposition strategy which finds a more efficient method to find the working set. By experiments he found that the best size for working set is 2.
- Shrink the problem during the optimization process. For this he divides the Lagrange multipliers into three categories λ^{eq} (set of multipliers which are equal to

zero), λ^{lo} (set of multipliers which are near lower bound) and λ^{up} (set of multipliers which are near upper bound). There is a heuristic given for both λ^{lo} and λ^{up}. During the optimization process, if the value of heuristic for any multiplier belonging to λ^{lo} and λ^{up} is greater than or equal to 0 for last h iterations, it is expected that these multipliers have already reached their optimal value and hence need not be considered in the further optimization process and hence they are dropped and frozen. Thus optimization process becomes faster.

- Kernel caching.

LibSVM. Keerthi et.al.[9] in his famous paper showed the limitations of Platt's SMO and modified it by giving more theory on selecting the two points, by considering bounds and also recommended the updation of β. In 2005, Fan et.al.[10] gave further modification to the decompostion algorithm. Both Keerthi and Joachim based their selection of working set on some objective function which only took first order information into consideration. Some previous experiments showed that second order information gives better results w.r.t. selection of working set. The issue with this approach is twofold - computation for 2nd order information is more expensive and the number of pairs that need to be checked is $O({}^{l}C_2)$ as compared to O(l) for objective functions based on 1st order information (actually it is $O({}^{l}C_2)$, but it can be reduced to O(l) by certain methods). Fan et. al. proposed the technique of selecting O(l) pairs using objective functions based on 1st order information and then selecting the final pair from these O(l) pairs using 2nd order information. LibSVM implements SVM classification using this algorithm. LibSVM is considered to be one of the most popular packages used for solving SVMs and has won various prizes and awards.

5 Implementation

We made a comparison of training time and accuracy with three standard packages – LibSVM [12], SVM Light [8] and SVM Torch [5-6] (explained in Section 4.1).
Two datasets were used:

A9a – Adult [13]

- # of classes: 2
- # of data: 32,561 / 16,281 (testing)
- # of features: 123 / 123 (testing)

Mnist – database of handwritten digits [13]

- # of classes: 10
- # of data: 60,000 / 10,000 (testing)
- # of features: 780 / 778 (testing)

Using stratified sampling, we made random subsets from both the datasets of various sizes for training. All subsets were trained with C-SVM using LibSVM, SVMLight and SVMTorch. The training time, time for classification and accuracy on standard test sets were noted down.

For A9A dataset, the parameters were : C =1, gamma = 1 (default values) and for Mnist dataset parameters were C = 2.828427 and gamma = 0.00728932. These values are based on reports of certain experiments which mentions that these parameters give good accuracy [14].

6 Results

The results obtained by implementing the packages – *LibSVM*, *SVM Light* and *SVMTorch* are shown in Tables 1-3 respectively. Graphs for comparing the three packages w.r.t. training time are shown in Figures 1 and 2. In the case of Mnist data-set, we stopped at a size of 5000 for SVM Light since it was taking more than 8 hrs for training 10,000 samples.

Table 1. Results for LibSVM

A9A dataset			Mnist Dataset		
Size	Training time (s)	Accuracy	Size	Training time(s)	Accuracy
10,000	9s	84.75%	1000	30	94.21%
20,000	34.5s	84.77%	10,000	240	96.88%
32,561	93 s	84.82%	20,000	960	97.14%
			30,000	3000	97.64%
			40,000	10800	98.02%
			50,000	8hrs	98.48%

Table 2. Results for SVM Light

A9A dataset			Mnist Dataset		
Size	Training time(s)	Accuracy	Size	Training time (s)	Accuracy
10,000	39.88	77.96%	1,000	77.16	73.85%
20,000	199	79.09%	2,000	547.82	69.52%
32,561	534.45	79.87%	3,000	1279.43	69.52%
			5,000	1817	66.35%

Table 3. Results for SVMTorch

A9A dataset			Mnist Dataset		
Size	Training time(s)	Accuracy	Size	Training time (s)	Accuracy
10,000	34.23	84.59%	1000	2.4	91.62%
20,000	204.46	84.87%	10,000	94.7	96.9%
32,561	629.51	84.94%	20,000	357.3	97.93%
			30,000	767.8	98.43%
			40,000	1244.4	98.62%
			50,000	1772.9	98.88%

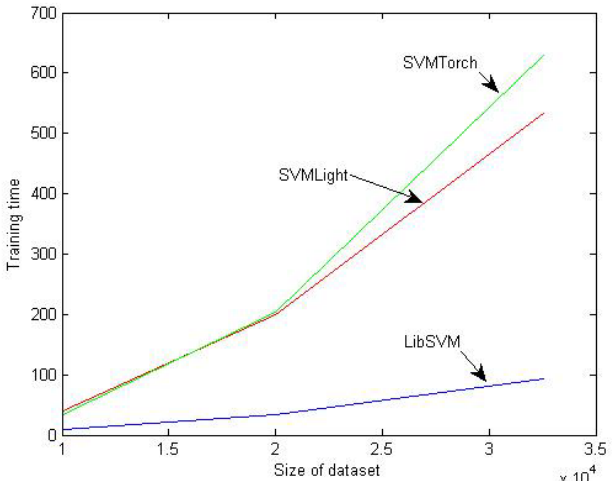


Fig. 1. Graph for comparison of training time with different packages – A9A dataset

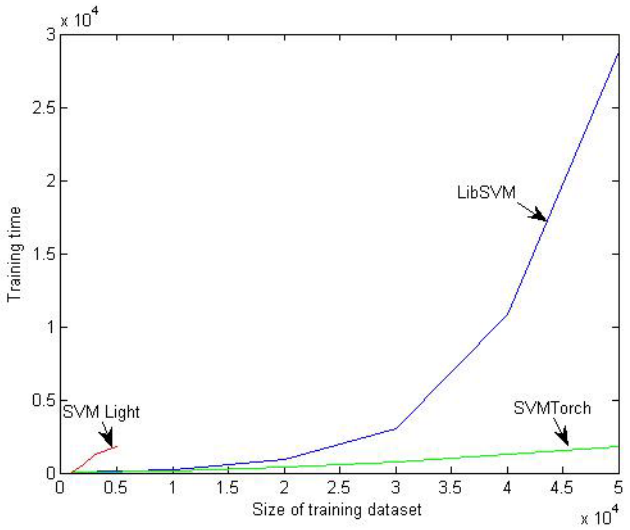


Fig. 2. Graph for comparison of training time with different packages - Mnist dataset

7 Conclusions and Discussions

Table 4 gives a brief overview of results. We can conclude that for problems with 2 classes, we can go for LibSVM and for multi class problem, SVM Torch is best. It is important to note that the format in which SVMTorch accepts data is quite different from the standard sparse dataset format. Hence little extra effort and care would be required while conversion. One more interesting thing to note on SVMTorch is

that the main developer, Ronan Collobert in his website http://ronan.collobert.com/ states that he has not updated the package since long time and so he recommends the users to use LibSVM instead.

Table 4. Overview of Results

Package	Training Time	Accuracy
LibSVM	For binary class, it is fastest and for multiclass it is much slower than SVM Torch	Good accuracy.
SVM Light	Slightly better than SVMTorch for binary class, but very slow for multiclass.	Not good as compared to both LibSVM and SVM Torch.
SVM Torch	Moderate speed for binary class, but quickest for multi class case.	Good accuracy.

8 Future Work

Currently we have tested the packages on only two datasets. Hence to get exact and accurate comparison more testing would be required. Therefore future work would be to have more detailed experiments.

Acknowledgements. I would like to acknowledge Dr. Arnab Bhattacharya for his help in understanding of SVM.

References

1. Crsitianini, N., Shawe Taylor, J.: An introduction to Support Vector Machines (2000)
2. Koggalage, R., Halgamuge, S.: Reducing the Number of Training Samples for Fast Support Vector Machine Classification. Neural Information Processing 2(3), 57–65 (2004)
3. Li, B., Wang, Q., Hu, J.: A Fast SVM Training Method for Very Large Datasets. In: Proceedings of International Joint Conference on Neural Networks, pp. 1784–1789
4. Xiong, J., Krzyzak, A.: Fast SVM Training Algorithm with Decomposition on Very Large Databasets. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(3), 603–618 (2005)
5. Collobert, R., Bengio, S., Bengio, Y.: A Parallel Mixture of SVMs for Very Large Scale Problems. Neural Computation 14(5) (2002)
6. Collobert, R., Bengio, S.: SVMTorch: Support Vector Machines for Large-Scale Regression Problems. Journal of Machine Learning Research 1, 143–160 (2001)
7. Platt, J.: Sequential minimal optimization: A fast algorithm for training support vector machines. Tech. Rep. MSR-TR 98-14, Microsoft Res., Redmond, WA (April 1998)

8. Joachims, T.: Making large-scale support vector machine learning practical. In: Schkolkopf, B., Burges, C., Smola, A. (eds.) Advances in Kernel Methods: Support Vector Machines. MIT Press, Cambridge (1998)
9. Keerthi, S.S., Shevade, S.K., Bhattacharyya, C., Murthy, K.R.K.: Improvements to Platt's SMO algorithm for SVM classifier design. Technical Report CD-99-14, Control Division, Dept. of Mechanical and Production Engineering, National University of Singapore (1999)
10. Fan, R.E., Chen, P.H., Lin, C.J.: Working set selection using second order information for training SVM. Journal of Machine Learning Research 6, 1889–1918 (2005)
11. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, Berlin (1995)
12. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27 (2011), Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvmSVMLight
13. Frank, A., Asuncion, A.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA (2010), http://archive.ics.uci.edu/ml
14. http://www.kernel-machines.org/

Solution to Economic Load Dispatch Problem Based on BFO,CBFO-S and CBFO-H Algorithms and Its Advantages over the PSO

Ananthanaryanan Rathinam and Ripunjoy Phukan

Dept. of Electrical and Electronics Engineering
SRM University, Chennai (603203), India
a_rathinam@yahoo.com, ripun000@gmail.com

Abstract. The Economic Load Dispatch problem (ELD) decides the minimum economic cost of production in a given power system, while keeping the environmental constraints and the load demand to an acceptable level without compromising on the generator ratings. Analysis on this application have been made using multi-modal Bacteria Foraging Optimization(BFO) algorithm, where the bacterial motion is incorporated as a search algorithm in finding the optimum values for economic generation. An improvisation on this algorithm is the cooperative bacteria foraging optimization using serial decomposition (CBFO-s) and CBFO-hybrid that combines BFO and CBFO-s. The search space decomposition and stochastic analysis ensures greater precision and accuracy in cost functions compared to conventional methods of Lagrange's multipliers and Particle Swarm Optimization (PSO). This paper illustrates these points with the ELD problem as a reference.

Keywords: Valve Point effect, emission constraints, chemotactic steps, elimination & dispersal, foraging.

1 Introduction

In Electrical power systems, the problem of unit commitment and economic generation are together considered as arduous and time consuming. The added effects of both are studied in a superficial approach termed as the Economic Load Dispatch Problem (ELD) [1-2].The overall scope of the problem gets elevated on adding multiple generators to the system. Since the inception of Artificial Intelligence (AI) in solving ELD [3], the scope of the solution has enhanced with decrease in computation times and balanced equilibrium solutions. One such approach was using neural networks [4]. A grueling approach to ELD was made when both the generation and transmission costs were considered for optimization [5]. But, eventually a polarized approach was recommended for a better convergence. However, these equilibrium points are not necessarily steady state points and are bound to change in the dynamic model and real time situation [6]. The idea of using heuristics to solve these problems stems from the fact that many local optimum points are generated within the solution space, and eventually the computations become restricted around this local point.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 314–322, 2012.

With this in mind, several other techniques were brimming up in the 80s, and one of these methods was the Bacteria Foraging Optimization (BFO). This algorithm was eventually tuned to make the step size adapt as per the problem nature [7]. Eventually, to avoid lags in computation, the nested looping feature was invoked into the algorithm with best iteration count for faster response [8].

Yet, another effective approach was adopted, namely the Co-operative Bacteria Foraging Algorithm involving serial decomposition in search space [9-10]. The hybrid algorithm, on the other hand, integrates the use of BFO and the serial decomposition integrated into a single algorithm. The CBFO-H algorithm is known for its clustering phenomenon, wherein the best value of a single cluster is taken for the next iteration. In this way the foraging process is sectionalized and helps in eradicating local optima conditions.

2 Economic Load Dispatch Formulation

2.1 Objective Function

The primary concern of economic dispatch is the minimization of fuel cost involved. Total cost is taken as the objective function and the fuel cost of a thermal generation unit is given as a second order polynomial as shown. Here a_i, b_i and c_i are the constants derived from the second order power cost curve.

$$F_i(Pg_i) = a_i+b_i.(Pg_i)+c_i.(Pg_i)^2 \tag{1}$$

2.2 Equality Constraints

It reduces the power system to a basic principle of equilibrium between system generation and system loads. Equilibrium is met when the total system generation equals the system losses and the system load as shown. Pgi refers to total generation, Pd refers to the available demand and Pl refers to the losses in the transmission system.

$$\sum_{i=1}^{n} P\,g_i = Pd_i + Pl_i \tag{2}$$

2.3 Inequality Constraints

Generation units have lower and upper bounds of power ratings. These power limits are defined as per generator size. For successful operation of power system, the generator output must be confined within these limits. These bounds are designated by inequality constraints as shown. $Pg_{i,min}$ refers to the minimum limit while $Pg_{i,max}$ is the maximum limit on the generator.

$$Pg_{i,min} < Pg_i < Pg_{i,max} \tag{3}$$

2.4 Ramp Rate Limits

In case of multiple stream valve systems, due to variations in ramp flow rates, accounted due to mechanical inconsistencies, sufficient ripples are produced which causes the ripple effect in the heat curve. So, a sinusoidal component is added with

the main cost function. Being a dynamic process, the AI based dynamic programming is a proper tool to treat this problem. The objective function corresponding to the combined effect of load dispatch and ramp rate effect becomes like equation (4).Here, f_i and e_i are generator coefficients with valve point effects.

$$Fi\ (Pg_i)= a_i+ (b_i).(Pg_i)+ (c_i).(Pg_i^2)+ (e_i).\sin\ (f_i \times (Pg_{i,min}- Pg_i\ )) \quad (4)$$

2.5 Emission Constraints

This is directed to control the extreme effect on environment, where d_i is the emission constant.

$$Fi\ (Pg_i)=a_i+ (b_i).(Pg_i)+ (c_i).(\ Pg_i^2) + \exp(d_i x Pg_{i,min}) \quad (5)$$

3 Bacteria Foraging Algorithm

3.1 Chemotaxis

Every unicellular organism has moving elements which allow it to traverse paths. In case of bacterium, it is called flagella. Its direction of motion determines the movement pattern of the bacterium. When it rotates in clock-wise direction the bacterium swims and anti-clockwise motion leads to tumble. In tumble, bacterium changes its original direction of motion and shifts to a new direction. Here, the run length is decided by the value of chemotactic parameter 'c'. It is the amount of length traversed per chemotactic loop. The tumble vector is decided by the value of Δi, i.e. the rotating vector defined in a random range between [-1,+1] using `unifrnd` function in MATLAB.

3.2 Reproduction

Once the bacterium has travelled too much, it gains better health. So, these bacterium split/reproduce to obtain daughter bacterium which replace the weaker bacteria. The same principle when applied to algorithm gives us variables with weaker fitness being replaced with fitness values of reproductive step. However, on splitting the fitness value, essential optimal solutions are lost. So, this feature is avoided in this paper and a simplified BFO is adopted.

3.3 Elimination and Dispersal

Final stage of bacterial movement involves the elimination of weaker bacterium and random movement of bacterium to avoid getting stuck with local optimum points. To incorporate this feature, generator outputs are extrapolated in different positions using random function. However, to ensure that the generator output does not extend beyond the search space, the random positions are maintained within the previous minimum and maximum values. Dispersal operation vehemently moves the bacterium to random locations, from where Chemotaxis is carried out freshly.

4 Co-operative Bacteria Foraging Algorithm

The cooperative algorithm uses the concept of decomposing a search space with progressive motion of bacteria. This is done to make the search confined within the nearest optimal solution. However the success of the algorithm was significant when the bacterial swarm is clustered into groups of two bacteria each. The unison of two approaches, one involving serial decomposition in search space and the other with clustering gives efficient results in some cases. The overall fitness function on including the valve point effects and emission constraints becomes as shown, where w_1 and w_2 are the weightages given for valve point effects and emission constraints respectively.

$$\text{Fi}\,(\text{Pg}_i) = w_1\left\{a_i + (b_i).(Pg_i) + (c_i).(\text{Pg}_i^2) + (e_i).\sin\left(f_i \times (\text{Pg}_{i,min} - \text{P}_i)\right)\right\} + w_2\left\{a_i + (b_i).(\text{P}_i) + (c_i).(\text{Pg}_i^2) + \exp(d_i x \text{Pg}_{i,min})\right\} \quad (6)$$

PSEUDO CODE (for three generator system):

```
p1,p2 and p3 are three initial generator output values
Parameters bm, em and vlv are defined in matrices
α=2,con=0.1
Chemotactic loop is run for each iteration
Start dispersal loop
for disp = 1 to 10
  define a random vector ∆i
  take average gen. values from overall set PP for p1, p2 and
p3.
  repeat chemotactic loop for p1, p2 and p3
  end chemotactic loop
end for loop
divide PP into subgroups of 2 systems each
for group = 1 : 2 : agents
  perform Chemotaxis for each group of 2 agents
  modify PP
end for
Start dispersal loop
for dispersal = 1 to 10
  define ∆i and get p1, p2 and p3 as done previously
  start chemotactic loop
    {
      ……
    }
  end loop
con=con/α
end dispersal loop
```

The highlighted statements within the chemotactic loop indicate the constraint adjustments performed in the algorithm. Another feature is the mutation operator, which is invoked at the end. This is done to limit the step size to smaller values as the search space collapses in size. The value of bm, em, vlv and α as 2.0 were taken from [10].

5 Test Systems and Results

5.1 A 3 Generator System was Considered and Experimentation Was Done without Valve Point Effects and Emission Constraints

Initial power setting at the start of iteration for BFO, CBFO-s and CBFO-h for two different demands is stated below. The 3 generator system and 6 generator system results for PSO,GA and Conventional method were obtained from [11]. The same parameter settings were used for section 5.3 too.

Load demand 800MW	
Gen 1	400
Gen 2	280
Gen 3	120
Load demand 700MW	
Gen 1	370
Gen 2	270
Gen 3	160

Parameter specifications		
Sl. No	**Operator**	**Magnitude**
1	C (initial) or run length	0.1
2	Mutation operator (α)	2
3	Iteration for BFO process	10
4	Chemotactic loops	100
5	Number of agents	3 and 6 generators
6	Dimensions	10 systems
7	Dispersal steps	10
8	Random variables taken	Range of (-1,1) matrix dimension (3 x 3000)
9	Load tolerance	±5 MW

Table 1. Unit allocation values

BFO in MW				
Load demand	**Gen 1**	**Gen 2**	**Gen 3**	**Total Production**
585	308.318	202.0982	69.799	580.2125
700	374.6064	246.708	75.2276	696.322
800	400.0582	295.2545	100.214	795.9802
CBFO-Serial in MW				
Load demand	**Gen 1**	**Gen 2**	**Gen 3**	**Total Production**
585	312.834	195.833	72.5433	581.2103
700	367.7621	243.9158	84.3906	696.0685
800	393.7477	285.0324	119.1679	797.948
CBFO-Hybrid in MW				
Load demand	**Gen 1**	**Gen 2**	**Gen 3**	**Total Production**
585	311.3531	209.4049	68.098	588.856
700	372.2638	235.5034	88.9727	696.7399
800	400.4107	277.2467	117.868	795.5254

Table 2. Cost comparison for different methods

load demand(MW)	Conventional method	GA (Rs/hr)	PSO (Rs/hr)	BFO (Rs/hr)	CBFO-S (Rs/hr)	CBFO-H (Rs/hr)
585	5821.4	5827.5	5821.4	**5786.4**	5794.7	5794.2
700	6838.4	6877.2	6838.4	6817.2	6811.2	**6805.2**
800	7738.51	7756.8	7738.5	7702.8	**7700.5**	7702.2

Table 3. Compilation time for 1000 chemotactic steps

time of completion	PSO (in sec)	BFO (in sec)	CBFO-serial (in sec)
585	0.141	383.361	192.772
700	0.141	7.135	7.698
800	0.156	36.501	82.369

Table 2 provides corroborative evidence that all units are committed to production under registered ratings. Also, the effect of tolerance is felt on the output produced, which is within ±5%. Table 3 gives a clear picture on the gradual decrease in cost functional values from conventional method to BFO and the variants. However, no such demarcation can be made between BFO, CBFO-s and CBFO-h in terms of economic costs. But, CBFO-S gives economic costs than BFO as seen from Table 3. Table 4 shows that BFO is relatively slower than PSO, but the success of BFO and the other two is in its ability to extricate from local solutions which trap the entire process in those regions. Higher computation times are attributed to the sluggish response of random function and its inability to give the exact value of tumble vector in short time. However, the lower value of cost function undermines this fact.

5.2 A 6 Generator System Was Considered and Experimentation was Done as under

The initial generator power values were taken for BFO, CBFO-s and CBFO-h as under. For simplicity in calculations the average of the upper and lower limits of rated generator outputs were taken as under.

Load Demand: 800 MW	
Gen 1	67.5 MW
Gen 2	60 MW
Gen 3	120 MW
Gen 4	102.5 MW
Gen 5	207.5 MW
Gen 6	242.5 MW

Load Demand: 700 MW	
Gen 1	67.5 MW
Gen 2	60 MW
Gen 3	120 MW
Gen 4	102.5 MW
Gen 5	207.5 MW
Gen 6	142.5 MW

Without Valve Point Effects and Emission Constraints

Table 4. Data of cost estimates

load demand (MW)	PSO (Rs/hr)	BFO (Rs/hr)	CBFO-S (Rs/hr)	CBFO-H (Rs/hr)
700	36987	36185	**36128**	36154
800	42114	41701	41020	**40822**

Table 5. Unit allocation for 700 and 800 MW demand

BFO (in MW)						
Total generation	**Gen1**	**Gen 2**	**Gen 3**	**Gen 4**	**Gen 5**	**Gen 6**
798.202	22.17	39.43	159.83	107.56	209.97	157.99
695.3536	33.33	42.55	141.11	108.48	223.75	246.94
CBFO-S (in MW)						
Total generation	**Gen1**	**Gen 2**	**Gen 3**	**Gen 4**	**Gen 5**	**Gen 6**
695.0058	33.074	50.11	132.7	111.1781	196.75	171.08
796.510	56.797	60.42	130.11	101.67	211.44	236.05
CBFO-H (in MW)						
Total generation	**Gen1**	**Gen 2**	**Gen 3**	**Gen 4**	**Gen 5**	**Gen 6**
699.6797	52.711	42.155	98.840	126.53	208.29	170.92
796.6794	63.197	53.73	110.40	101.1229	226.02	242.19

Table 5 shows lesser costs incurred for CBFO-s at 700 MW and even lesser values for CBFO-h at 800MW, compared to PSO.

With Valve Point Effects and Emission Constraints: Under the six generator system, weightage given to emission constraint was around 35% while to valve point effect was about 65%. By varying the weightage, the effect of oscillations due to ramp limits and environmental considerations can be mitigated.

Table 6. Data of cost estimates

Load demand (MW)	BFO (Rs.)	CBFO-S (Rs.)	CBFO-H (Rs.)
700	**40750**	40940	40843
800	**35758**	36171	35774

Table 7. Unit allocation values at 700 and 800 MW demand

BFO (in MW)						
Total generation	**Gen1**	**Gen 2**	**Gen 3**	**Gen 4**	**Gen 5**	**Gen 6**
798.202	71.34	14.62	101.12	145.556	167.837	297.704
695.3536	53.05	27.54	111.42	141.7228	203.0922	158.5076
CBFO-S (in MW)						
Total generation	**Gen1**	**Gen 2**	**Gen 3**	**Gen 4**	**Gen 5**	**Gen 6**
795.58	54.06	44.84	146.89	85.00	212.10	252.66
698.62	49.148	43.60	122.80	120.55	209.89	152.61
CBFO-H (in MW)						
Total generation	**Gen1**	**Gen 2**	**Gen 3**	**Gen 4**	**Gen 5**	**Gen 6**
798.2	71.348	14.62	101.12	145.55	167.837	297.70
695.35	53.05	27.54	111.48	141.722	203.092	158.50

5.3 Convergence Values

The graph in fig 1 shows the costs incurred with changes in chemotactic loops. It is found that for 3 generator system, the lowest cost attained is at 800 loops, while for 100 loops, it is 7705, close to the data in table 2. The slight deviation (7702) is due to inclusion of valve point effects and emission constraints. Fig 2 indicates a relatively smoother curve compared to the 3 gen. system. Also, lesser number of Chemotaxis movements are needed for 3 generator system than 6. The main point of these findings is the fact that BFO can be contained up to a certain number of chemotactic loops(say 1400 in Fig 2) so that the randomizing parameter does not disturb the solution as seen. Stopping criteria can be invoked and computation times could be decreased as well. Another notable feature of CBFO-s was with the use of ramp rate limits. This can be seen from fig. 3 where the Michalewicz function is shown to vary with agents and dimensions. Due to presence of cosine function, it shows better fitness values with increasing agents, which is analogous to the sinusoidal component of the valve point effect in equation (4). So CBFO-h and even BFO itself provides ulterior optimal points.

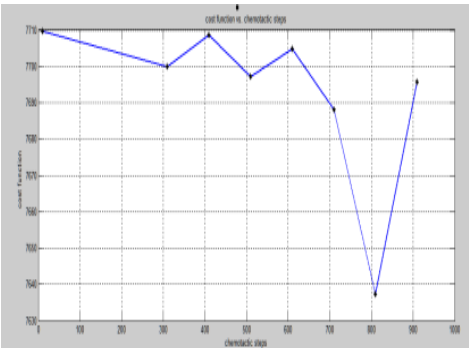


Fig. 1. Function cost vs. chemotactic loops for CBFO-s in 3 generator system with valve point effects

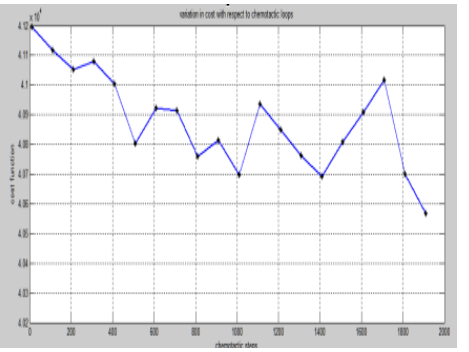


Fig. 2. Function cost vs. chemotactic loops for BFO in 6 generator system without valve point effect

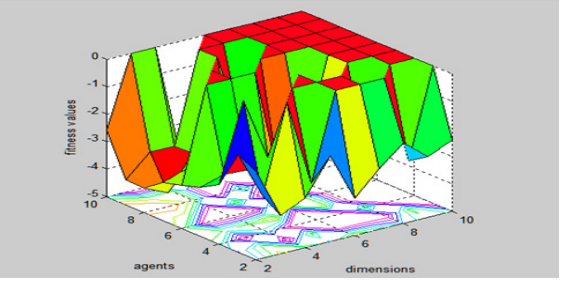


Fig. 3. Fitness value vs. dimensions and agents

6 Conclusion

The proposed BFO, CBFO-S and CBFO-H algorithm were tested successfully on the non-convex economic load dispatch problem with and without valve point effects and emission constraints. The disadvantages of PSO being higher cost values and poor convergence. The computation time was seen to be faster as we increased the number of generators. It was deduced that computation time could be reduced by decreasing number of chemotactic steps too. In CBFO-S even smaller chemotactic steps yielded same results as larger ones. Therefore, computation time reduces in CBFO-S with better convergence. From the results obtained, it can be concluded that Bacteria Foraging Algorithm and its variants are a promising technique for solving complex non-smooth optimization problems in power system operation and the results were comparable with other evolutionary algorithms.

References

1. Parti, S.C., Kothari, D.P., Gupta, P.V.: Economic Thermal Power Dispatch. Institution for Engineers, (India) Journal 64, 123–132 (1983)
2. Yalanoz, T., Altun, H.: Environmentally constrained Economic Dispatch via. G.A. with arithmetic crossover. In: 6th Africon Conference in Africa, October 2-4, vol. 2, pp. 923–928 (2002)
3. Rahman, T.K.A., Yasim, Z.M.: Artificial Immune-Based Foraging for Solving Economic Dispatch in Power system. In: National Power and Energy Conference, vol. 1, pp. 31–35 (2004)
4. Da Silva, I.N., Nepomuceno, L.: An Efficient Neural Approach to Economic Load Dispatch in Power systems. In: Power Engineering Society Summer Meeting, vol. 2, pp. 1269–1274 (2001)
5. Kumaran, G., Mouly, V.S.R.K.: Using Evolutionary Computation to solve the Economic Load Dispatch Problem. In: Congress on Evolutionary Computation, vol. 1, pp. 296–301 (2001)
6. Praveena, P., Vaisakh, K.: A Bacterial Foraging PSO-DE Algorithm for Solving Dynamic Economic Load Dispatch Problem with Security Constraints. In: Joint International Conference on Power Electronics, Drives and Energy Systems (PEDES), pp. 1–7 (2010)
7. Supriyono, H., Tokhi, M.O.: Bacterial Foraging Algorithm with Adaptable Chemotactic step Size. In: Second International Conference on Computational Intelligence, Communication Systems and Networks, pp. 72–77 (2010)
8. Munoz, M.A., Halgamuge, S.K.: Simplifying the Bacteria Foraging Optimization Algorithm. In: IEEE Conference on Evolutionary Computation, pp. 1–7 (2010)
9. Panigrahi, B.K., Yadav, S.Y.: A clonal algorithm to solve Economic Load Dispatch. Electric Power Systems Research 77, 1381–1389 (2007)
10. Panigrahi, B.K., Ravikumar Pandi, V.: An Adaptive Particle Swarm Optimization Approach for Static and Dynamic Economic Load Dispatch. International Journal on Energy Conversion and Management 49, 1407–1415 (2008)
11. Raj, P.A.D.V.: Performance Evaluation of Swarm Intelligence based Power system Optimization Techniques, ch. 2 (November 29, 2010)

Rough Set Based Fuzzy K-Modes for Categorical Data

Indrajit Saha, Jnanendra Prasad Sarkar, and Ujjwal Maulik

Department of Computer Science and Engineering,
Jadavpur University, Kolkata - 700032, West Bengal, India
indra.raju@gmail.com, jpsarkar@outlook.com, umaulik@cse.jdvu.ac.in

Abstract. With the growing demand of categorical data clustering, a new hybrid clustering algorithm, namely Rough set based Fuzzy K-Modes, is proposed in this paper. The principles of rough and fuzzy sets are used in integrated form. It gives the better handling of uncertainty, vagueness, and incompleteness in class definition, while using the concept of lower and upper approximations of rough, on the other hand, the membership function of fuzzy sets enables efficient handling of overlapping partitions. Superiority of the proposed method over state-of-the-art methods is demonstrated quantitatively. For this purpose, two artificial and two real life categorical data sets are used. Also statistical significance test has been carried out to establish the statistical significance of the proposed clustering results.

Keywords: Categorical attributes, fuzzy clustering, rough set, statistical test.

1 Introduction

Clustering [1–4] is a useful unsupervised data mining technique which partitions the input space into K regions depending on some similarity/dissimilarity metric where the value of K may or may not be known a priori. If each data point is assigned to a single cluster, then the clustering is called crisp clustering. On the other hand, if a data point has certain degrees of belongingness to each cluster, the partitioning is called fuzzy. It has a wide application area ranging from engineering to scientific research areas, such as medicine, biology, computer vision, pattern recognition etc.

In last few years, most of the clustering algorithms are designed to focus on numerical data, whose inherent geometric properties can be exploited naturally to define distance functions between data points. However, many real life data sets are categorical in nature, where no natural ordering can be found among the elements in the attribute domain. In such situations, the state-of-the-art clustering algorithms, such as K-Means [1], Fuzzy C-Means (FCM) [5], etc. cannot be applied. K-Means is a traditional partitional clustering algorithm which starts with K random cluster centroids and the centroids are updated in successive iterations by computing the numerical averages of the feature vectors in

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 323–330, 2012.

each cluster. However, as categorical data sets do not have any inherent distance measure, computing the mean of a set of feature vectors is meaningless. A variation of the K-Means algorithm for crisp clustering, namely K-Modes (KMd) [6] has been proposed for such kind of data sets. In KMd, instead of the cluster center, the cluster modes are computed by a frequency based method. Later a fuzzy version of the K-Modes algorithm, i.e., Fuzzy K-Modes (FKMd) and hybrid Genetic Algorithm based Fuzzy K-Modes (GAFKMd) are also proposed in [7] and in [8], respectively. Moreover, another extension of the K-Means is the K-Medoids algorithm [9], which is also widely used for clustering categorical data. However, all these algorithms rely on optimizing a single objective to obtain the partitioning. The major disadvantage of these algorithms is often tend to converge to local optimum solutions. Thus, it has been observed that they are not working uniformly well for uncertain, vague, incomplete and overlapping kinds of categorical data. Hence, we have induced Rough Set [10] as an alternative tool to overcome these drawbacks.

Rough set theory [10] is a new concept and along with Fuzzy set [11], it has been applied for fuzzy modeling, fuzzy-rule extraction, reasoning with uncertainty, ect.. Both rough and fuzzy sets provide a mathematical framework to capture uncertainties that are associated with the data, because they are complementary in some aspects. Recently, it has also been introduced in clustering purposes [12–15] to deal with uncertainty, vagueness, and incompleteness for numerical data sets. Hence, we are attempting to give a humble contribution by proposing a new Rough set based Fuzzy K-Modes (RFKMd) clustering method for categorical data, where each partition is represented by a set of three parameters, namely, a cluster mode, a crisp lower approximation, and a fuzzy boundary. The lower approximation influences the fuzziness of the final partition. The cluster prototype depends on the weighting average of the crisp lower approximation and fuzzy boundary. The effectiveness of the proposed algorithm is demonstrated in comparison with the existing state-of-the-art methods like K-Modes (KMd), Fuzzy K-Modes (FKMd), Average Linkage (AL) [1] hierarchical clustering, Genetic Algorithm based Fuzzy K-Modes (GAFKMd) and Min-Min-Roughness (MMR) [16] algorithms for a set of four benchmark data sets. Also statistical significance tests have been carried out in order to confirm that the superior performance of the RFKMd clustering method is significant and has not occurred by chance.

2 Proposed Rough Set Based Fuzzy K-Modes

In this section, we describe the newly proposed method, called Rough set based Fuzzy K-Modes (RFKMd), which incorporates both the concepts rough and fuzzy sets.

2.1 Initialization

During the initialization stage, the RFKMode randomly selects K number of data objects as modes, where each mode is denoted by v_l, $1 \leq l \leq K$, from

Assumption: Number of clusters is given as K

1. Select random K objects from data set as K clusters. Say v_l is the mode for $l = 1, 2, \ldots, K$.
2. Set a value for fuzzifier m'.
3. Calculate μ_{li} for all n objects using Eqn. 1.
4. Compute difference of maximum two out of K fuzzy membership values for all the n objects. Let μ_{li} is the highest and μ_{hi} is the second highest fuzzy membership value of object x_i among all K clusters where $1 \leq l, h \leq K$ *and* $h \neq l$.
5. Compute the threshold value, ∂. ∂ is the median of $(\mu_{li} - \mu_{hi}), \forall i = 1, 2, \ldots, n$
6. Check if the object is eligible to be classified to lower approximation of any cluster.
   (a) if $(\mu_{li} - \mu_{hi}) > \partial$ then x_i can be exactly classified to lower approximation of cluster, X_l. Also it is in upper approximation of that cluster as per rough set theory. So in this case x_i can be classified to $\underline{B}(X_l)$ and $\overline{B}(X_l)$.
   (b) Set $\mu_{li} = 1$ and $\mu_{hi} = 0, \forall h = 1, 2, \ldots, K$ *and* $h \neq l$.
7. Otherwise x_i can belong to upper approximation of multiple clusters.
   (a) Hence, x_i belong to $\overline{B}(X_l)$ and $\overline{B}(X_h)$.
   (b) Keep μ_{hi} unchanged $\forall h = 1, 2, \ldots, K$ where $(\mu_{li} - \mu_{hi}) \leq \partial$.
8. Compute new mode with the help of Eqn. 3.
9. Repeat steps(3) to (9) till it converges.

Fig. 1. RFKMd Algorithm

the data set of $\{x_i |\ 1 \leq i \leq n\}$, where n is the number of categorical objects. Thereafter, fuzzy membership value is calculated for all the data objects as follow:

$$\mu_{li} = \begin{cases} 1, & if x_i = v_l \\ 0, & if x_i = v_h, l \neq h \\ \frac{1}{\sum_{h=1}^{K} [\frac{D(v_l, x_i)}{D(v_h, x_i)}]^{\frac{1}{m'-1}}} & if x_i \neq v_l, x_i \neq v_h, 1 \leq h \ \& \ l \leq K \end{cases} \quad (1)$$

Here, $D(v_l, x_i)$ is the dissimilarity measure between cluster mode v_l and object x_i. While m' is the fuzzy exponent.

2.2 Objective Evaluation

The RFKMd uses the Eqn. 2 to compute the objective. According to rough set concept, if U represents the set of whole data objects, then every set, $X_l \subseteq U$ for $1 \leq l \leq K$, where K number of clusters, can be represented as a pair of lower and upper approximation $[\underline{B}(X_l), \overline{B}(X_l)]$. Hence, $BN(X_l) = [\overline{B}(X_l) - \underline{B}(X_l)]$ is the boundary region of cluster X_l and v_l is the mode of cluster X_l. Then the objective function J_{RFKMd} is to compute as follow:

$$J_{RFKMd} = \begin{cases} \omega_{low} \times P + \omega_{up} \times Q, & if \underline{B}(X_l) \neq \emptyset, BN(X_l) \neq \emptyset \\ P, & if \underline{B}(X_l) \neq \emptyset, BN(X_l) = \emptyset \\ Q, & if \underline{B}(X_l) = \emptyset, BN(X_l) \neq \emptyset \end{cases} \quad (2)$$

$$P = \sum_{l=1}^{K} \sum_{x_i \in \underline{B}(X_l)} D(v_l, x_i), \quad Q = \sum_{l=1}^{K} \sum_{x_i \in BN(X_l)} (\mu_{li})^{m'} D(v_l, x_i), \quad 1 \leq i \leq n$$

where ω_{low} and ω_{up} denote the relative importance of lower approximation and boundary region with the relation $\omega_{low} + \omega_{up} = 1$ and $0 < \omega_{low}, \omega_{up} < 1$. According to the rough set theory any object, x_i can belong to at most one lower approximation. If x_i belongs to lower approximation of any cluster then x_i is also in upper approximation of that cluster. In that case, we definitely can say that x_i belong to that cluster. Hence, the fuzzy membership value of that object is 1 and do not influence any other cluster. If x_i cannot be exactly classified to lower approximation of any cluster then the object can belong to upper approximation of multiple clusters. In that case, x_i cannot belong to lower approximation of any cluster. Hence, ω_{low} and ω_{up} are related as $0 < \omega_{up} < \omega_{low} < 1$.

2.3 Updating Modes

In mode updating process, we have assumed that, U is the set of categorical objects with categorical attributes $A_1, A_2, \ldots, A_m$, where m is the number of attributes and $DOM(A_j) = \{a_1, a_2, \ldots, a_{n_j}\}$ for $1 \leq j \leq m$, where n_j = the number of categories of attribute A_j. Also let v_l is the cluster mode denoted by $[v_{l1}, v_{l2}, \ldots, v_{lm}]$ for $1 \leq l \leq K$ and ith object is denoted as x_i with m categorical attributes as $x_{i1}, x_{i2}, \ldots, x_{im}$. Then the Eqn. 2 is to optimize if $v_{lj} = a_r \in DOM(A_j)$ where

$$r = argmax_{1 \leq t \leq n_j} \begin{cases} \sum_{1 \leq i \leq n, x_i \in \underline{B}(X_l), x_{ij} = a_{jt}} (\omega_{low}) \\ + \sum_{1 \leq i \leq n, x_i \in BN(X_l), x_{ij} = a_{jt}} (\omega_{up} \times (\mu_{li})^{m'}) \\ \qquad if \underline{B}(X_l) \neq \emptyset, BN(X_l) \neq \emptyset \\ | \{x_i \in \underline{B}(X_l) : x_{ij} = a_{jt}, 1 \leq i \leq n\} | \\ \qquad if \underline{B}(X_l) \neq \emptyset, BN(X_l) = \emptyset \\ \sum_{1 \leq i \leq n, x_i \in BN(X_l), x_{ij} = a_{jt}} ((\mu_{li})^{m'}) \\ \qquad if \underline{B}(X_l) = \emptyset, BN(X_l) \neq \emptyset \end{cases} \tag{3}$$

where $X_l \subseteq U$ and $| \{\ldots\} |$ signifies the cardinality of the set.

The mode of the cluster is a point, whose attribute values are computed according to the concept of rough set, where each cluster has two parts - lower approximation and upper approximation. Therefore, maximum appearance of any category of each attribute is calculated by combining two parts - one for lower approximation and other for boundary region. The fuzzy membership value of any object in lower approximation is 1, whereas the membership value in boundary region is same as fuzzy membership value. Hence, maximum appearance of any category in lower approximation is equal to the summation of how many times it has appeared in that attribute for the particular cluster. On the other hand, maximum appearance of any category in boundary region is also equal

to the summation of fuzzy membership values of the attribute for a category in the particular cluster. Thereafter, the mode is calculated in combination of both the facts. Moreover, in Eqn. 3, if lower approximation area and boundary region both are not empty then a relative importance is given to both lower approximation area and boundary region. ω_{low} is multiplied with lower approximation and ω_{up} is multiplied with boundary region for that particular cluster. Different steps of the RFKMd algorithm are given in Fig. 1.

3 Experimental Results

3.1 Synthetic and Real Life Data Sets

Cat_100_8_3: This synthetic data set[1] has a one-layer clustering structure with 8 attributes and 100 points. It has 3 clusters.
Cat_300_15_5: This is a synthetic data set with 300 points and 15 attributes. The data set has 5 clusters.
Soybean: The Soybean data set contains 47 data points, each having 35 categorical attributes classified as one of the four diseases.
Zoo: The Zoo data consists of 101 instances of animals in a zoo with 17 features. The data set consists of 7 different classes of animals.

3.2 Distance Measures

Distance between two categorical objects is computed as in [17]. Let two categorical objects described by p categorical attributes are $x_i = [x_{i1}, x_{i2}, \ldots, x_{im}]$, and $x_j = [x_{j1}, x_{j2}, \ldots, x_{jm}]$. The distance measure between x_i and x_j , $D(x_i, x_j)$, can be defined by the total number of mismatches of the corresponding attribute categories of the two objects. Formally,

$$D(x_i, x_j) = \sum_{k=1}^{m} \delta(x_{ik}, x_{jk}), \quad where\ \delta(x_{ik}, x_{jk}) = \begin{cases} 0 & \text{if } x_{ik} = x_{jk} \\ 1 & \text{if } x_{ik} \neq x_{jk} \end{cases} \tag{4}$$

3.3 Performance Metrics and Input Parameters

Performance of the KMd, FKMd, AL, GAFKMd, MMR and RFKMd is evaluated by the measure of Minkowski Score (MS) [18] and Percentage of Correct Pair (%CP) [19]. Note that, after completion of the RFKMd clustering method, the fuzzy membership matrix is used to assign the rough data objects to their respective cluster. The FKMd, FKMd, GAFKMd, MMR and RFKMd algorithms are executed till it converges to the final solution and the fuzzy exponent (m') is set to 2. Also for RFKMd algorithm, the value of ω_{low} is given as 0.95. However, the performance of the RFKMd depends on the parameter $\partial, m', \omega_{low}$ and ω_{up}.

[1] http://www.datgen.com

Table 1. Average MS and %CP values over 20 runs of different algorithms for four categorical data sets

Data Sets	Algorithms	MS	%CP
Cat_100_8_3	KMd	0.88202	76.18283
	FKMd	0.79713	79.91267
	AL	0.76037	81.13111
	GAFKMd	0.75332	83.11590
	MMR	0.55900	88.29480
	RFKMd	0.43724	92.75930
Cat_300_15_5	KMd	0.84465	77.51644
	FKMd	0.74558	82.68290
	AL	0.73566	83.51650
	GAFKMd	0.71335	84.76120
	MMR	0.64432	86.63930
	RFKMd	0.53695	89.07250
Soybean	KMd	0.64363	86.79560
	FKMd	0.39077	92.96750
	AL	0.44982	91.72570
	GAFKMd	0.37457	93.45920
	MMR	0.33104	95.94930
	RFKMd	0.22922	97.15681
Zoo	KMd	0.68839	84.82920
	FKMd	0.43895	92.29820
	AL	0.45844	91.49410
	GAFKMd	0.42973	92.88530
	MMR	0.39099	93.89980
	RFKMd	0.36688	94.70160

3.4 Performance

Table 1 shows the comparative results obtained for the four data sets. It can be noted from the table that the proposed RFKMd consistently outperforms other five state-of-the-art and receltly proposed methods in terms of the MS and %CP values. For example, for Soybean, the RFKMd technique achieves better average MS and %CP values of 0.22922 and 97.15681 while the KMd, FKMd, AL, GAFKMd and MMR provide values of 0.64363, 86.79560, 0.39077, 92.96750, 0.44982, 91.72570, 0.37457, 93.45920 and 0.33104, 95.94930, respectively. Similar results are also found for the other data sets.

3.5 Statistical Significance Test

In this article, a statistical significance test called t-test [20] has been carried out at the 5% significance level, to establish that the better average MS values provided by RFKMd is statistically significant and does not come by chance.

Table 2 reports the results of the t-test for the four data sets. The null hypothesis (the means of two groups are equal) are shown in the table. The alternative hypothesis is that the mean of the first group is larger than the mean of the second group. For each test, the degree of freedom is $M + N$ - 2, where M and N are the sizes of two groups considered. Here $M = N = 20$. Hence the degree of freedom is 38. Also the values of t-statistic and the probability (P-value) of accepting the null hypothesis are shown in the table. It is clear from the table that the P-values are much less than 0.05 (5% significance level) which are strong evidences for rejecting the null hypothesis. This proves that the better average MS values produced by the RFKMd method is statistically significant and has not come by chance.

Table 2. The t-test results for four categorical data sets

Data Sets	Test No.	Null hypothesis (H0 : $\mu_1 = \mu_2$)	t-test statistic	P-value	Accept/Reject
Cat_100_8_3	1	$\mu_{RFKMd} = \mu_{KMd}$	41.1226	8.4323e-021	Reject
	2	$\mu_{RFKMd} = \mu_{FKMd}$	28.5558	1.1015e-017	Reject
	3	$\mu_{RFKMd} = \mu_{AL}$	24.4567	3.1204e-015	Reject
	4	$\mu_{RFKMd} = \mu_{GAFKMd}$	18.7901	2.3322e-012	Reject
	5	$\mu_{RFKMd} = \mu_{MMR}$	11.4767	2.9768e-010	Reject
Cat_300_15_5	1	$\mu_{RFKMd} = \mu_{KMd}$	58.8853	6.7970e-024	Reject
	2	$\mu_{RFKMd} = \mu_{FKMd}$	35.4813	1.5522e-019	Reject
	3	$\mu_{RFKMd} = \mu_{AL}$	27.3157	2.6205e-017	Reject
	4	$\mu_{RFKMd} = \mu_{GAFKMd}$	19.9846	1.0960e-014	Reject
	5	$\mu_{RFKMd} = \mu_{MMR}$	11.1720	2.4170e-010	Reject
Soybean	1	$\mu_{RFKMd} = \mu_{KMd}$	24.3057	2.5357e-016	Reject
	2	$\mu_{RFKMd} = \mu_{FKMd}$	16.5700	5.9619e-014	Reject
	3	$\mu_{RFKMd} = \mu_{AL}$	14.4637	4.7007e-012	Reject
	4	$\mu_{RFKMd} = \mu_{GAFKMd}$	9.1084	2.8047e-009	Reject
	5	$\mu_{RFKMd} = \mu_{MMR}$	6.6068	1.9571e-006	Reject
Zoo	1	$\mu_{RFKMd} = \mu_{KMd}$	14.8107	3.0411e-012	Reject
	2	$\mu_{RFKMd} = \mu_{FKMd}$	10.0465	2.9264e-009	Reject
	3	$\mu_{RFKMd} = \mu_{AL}$	8.5231	4.3218e-008	Reject
	4	$\mu_{RFKMd} = \mu_{GAFKMd}$	6.7588	1.4206e-006	Reject
	5	$\mu_{RFKMd} = \mu_{MMR}$	4.06331	3.6035e-003	Reject

4 Conclusion

In this paper, we have developed a new Rough set based Fuzzy K-Modes clustering method for categorical data. For this purpose, we have used the amalgamation of both fuzzy set and rough set concept to deal efficiently the overlapping partition area, as well as the problem of uncertainty, vagueness and incompleteness. The superior of the proposed method has been demonstrated by comparing with the K-Modes, Fuzzy K-Modes, Average Linkage, Genetic Algorithm based Fuzzy K-Modes and recently developed method, Min-min-Roughness for four categorical data sets.

As a scope for future research, Simulated Annealing and Genetic Algorithm can be used to extend Rough set based Fuzzy K-Modes clustering. Also, the different dissimilarity measures [21, 22] need to be studied.

Acknowledgements. Mr. Saha is grateful to the All India Council for Technical Education (AICTE) for providing National Doctoral Fellowship (NDF) to support the work.

References

1. Jain, A.K., Dubes, R.C.: Data clustering: A review. ACM Computing Surveys 31 (1999)
2. Maulik, U., Saha, I.: Modified differential evolution based fuzzy clustering for pixel classification in remote sensing imagery. Pattern Recognition 42(9), 2135–2149 (2009)

3. Saha, I., Maulik, U., Bandyopadhyay, S.: An Improved Multi-objective Technique for Fuzzy Clustering with Application to IRS Image Segmentation. In: Giacobini, M., Brabazon, A., Cagnoni, S., Di Caro, G.A., Ekárt, A., Esparcia-Alcázar, A.I., Farooq, M., Fink, A., Machado, P. (eds.) EvoWorkshops 2009. LNCS, vol. 5484, pp. 426–431. Springer, Heidelberg (2009)
4. Saha, I., Maulik, U., Plewczynski, D.: A new multi-objective technique for differential fuzzy clustering. Applied Soft Computing 11(2), 2765–2776 (2011)
5. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum, New York (1981)
6. Huang, Z.: Extension of k-means algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery 2, 283–304 (1998)
7. Huang, Z., Ng, M.K.: A fuzzy k-modes algorithm for clustering categorical data. IEEE Transactions on Fuzzy Systems 7(4) (1999)
8. Gan, G., Wu, J., Yang, Z.: A genetic fuzzy k-Modes algorithm for clustering categorical data. Expert Systems with Applications 36, 1615–1620 (2009)
9. Kaufman, L., Roussenw, P.J.: Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley & Sons, NY (1990)
10. Pawlak, Z.: Rough Sets: Theoretical Aspects of Resoning About Data. Kluwer Academic, MA (1992)
11. Dubois, D., Prade, H.: Rough fuzzy sets and fuzzy rough sets. International Journal of General Systems 17(2/3), 191–209 (1990)
12. Lingras, P., West, C.: Interval set clustering of web users with rough k-means. Journal of Intelligent Information Systems 23(1), 5–16 (2004)
13. Mitra, S., Banka, H., Pedrycz, W.: Rough-fuzzy collaborative clustering. IEEE Transactions on Systems, Man, and Cybernetics - Part B 36(4), 795–805 (2006)
14. Maji, P., Pal, S.K.: Rough set based generalized fuzzy c-means algorithm and quantitative indices. IEEE Transactions on Systems, Man, and Cybernetics - Part B 37(6), 1529–1540 (2007)
15. Maulik, U., Bandyopadhyay, S., Saha, I.: Integrating clustering and supervised learning for categorical data analysis. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans 40, 664–675 (2010)
16. Parmar, D., Wu, T., Blackhurst, J.: MMR: An algorithm for clustering categorical data using rough set theory. Data and Knowledge Engineering 63, 879–893 (2007)
17. Vermeulen-Jourdan, L., Dhaenens, C., Talbi, E.-G.: Clustering Nominal and Numerical Data: A New Distance Concept for a Hybrid Genetic Algorithm. In: Gottlieb, J., Raidl, G.R. (eds.) EvoCOP 2004. LNCS, vol. 3004, pp. 220–229. Springer, Heidelberg (2004)
18. Jardine, N., Sibson, R.: Mathematical Taxonomy. John Wiley and Sons (1971)
19. Mukhopadhyay, A., Bandyopadhyay, S., Maulik, U.: Clustering using multi-objective genetic algorithm and its application to image segmentation. In: Proc. IEEE International Conference on Systems, Man and Cybernetics (SMC 2006), vol. 3, pp. 2678–2683 (2006)
20. Ferguson, G.A., Takane, Y.: Statistical analysis in psychology and education (2005)
21. He, Z., Xu, X., Deng, S.: Attribute value weighting in k-modes clustering. Expert Systems with Applications 38, 15365–15369 (2011)
22. Cao, F., Liang, J., Li, D., Bai, L., Dang, C.: A dissimilarity measure for the k-Modes clustering algorithm. Knowledge-Based Systems 26, 120–127 (2012)

Teaching Learning Opposition Based Optimization for the Location of Median Line in 3-D Space

Anguluri Rajasekhar[1] and Swagatam Das[2]

[1] Dept of Electrical and Electronics Engineering, National Institute of Technology-Warangal
[2] Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata
{rajasekhar.anguluri,swagatam.das}@ieee.org

Abstract. In this paper we propose a new Teaching Learning Opposition Based Optimization (TLOBO) method for the location of median line in the Euclidean 3-D space. Optimization is carried out in such a way that, the sum of distances from the line to a given finite set of three-dimensional data points is minimized. The results obtained using TLOBO are compared with other state-of-are methods like TLBO and ABC which shows the superiority of TLOBO in finding median line problem.

Keywords: Opposition, Global Optimization, median line location.

1 Introduction

Median Line Location problem in the Euclidean three-dimensional (3-D) is a line which minimizes the sum of Euclidean distances to some given data or demand points in $\Re^3$. In the context of location theory, median line problem in two dimensions was first analyzed by Wesolowsky [1]. According to Wesolowsky there exists an optimal line intersecting two data points which leads to a polynomial-time solution algorithm. Lot of research had been carried out in locating median line problem in two dimensions; few of them include general distance measures, line segments etc., [2-4].

Even though the Euclidean two-dimensional median line problem is a quite challenging optimization problem and exact polynomial time algorithms are available, the 3-D line location problem becomes much harder. A 3-D median line problem with some restrictions was studied in Brimberg et al [5]. In this paper we considered a parameterized oriented model proposed by Blanquero et al [6] for line location and the authors had made use of branch and bound optimization procedure for solving this location problem.

To obtain the optimal line with less computational complexity we proposed a new variant of "Teaching Learning based optimization" (TLBO) method known as "Teaching Learning Opposition based Optimization" (TLOBO) based on the concept of Opposition-based learning. Opposition-based learning (OBL) works on the principle of simultaneous consideration of a guess and opposite guess in order to achieve a better approximation for the current candidate solution. This concept of *opposition-based learning* (OBL) was introduced by Tizhoosh [7] and has been applied to

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 331–338, 2012.

accelerate reinforcement learning [8]. In this paper opposition has been applied to increase the convergence rate with a promising solution as well.

The rest of this paper is structured as follows. In Section 2 the problem formulation and reduction of the parameters to be optimized are discussed. Section 3 & 4 briefs the TLBO and proposed Opposition based TLBO respectively. Experimental studies of TLOBO and comparisons along with other state of art methods are summarized in Section 5. In Section 6 we put forth some conclusions and future ideas.

2 Problem Formulation

In this section we provide an objective function to be minimized based on the proofs and lemmas provided in [6]. A line *r* in $\Re^3$ has the form

$$r = r(x,d) = \{x + td : t \in \Re\}, \tag{1}$$

where $d \in \Re^3 \setminus \{0\}$ is the direction of *r* and $x \in \Re^3$. Moreover the following notation is used to denote a Euclidean distance from a point to line. For any $a \in \Re^3$ and $x, d \in \Re^3$ with $d \notin 0$ denote by

$$\delta_a(x,d) := \min_{t \in \Re} \|x + td - a\|_2 \tag{2}$$

δ_a is the Euclidean distance from a to the line $r(x,d)$.

The line $r(x,d)$ is not crisply defined by pair (x,d). It is defined as $r(x,d) = r(x+ud,d)$ for any $u \in \Re$. Hence it can be assumed that x is the intersection of r with the hyperplane

$$H_d = \{y \in \Re^3 : d^T y = 0\}. \tag{3}$$

For any $a \in \Re^3$ and $x, d \in \Re^3$ with $d \notin 0$ and $d^T x = 0$

The median line problem with fixed direction $d \in \Re^3 \setminus \{0\}$ and Hyperplane H_d defined in Eqn (3) can be shown equivalent to a planar Weber problem [9]. At last a mapping is defined

$$p_d : \Re^3 \to H_d \text{ with } p_d(x) = x - \frac{d^T x}{d^T d} \cdot d$$

here p_d is the projection of x onto H_d. For a fixed direction $d \in \Re^3 \setminus \{0\}$ and also for all $x, a \in \Re^3$

$$\delta_a(x,d) = \left\| p_d(x) - p_a(a) \right\|_2 \tag{4}$$

Hence 3-D median line problem with $d \in \Re^3 \setminus \{0\}$ is equivalent to a 2-D Weber problem. It is mathematically represented as

$$\min_{x\in\Re^3} \sum_{k=1}^{n} \delta_a(x,d) = \min_{x\in\Re^3} \sum_{k=1}^{n} \left\| p_d(x) - p_a(a) \right\|_2 = \min_{x\in H_d} \sum_{k=1}^{n} \left\| x - p_a(a) \right\|_2 \tag{5}$$

As said before that, this median line problem is equivalent to a planar Weber problem for any fixed $d \in \Re^3 \setminus \{0\}$. There exists an optimal solution x^* to weber problem which intersects the convex hull of projected (demand) points $A_d=\{p_d(a_1),..., p_d(a_1)\}$ [9], i.e., x^* is the median of $p_d(a_1),...,p(a_2) \in H_d$. From above discussion for any fixed $d \in \Re^3 \setminus \{0\}$ there exists a $x^* \in H_d$, such that

$$\begin{aligned} \min_{x\in H_d} \sum_{k=1}^{n} \left\| x - p_a(a) \right\|_2 &= \sum_{k=1}^{n} \left\| x^* - p_a(a) \right\|_2 = \sum_{k=1}^{n} \left\| p(x^*) - p_a(a) \right\|_2 \\ &= \sum_{k=1}^{n} \delta_a(x^*,d) = \min_{x\in\Re^3} \sum_{k=1}^{n} \delta_a(x^*,d), \end{aligned} \tag{6}$$

2.1 Problem Parameterization

From above defined mathematical representations it is very clear that the optimization problem to be solved is a six-dimensional problem which can be reduced to a four-dimensional problem. For any $\tau \neq 0$ we have $r(x,d) = r(x,\tau d)$ and thus it can be assumed without loss of generality that $\left\| d \right\|_\infty = 1$. Hence, any line $r = r(x,d)$ can be parameterized by its associated pair (x,d) with $\left\| d \right\|_\infty = 1$ and $d^T x = 0$. Since $r(x,d) = r(x,-d)$ the following assumption can be considered.

$$\max_{i=1,2,3} |d_i| = \max_{i=1,2,3} d_i = 1 \tag{7}$$

Supposing $d = (d_1, d_2, d_3) \in \Re^3$ is satisfying and assuming $d_3 = 1$ is fixed. Then by considering $x = (x_1, x_2, x_3) \in \Re^3$ such that $d^T x = 0$, the following result can be obtained

$$x_3 = -(x_1 d_1 + x_2 d_2)$$

Now a reduced objective function (in the case $d_3 = 1$) can be formulated for a given facility (demand) points $a_k = (\alpha_k, \beta_k, \gamma_k)$ where $k \in 1,...,n$.

$$f_3(x_1, x_2, d_1, d_2) := \frac{1}{\sqrt{d_1^2 + d_2^2 + 1}} \cdot \sum_{k=1}^{n} \sqrt{g_3^k(x_1, x_2, d_1, d_2)},$$

where

$$g_3^k(x_1, x_2, d_1, d_2) := ((x_1 - \alpha_k)^2 + (x_2 - \beta_k)^2 + (x_1 d_1 + x_2 d_2 + \gamma_k)^2) \\ .(d_1^2 + d_2^2 + 1) - (d_1 \alpha_k + d_2 \beta_k + \gamma_k)^2.$$

Similarly, by fixing $d_1 = d_2 = 1$; $f_1(x_1, x_2, d_1, d_2)$ and $f_2(x_1, x_2, d_1, d_2)$ can also be obtained (renaming the four remaining variables always as x_1, x_2, d_1 and d_2 without loss of generality)

Hence, the six-dimensional problem (before parameterization) can be made equivalent to four-dimension problem which is stated as

$$\min_{x_1, x_2, d_1, d_2 \in \Re} f(x_1, x_2, d_1, d_2) \tag{8}$$

with

$$f(x_1, x_2, d_1, d_2) := \min\{f_1(x_1, x_2, d_1, d_2), f_2(x_1, x_2, d_1, d_2), \\ f_3(x_1, x_2, d_1, d_2)\}.$$

3 Teaching Learning Opposition based Optimization

3.1 Teaching Learning Based Optimization

Teaching Learning Based Optimization or simply TLBO is a new meta-heuristic optimization algorithm proposed by Rao et al. [10]. The search strategy of algorithm falls under two categories (i) Teacher Phase and (ii) Learner Phase.

1. Teacher Phase

In this phase a teacher tries to ameliorate the mean result of class in the subject taught by him/her based on level of knowledge and skill he/she had in that particular subject. For any i_{th} iteration, let us consider there are m number of subjects (design variables), n number of learners (population size, k=1, 2,..,n) and $T_{j,i}$ be the mean result of the learners in j_{th} subject (where j=1,2,…m). However the best overall result $X_{total\text{-}k\text{-}best,i}$ (considering all the subjects together) in a class of learners can be considered as result of best leaner *k-best* and the best learner identified is replaced by the teacher. As the

teacher $X_{total\text{-}k\text{-}best,i}$ will try to move mean T_i towards its own level, an adaptive heuristic is used to update the solution and is done according to the difference between the existing mean result of each subject and the corresponding result of the teacher for each subject is given by.

$$Difference\_mean_{j,k,i} = rand_i\left(X_{j,k-best,i} - T_F T_{j,i}\right) \tag{9}$$

where T_F is termed as teaching factor, which decides whether the value of mean is to be changed or not. The value of T_F can be either 1 or 2 which is decided randomly with equal probability and $rand_i$ is a random number in the range [0, 1]. $X_{j,k\text{-}best,i}$ is the result of the teacher in subject j. $Difference\_Mean_{j,k,i}$, defined in Eqn (9) is used in updating the existing solution according to the following expression

$$X^{new}_{j,k,l} = X_{j,k,l} + Difference\_Mean_{j,k,i} \tag{10}$$

where $X^{new}_{j,k,i}$ and $X_{j,k,i}$ are the new and existing values corresponding to j_{th} subject of k_{th} learner of i_{th} iteration. A greedy mechanism is performed between $X^{new}_{j,k,i}$ and $X_{j,k,i}$, the learner with better function value is retained. After all the learners completed their knowledge update, then they are ready to enter the learner phase and learner phase depends on the teacher phase.

3.2 Learner Phase

Learners always try to increase their knowledge and one of best way can they do is by interacting among themselves. In the course of time a learner may interact randomly with other learners with the help of communications, discussions, etc. For a class of n learners the learning phenomenon of this phase is expressed with following pseudo code.

Pseudo code of Learner Phase

For k = 1 to n

Randomly select another learner Q, such that $X^{new}_{total-k,i} \neq X^{new}_{total-Q,i}$

IF $X^{new}_{total-k,i} < X^{new}_{total-Q,i}$

$$\mathbf{X}^{new}_{j,k,i} = X^{new}_{j,k,i} + rand_i\left(X^{new}_{j,k,i} - X^{new}_{j,Q,i}\right)$$

ELSE

$$\mathbf{X}^{new}_{j,k,i} = X^{new}_{j,k,i} + rand_i\left(X^{new}_{j,Q,i} - X^{new}_{j,k,i}\right)$$

End IF

End FOR

Accept $\mathbf{X}^{new}_{j,k,i}$ if it gives a better function value.

4 Teaching Learning Opposition Based Optimization

4.1 Opposition-Based Optimization

Let $P = \{x_1, x_2, ..., x_D\}$ be a point in D-dimensional space, where $x_1, x_2, ..., x_D \in R$ and $x_i \in [a_i, b_i]\ \forall i \in \{1,2,...,D\}$. Now the opposite point $P^{'} = \{x_1^{'}, x_2^{'}, ..., x_D^{'}\}$ is defined as

$$x_i^{'} = a_i + b_i - x_i \tag{11}$$

Now, with above definition of opposite point the opposition based optimization can be formulated as follows. Assuming $f(\cdot)$ is fitness function via which candidate fitness is measured and according to the above given definitions of P and $P^{'}$, if $f(P^{'}) \geq f(P)$ then the point P can be replaced with $P^{'}$; hence, the point and its opposite point are evaluated simultaneously in order to go with the fitter one.

4.2 Proposed Algorithm

Opposition scheme discussed above is applied two times for the proposed TLOBO method at starting of teaching phase and learning phase respectively. Once the algorithm has started with random initial population simultaneously opposite population are also calculated and then best *n* values are picked up (based on the fitness value) and then passed in to the teacher phase. Similarly before entering in to the learning phase opposite population is evaluated and the best *n* values are passed in to the learning phase and the rest is same as that of TLBO. This is continued till the termination criterion is reached.

To get a better performance the limits a_i and b_i should be updated dynamically in terms of search space of the current population. Instead of using predefined interval boundaries $[a_i, b_i]$ here we used the minimum and maximum values ($[a_j^{\min}, b_j^{\max}]$) of each dimension in current population to calculate the opposite population. This type of opposition helps the learners to get good information and it is computed as

$$OP_{i,j} = a_j^{\min} + b_j^{\max} - P_{i,j} \tag{12}$$

where $P_{i,j}$ is the j_{th} vector of the i_{th} learner in the population. $OP_{i,j}$ is the opposite position of $P_{i,j}$; $a_j^{\min}$ and $b_j^{\max}$ are the minimum and maximum values of the j_{th} dimension in current population respectively.

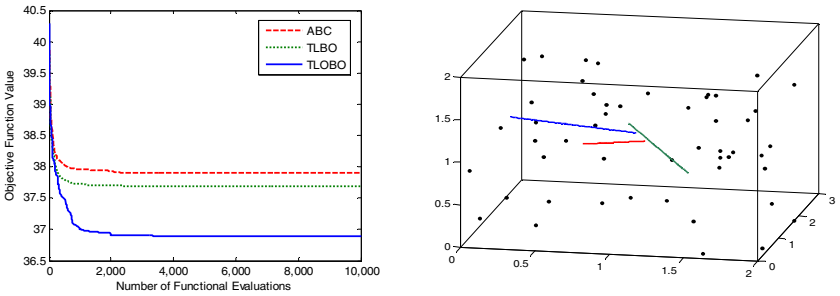
5 Experimental Section

5.1 Problem Instance Considered

To validate the performance of proposed approach on the 3-D median line location problem we had considered a problem instance with *n=50* demand points (presented in [6])

5.2 Discussion on Results

The main aim of the paper is to reduce the time complexity of solving this median line location problem. From the Table 1 it is very clear that although the optimum distance is same for the TLOBO method and Branch and Bound (BB) method the time of execution of TLOBO method is appreciable when compared to BB method. The proposed method is also compared with the existing optimization methods like Artificial Bee Colony [11], original TLBO [10] and the results and their convergence is shown in Table 1, Table 2 and Fig.1 respectively.



(a) (b)

Fig. 1. (a) Convergence of TLOBO with different competitor methods (b) Optimal Line for problem discussed for different methods (legend of fig 1(a) is also same for fig 1(b))

Table 1. Mean and Standard Deviation (STD) of the Best-of-Run Solution for 30 Runs of 3 Algortihms along with branch and bound method on Median Line Location Problem

Method	Mean	STD	Avg time per run	No. Of iterations
ABC	37.8953	1.3321E+00	6.956 sec	500
TLBO	37.6833	9.5278E-01	4.092 sec	500
TLOBO	**36.8930**	**3.3569e-04**	**4.091 sec**	500
Branch & Bound	36.8932	-	47.62 sec	1,223,403

Table 2. Optimal $r(x^*, d^*)$ values obtained for TLOBO, TLBO and ABC

TLOBO	TLBO	ABC
$\begin{pmatrix}1.0204\\1.1751\\1.1188\end{pmatrix}+t.\begin{pmatrix}-0.9819\\1.0000\\-0.1549\end{pmatrix}$	$\begin{pmatrix}0.5454\\1.4569\\0.8437\end{pmatrix}+t.\begin{pmatrix}1.0000\\-0.5372\\0.2812\end{pmatrix}$	$\begin{pmatrix}1.1046\\0.9833\\1.0816\end{pmatrix}+t.\begin{pmatrix}-0.5527\\1.0000\\-0.3447\end{pmatrix}$

6 Conclusions

An new modified TLBO method based on opposition has been proposed for locating the median line in a set of facility (demand) points such that the sum of distances from the line to a given finite set of three-dimensional data points is minimized. From Experimentation section it is very clear that the proposed method had outperformed ABC, TLBO in terms of objective function value, and average time per run. Our further research would focus on solving line location problem for much more complex environment and also location of structures in a plane etc.

References

1. Wesolowsky, G.O.: Location of the median line for weighted points. Environment and Planning A 7(2), 163–170 (1975)
2. Morris, J.G., Norback, J.P.: Linear facility location-Solving extensions of the basic problem. European Journal of Operational Research 12, 90–94 (1983)
3. Korneenko, N.M., Martini, H.: Hyperplane approximation and related topics. In: Pach, J. (ed.) New Trends in Discrete and Computational Geometry, pp. 135–162. Springer, New York (1993)
4. Schobel, A.: Locating Lines and Hyperplanes. In: Theory and Algorithms, 1st edn. Kluwer Academic Publisher, Dordrecht (1999)
5. Brimberg, J., Juel, H., Schobel, A.: Linear facility location in three dimensions-Models and solution methods. Operations Research 50, 1050–1057 (2002)
6. Blanquero, R., Carrizosa, E., Schobel, A., Scholz, D.: A global optimization procedure for the location of a median line in the three-dimensional space. European Journal of Operational Research 215(1), 14–20 (2011)
7. Tizhoosh, H.R.: Opposition-based learning: A new scheme for machine intelligence. In: Proc. Int. Conf. Comput. Intell. Modeling Control and Autom., Vienna, Austria, vol. I, pp. 695–701 (2005)
8. Shokri, M., Tizhoosh, H.R., Kamel, M.: Opposition-based Q(λ) algorithm. In: Proc. IEEE World congress on Comput. Intell., Vancouver, BC, Canada, pp. 646–653 (2006)
9. Drezner, Z., Klamroth, K., Schobel, A., Wesolowsky, G.: The Weber Problem. In: Location Theory-Applications and Theory, pp. 1–36. Springer (2001)
10. Rao, R.V., Savsani, V.J., Vakharia, D.P.: Teaching-Learning-based optimization: A novel method for constrained mechanical design optimization problems. Computer-Aided Design 43, 303–315 (2011)
11. Karaboga, D.: An idea based on honey bee swarm for numerical optimization, Technical Report-TR06, Erciyes University (2005)

Modified Onlooker Phase in Artificial Bee Colony Algorithm

Tarun Kumar Sharma[1], Millie Pant[1], and V.P. Singh[2]

[1] Indian Institute of Technology Roorkee, India
[2] SCET, Saharanpur
{taruniitr1,millidma,singhvp3}@gmail.com

Abstract. Artificial bee colony (ABC) algorithm is relatively a new bio-inspired swarm intelligence optimization technique comparative to other population based algorithms. In this study BGA (breeder GA) mutation is embedded into onlooker bee phase to improve the capability of local search. The proposed variant is named B-ABC. The experimental results on 10 constrained benchmark functions demonstrate the performance of the proposed variant against those of state-of-the-art algorithms for a set of constrained test problems. Further the efficiency of the proposed variant is tested on the car side impact problem.

Keywords: Artificial Bee Colony, Convergence, Optimization, Evolutionary Algorithm, BGA.

1 Introduction

Evolutionary Programming (EP) [1], Genetic Algorithms (GA) [2], Particle Swarm Optimization (PSO) [3], Differential Evolution (DE) [4], Ant colony optimization (ACO) [5], and the like, are some algorithms to solve complex real world problems that include engineering, biology, finance, etc. PSO and ACO, algorithms are inspired by the collective intelligent behavior of some species such as school of fishes, colonies of ants, and swarm behavior. Researchers modeled this swarm behavior and engineers tested the efficiency and competitiveness of these designed models on complex design problems.

In the present study, the focus is on ABC, which is a recently proposed algorithm introduced by Karaboga in 2005 [6]-[7]. ABC follows the analogy of the socio-cooperative behavior demonstrated by honey bees in their search for nectar. A brief outline of ABC algorithm is given in Section 2.

Karaboga and Basturk have compared the performance of the ABC algorithm with the performance of other well-known modern heuristic algorithms such as genetic algorithm (GA), differential evolution (DE), particle swarm optimization on unconstrained and constrained problems [8]-[9]. A comprehensive survey on ABC can be found in [10].

In this paper, we modified onlooker phase to improve the local search capability of the food sources by embedding the BGA mutation in the artificial bee colony algorithm for solving constrained benchmark and further implemented on car side impact

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 339–347, 2012.

problem taken from the literature. B-ABC algorithm showed a comparatively good performance.

The remaining of the paper is organized as follow. Section 3 presents the motivation and the proposed B-ABC, a variant of ABC. The proposed variant is tested on car side impact problem that is discussed in section 4. The experimental settings and simulated results are analysed in section 5. Finally the conclusion is given in section 6.

2 Artificial Bee Colony: An Outline

The artificial bee colony classifies the foraging artificial bees into three groups of bees: employed bees, onlookers and scouts. The first half of the colony consists of the employed artificial bees and the second half includes the onlookers. For every food source, there is only one employed bee. In other words, the number of employed bees is equal to the number of food sources around the hive. The employed bee whose food source has been exhausted by the bees becomes a scout.

Step-by-step procedure of ABC is given as follows:.

1. Randomly distributed food sources are generated over a D-dimensional search space.
2. An artificial onlooker bee chooses a food source depending on the probability value associated with that food source, P_i, calculated by the expression:

$$P_i = \frac{fit_i}{\sum_{i=1}^{SN} fit_i} \tag{1}$$

where fit_i is the fitness value of the solution i which is proportional to the nectar amount of the food source in the position i and SN is the number of food sources which is equal to the number of employed bees. In order to produce a candidate food position from the old one in memory, the ABC uses the following expression:

$$V_{ij} = x_{ij} + r_{ij}(x_{ij} - x_{kj}) \tag{2}$$

where k$\in$ {1, 2, …, SN} and j$\in$ {1, 2, …, D} are randomly chosen indexes. Further, $k \neq i$ and r_{ij} is a random number between [-1, 1].

3. In ABC algorithm, "*limit*" is an important control parameter, it controls the times of updates of a certain solution. If a solution cannot be improved further through a predetermined number of cycles called limit then that solution is assumed to be abandoned, and the employed bee becomes a scout. If the solution is x_i and $j\in$ {1, 2, …, D}to be abandoned, then a new solution produced randomly would replace x_i by:

$$x_{ij} = x_{\min,j} + rand(0,1)(x_{\max,j} - x_{\min,j}) \tag{3}$$

where $x_{\min}^{j}$ is the lower bound of the parameter j and $x_{\max}^{j}$ is the upper bound of the parameter j. However for solving the constrained real parameter optimization problems a modified version of ABC employs Deb's rule [11] is proposed in [9]. In order to produce a candidate food position from the old one in memory, the ABC algorithm uses the following expression:

$$V_{ij} = \begin{cases} x_{ij} + \phi_{ij}(x_{ij} - x_{kj}), & if \ R_j \le MR \\ x_{ij} \ , & otherwise \end{cases} \tag{4}$$

where k∈ {1, 2,..., SN} is randomly chosen index. Although k is determined randomly, it has to be different from *i*. R_j is randomly chosen real number in the range [0,1] and j∈ {1, 2,...,D}. MR (modification rate) is a control parameter that controls whether the parameter x_{ij} will be modified or not.

3 Proposed Variant: B-ABC

The basic structure of ABC is such that it is good at exploration while poor at exploitation [12]. Further it was observed in [12] that according to the solution search equation of ABC algorithm described by Eq. (2), the new candidate solution is generated by moving the old solution towards (or away from) another solution selected randomly from the population. However, the probability that the randomly selected solution is a good solution is the same as that the randomly selected solution is a bad one, so the new candidate solution is not promising to be a solution better than the previous one. On the other hand, in Eq. (2), the coefficient r_{ij} is a uniform random number in [-1, 1] and x_{kj} is a random individual in the population, therefore, the solution search dominated by Eq. (2) is random enough for exploration.

As a conclusion for the success of any nature inspired algorithm the two antagonist factors exploration and exploitation must be balanced. So in order to improve the performance of basic ABC, BGA (breeder GA) mutation proposed by Mühlenbein and Schlierkamp-Voosen [13] is embedded into the structure of ABC. The goal of the BGA mutation is to improve the capability of local search. Moreover, the dynamic setting of $rang_i$ makes the population favor exploration at the early stage and exploitation at the later stage of the search.

- BGA mutation

Suppose $x' = (x_1, x_2,\ldots, x_n)$ is a chromosome and x_i is a variable to be mutated. A new value x'_i is computed according to:

$$v'_{ij} = x_{ij} \pm rang_i \cdot \alpha \tag{5}$$

where $rang_i$ defines the mutation range and it is normally set to $0.1.(u_i - l_i)$, the + or - sign is chosen with a flip probability of 0.5. α is set by the following expression

$$\alpha = \sum_{k=0}^{15} \alpha_k 2^{-k} \tag{6}$$

where $\alpha_k \in [0, 1]$. Before BGA mutation is applied, each α_k equal to 0, and then each αk is mutated to 1 with probability p ($\alpha_k = 1$) = 1/16. Only $\alpha_k = 1$ contributes to the sum. On average, there will be just one α_k with value 1, say α_j. Then α is fixed to 2^{-j}.

Hence the onlooker bee phase in B-ABC is modified by using the following equation

$$v_{ij} = \begin{cases} x_{ij} \pm rang_i \cdot \alpha & if \ (u(0,1) \le 0.5) \\ x_{ij} \end{cases} \tag{7}$$

4 Car Side Impact Problem

Car side-impact problem is used as a benchmark problem and is taken from [14]. For the safety measures of the passenger's cars have to pass the standard tests. The effect of side impact, according to several measuring criteria's, is measured on a dummy. European Enhanced Vehicle Safety Committee (EEVC) has led the foundation for the side impact. The velocity of the front area (object) makes direct impact on the dummy and can be reduced by increasing the car parameters dimension. This, of course, will add weight to the car and will the fuel consumption and increase the material costs. Thus here the objective is to minimize the weight. The minimization problem is mathematically formulated as:

$$Minimize \quad f(x) = Weight \tag{8}$$

subject to:

$$g_1(x) = F_a(load\ in\ abdomen) \le 1\ kN\ ;\ g_2(x) = VC_u(dummy\ \ upper\ \ chest) \le 0:32\ m/s$$
$$g_3(x) = VC_m(dummy\ \ \ middle\ \ chest) \le 0:32\ m/s\ \ g_4(x) = VC_l(dummy\ \ lower\ \ chest) \le 0:32\ m/s$$
$$g_5(x) = \Delta_{ur}(upper\ rib\ deflection) \le 32\ mm\ \ g_6(x) = \Delta_{mr}(middle\ rib\ deflection) \le 32\ mm$$
$$g_7(x) = \Delta_k(lower\ rib\ deflection) \le 32\ mm\ \ g_8(x) = F_p(Public\ force) \le 4\ kN$$
$$g_9(x) = V_{MBP}(Velocity\ of\ V-Pillar\ at\ middle\ po\,\text{int}) \le 9:9\ mm/ms$$
$$g_{10}(x) = V_{FD}(Velocity\ of\ \ front\ door\ at\ V-Pillar) \le 15:7\ mm/ms$$

Structural weight and response to impact can be approximated using global response surface methodology in order to simplify the analytical formulation of the optimization problem and speed up computations. The simplified models are defined as follows [15]:

$$Weight = 1.98 + 4.90x_1 + 6.67x_2 + 6.98x_3 + 4.01x_4 + 1.78x_5 + 2.73x_7 \tag{9}$$

$$F_a = 1.16 - 0.3717x_2x_4 - 0.00931x_2x_{10} - 0.484x_3x_9 + 0.01343x_6x_{10} \tag{10}$$

$$VC_u = 0.261 - 0.0159x_1x_2 - 0.188x_1x_8 - 0.019x_2x_7 + 0.0144x_3x_5 + 0.0008757x_5x_{10} + 0.08045x_6x_9 + 0.00139x_8x_{11} + 0.00001575x_{10}x_{11} \tag{11}$$

$$VC_m = 0.214 + 0.00817x_5 - 0.131x_1x_8 - 0.0704x_1x_9 + 0.03099x_2x_6 - 0.018x_2x_7 + 0.0208x_3x_8 + 0.121x_3x_9 - 0.00364x_5x_6 + 0.0007715x_5x_{10} - 0.0005354x_6x_{10} + 0.00121x_8x_{11} \tag{12}$$

$$VC_l = 0.74 - 0.061x_2 - 0.163x_3x_8 + 0.001232x_3x_{10} - 0.166x_7x_9 + 0.227x_2^2 \tag{13}$$

$$\Delta_{ur} = 28.98 + 3.818x_3 - 4.2x_1x_2 + 0.0207x_5x_{10} + 6.63x_6x_9 - 7.7x_7x_8 + 0.32x_9x_{10} \tag{14}$$

$$\Delta_{mr} = 33.86 + 2.95x_3 + 0.1792x_{10} - 5.057x_1x_2 - 11.0x_2x_8 - 0.0215x_5x_{10} - 9.98x_7x_8 + 22.0x_8x_9 \tag{15}$$

$$\Delta_{lr} = 46.36 - 9.9x_2 - 12.9x_1x_8 + 0.1107x_3x_{10} \tag{16}$$

$$F_p = 4.72 - 0.5x_4 - 0.19x_2x_3 - 0.0122x_4x_{10} + 0.009325x_6x_{10} + 0.000191x_{11}^2 \tag{17}$$

$$V_{MBP} = 10.58 - 0.674x_1x_2 - 1.95x_2x_8 + 0.02054x_3x_{10} - 0.0198x_4x_{10} + 0.028x_6x_{10} \quad (18)$$

$$V_{FD} = 16.45 - 0.489x_3x_7 - 0.843x_5x_6 + 0.0432x_9x_{10} - 0.0556x_9x_{11} - 0.000786x_{11}^2 \quad (19)$$

The simple bounds of this problem are: $0.5 \leq x_1, x_3, x_4 \leq 1.5$; $0.45 \leq x_2 \leq 1.35$; $0.875 \leq x_5 \leq 2.625$; $0.4 \leq x_6, x_7 \leq 1.2$; $x_8, x_9 \in \{0.192, 0.345\}$; $0.5 \leq x_{10}, x_{11} \leq 1.5$.

5 Experimental Settings, Simulated Results and Discussions

Two experiments were performed in order to determine the performance of the B-ABC with respect to the original ABC version and comparison with respect to state-of-the-art approaches. 10 well-known benchmark problems were taken from the literature [16] and is further implemented to solve car side impact problem.

For simulating ABC and B-ABC algorithms the control parameters, colony size (SN) of 40 food sources is taken, modification rate (MR), maximum cycle numbers and limit are fixed to 0.4, 5000 and MCN/2*SN respectively. All the experiments were run 25 times. And for optimizing the car side impact problem the *SN* is taken as 20 with 1000 number of iterations.

All the algorithms have been executed on dual core processor with 1GB RAM. The programming language used is DEV C++. The random numbers are generated using inbuilt rand () function with same seed for every algorithm.

Result analysis of constrained benchmark functions:

The simulated results obtained by B-ABC of 10 constrained benchmark problems are presented in Table 1(a) and 1(b). The best results are highlighted in bold. From the results it is noted that B-ABC reached the best known or optimum feasible solution in 5 problems: g01, g03, g04, g0, and g08. B-ABC was able to find close results to global optimum on five of rest functions (g02, g05, g07, g09, g10). Comparative results of the best and mean solutions of the state-of-art algorithms are also presented in Table 1(a) & (b). From the table according to best results, it can be seen that B-ABC performs better than [17] - [22] and ABC show equal performances; [19] ISR, demonstrate better performance than B-ABC.

However, in terms of the mean results presented B-ABC is performs better than [17], [18], [20]-[22] and ABC algorithms. While [19] perform better than B-ABC with respect to the mean results, ABC and OPA show similar performances according to the mean results. It means that B-ABC is more robust than [17], [18], [20] -[22]. [19] is better than B-ABC in both cases.

Result analysis of car side impact problem:

The comparative simulated results of car side impact problem are summarized in Table 2. The results of the problem are compared with the results obtained and taken from literature [23], using other meta-heuristic techniques such as PSO, GA, DE and ABC.

For the fair comparison, we tried to maintain the uniform settings, since implementation details may affect algorithm performance in terms of computation time. From the table it can be observed that the B-ABC a proposed variant of ABC in this study outperformed GA and DE was slightly better than PSO and FA. Both, PSO and B-ABC shows a smaller standard deviation on optimized weight.

Table 1. The best & (mean) solutions obtained by the HM, SR, ISR, OPA, ASCHEA, SMES, GA, DE, PSO and ABC algorithms for (a) (*g01 - g05*) (b) (*g06 - g10*) test functions over 25 independent runs. (–) Means that no feasible solutions were found

(a)

Algorithm	**g01**	**g02**	**g03**	**g04**	**g05**
Optimal	−15.000	0.803619	1.000	−30665.539	5126.498
	(−15.000)	(0.803619)	(1.000)	(−30665.539)	(5126.498)
HM[1] [17]	−14.7864	0.79953	0.9997	−30664.5	–
	(−14.7082)	(**0.79671**)	(0.9989)	(−30655.3)	(–)
SR[2] [20]	−15.000	0.803515	1.000	−30665.539	5126.497
	(**−15.000**)	(0.781975)	(1.000)	(**−30665.539**)	(5128.881)
ISR[2] [19]	−15.000	0.803619	1.001	−30665.539	5126.497
	(**−15.000**)	(0.782725)	(1.001)	(**−30665.539**)	(**5126.497**)
OPA[3] [19]	−15.000	0.803619	0.747	−30665.539	5126.497
	(**−15.000**)	(0.776283)	(0.257)	(**−30665.539**)	(5268.610)
ASCHEA[4] [18]	−15.0	0.785	1.0	−30665.5	5126.5
	(−14.84)	(0.59)	(0.99989)	(−30665.5)	(5141.65)
GA[5] [21]	14.440	0.796231	0.990	−30626.053	–
	(−14.236)	(0.788588)	(0.976)	(−30590.455)	(–)
SMES[5] [21]	−15.000	0.803601	1.000	−30665.539	5126.599
	(**−15.000**)	(0.785238)	(1.000)	(**−30665.539**)	(5174.492)
PSO[5] [22]	15.000	0.669158	0.993930	−30665.539	5126.484
	(−14.710)	(0.419960)	(0.764813)	(**−30665.539**)	(5135.973)
DE[5]	−15.000	0.472	1.000	−30665.539	5126.484
	(−14.555)	(0.665)	(1.000)	(**−30665.539**)	(5264.270)
ABC[5]	−15.000	0.803598	1.000	−30665.539	5126.484
	(**−15.000**)	(0.792412)	(1.000)	(**−30665.539**)	(5185.714)
B-ABC	-15.000	0.802781	1.000	-30665.539	5126.498
(Present Study)	(**-15.000**)	(0.79802)	(**1.000**)	(**-30665.539**)	(5126.487)

(b)

Algorithm	**g06**	**g07**	**g08**	**g09**	**g10**
Optimal	−6961.814	24.306	0.095825	680.63	7049.25
	(−6961.814)	(24.306)	(0.095825)	(680.63)	(7049.25)
HM[1] [17]	−6952.1	24.620	0.0958250	680.91	7147.9
	(−6342.6)	(24.826)	(0.0891568)	(681.16)	(8163.6)
SR[2] [20]	−6961.814	24.307	0.095825	680.630	7054.316
	(−6875.940)	(24.374)	(**0.095825**)	(680.656)	(7559.192)
ISR[2] [19]	−6961.814	24.306	0.095825	680.630	7049.248
	(**−6961.814**)	(**24.306**)	(**0.095825**)	(**680.630**)	(7049.250)
OPA[3] [19]	−6961.814	24.306	0.095825	680.630	7049.248
	(**−6961.814**)	(24.307)	(**0.095825**)	(**680.630**)	(**7049.248**)
ASCHEA[4] [18]	−6961.81	24.3323	0.095825	680.630	7061.13
	(−6961.81)	(24.6636)	(**0.095825**)	(680.641)	(7497.434)

Table 1. (*continued*)

GA[5] [21]	−6952.472	31.097	0.095825	685.994	9079.770
	(−6872.204)	(34.980)	(0.095799)	(692.064)	(10003.225)
SMES[5] [21]	−6961.814	24.327	0.095825	680.632	7051.903
	(−6961.284)	(24.475)	(**0.095825**)	(680.643)	(7253.047)
PSO[5] [22]	−6161.814	24.370153	0.095825	680.630	7049.381
	(**−6961.814**)	(32.407)	(**0.095825**)	(680.630)	(7205.5)
DE[5]	−6954.434	24.306	0.095825	680.630	7049.248
	(–)	(24.310)	(**0.095825**)	(**680.630**)	(7147.334)
ABC[5]	−6961.814	24.330	0.095825	680.634	7053.904
	(−6961.813)	(24.473)	(**0.095825**)	(680.640)	(7224.407)
B-ABC	-6961.814	24.312	0.095825	680.63	7050.591
(Present Study)	(**-6961.814**)	(24.309)	(**0.095825**)	(680.631)	(7049.467)

[1]Using decoder-based; [2]Using stochastic ranking; [3]Using over-penalty approach; [4]Using penalty-based; [5]Using Deb's method.

Table 2. Optimization results for the car side impact design problem

Algorithms	**PSO**	**DE**	**GA**	**FA**	**ABC**	**B-ABC**
Best objective	22.84474	22.84298	22.85653	22.84298	22.84839	22.84311
x_1	0.50000	0.50000	0.50005	0.50000	0.5	0.5000
x_2	1.11670	1.11670	1.28017	1.36000	1.183	1.11672
x_3	0.50000	0.5000	0.50001	0.50000	0.50001	0.5000
x_4	1.30208	1.30208	1.03302	1.20200	1.202	1.30209
x_5	0.50000	0.50000	0.50001	0.50000	0.5001	0.5000
x_6	1.50000	1.50000	0.50000	1.12000	1.12	1.50000
x_7	0.50000	0.50000	0.50000	0.50000	0.5000	0.50000
x_8	0.34500	0.34500	0.34994	0.34500	0.34491	0.34501
x_9	0.19200	0.19200	0.19200	0.19200	0.192	0.19200
x_{10}	-19.54935	-19.54935	10.3119	8.87307	8.87295	12.833
x_{11}	-0.00431	-0.00431	0.00167	-18.99808	-18.99749	-1.0827
Mean objective	22.89429	23.22828	23.51585	22.89376	22.8857	22.8449
Worst objective	23.21354	24.12606	26.240578	24.06623	24.8193	23.6738
S.D.	0.15017	0.34451	0.66555	0.16667	0.17393	0.15073

6 Conclusions

In the present study we modified the onlooker's phase to improve local search capability by embedding BGA mutation in the structure of basic ABC. The performance of B-ABC algorithm is tested and compared with the results of basic ABC and state-of-art algorithm on 10 well-known constrained optimization benchmark problems. It is shown that our modification of ABC algorithm for constrained optimization problems can handle tested functions very well. Further the efficiency of the proposed

algorithm is implemented on car side impact problem that also justifies the performance of B-ABC. This indicates the practical usage of the variant since many real word problems are constrained problems.

References

1. Fogel, L.J., Owens, A.J., Walsh, M.J.: Artificial intelligence through a simulation of evolution. In: Maxfield, M., Callahan, Fogel, L.J. (eds.) Biophysics and Cybernetic Systems, Proceeding of the 2nd Cybernetic Sciences Symposium, pp. 131–155 (1965)
2. Goldberg, D.: Genetic Algorithms in Search Optimization and Machine Learning. Addison Wesley Publishing Company, Reading (1986)
3. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proceeding of IEEE International Conference on Neural Networks, Perth, Australia, pp. 1942–1948. IEEE Service Center, Piscataway (1995)
4. Price, K., Storn, R.: Differential Evolution – a Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces, Technical Report, International Computer Science Institute, Berkley (1995)
5. Dorigo, M., Maniezzo, V., Colorni, A.: Positive feedback as a search strategy, Technical Report 91-016, Politecnico di Milano, Italy (1991)
6. Karaboga, D.: An Idea based on Bee Swarm for Numerical Optimization, Technical Report, TR-06, Erciyes University Engineering Faculty, Computer Engineering Department (2005)
7. Karaboga, D., Basturk, B.: A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) algorithm. Journal of Global Optimization 39, 459–471 (2007)
8. Karaboga, D., Basturk, B.: On the performance of artificial bee colony (ABC) algorithm. Applied Soft Computing 8, 687–697 (2008)
9. Karaboga, D., Basturk, B.: Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems. In: Melin, P., Castillo, O., Aguilar, L.T., Kacprzyk, J., Pedrycz, W. (eds.) IFSA 2007. LNCS (LNAI), vol. 4529, pp. 789–798. Springer, Heidelberg (2007)
10. Karaboga, D., Gorkemli, B., Ozturk, C., Karaboga, N.: A comprehensive survey: artificial bee colony (ABC) algorithm and applications. Artif. Intell. Rev. (2011), doi:10.1007/s10462-012-9328-0
11. Deb, K.: An efficient constraint handling method for genetic algorithms. Comput. Methods Appl. Mech. Engrg. 186, 311–338 (2000)
12. Zhu, G.P., Kwong, S.: Gbest-guided artificial bee colony algorithm for numerical function optimization. Applied Mathematics and Computation (2010), doi:10.1016/j.amc.2010.08.049
13. Mühlenbein, H., Schlierkamp-Voosen, D.: Predictive models for the breeder genetic algorithm I: Continuous parameter optimization. Evol. Comput. 1(1), 25–49 (1993)
14. Gu, L., Yang, R.J., Cho, C.H., Makowski, M., Faruque, M., Li, Y.: Optimization and robustness for crashworthiness. Int. J. Vehicle Des. 26(4), 348–360 (2001)
15. Youn, B.D., Choi, K.K.: A new response surface methodology for reliability-based design optimization. Comput. Struct. 82, 241–256 (2004)
16. Liang, J.J., Runarsson, T.P., Mezura-Montes, E., Clerc, M., Suganthan, P.N., Coello Coello, C.A., Deb, K.: Problem Definitions and Evaluation Criteria for the CEC 2006 Special Session on Constrained Real-Parameter Optimization, Technical Report (September 2006), http://www.lania.mx/~emezura/documentos/tr_cec06.pdf

17. Koziel, S., Michalewicz, Z.: Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. Evolutionary Computation 7(1), 19–44 (1999)
18. Hamida, S.B., Schoenauer, M.: ASCHEA: new results using adaptive segregational constraint handling. In: Proceedings of the Congress on Evolutionary Computation 2002 (CEC 2002), vol. 1, pp. 884–889. IEEE Service Center, Piscataway (2002)
19. Runarsson, T.P., Yao, X.: Search biases in constrained evolutionary optimization. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 35(2), 233–243 (2005)
20. Runarsson, T.P., Yao, X.: Stochastic ranking for constrained evolutionary optimization. IEEE Transactions on Evolutionary Computation 4(3), 284–294 (2000)
21. Mezura-Montes, E., Coello Coello, C.A.: A simple multimembered evolution strategy to solve constrained optimization problems. Technical Report EVOCINV-04–2003
22. Muñoz-Zavala, A., Hernández, A.A., Diharce, E.R.V.: Constrained optimization via particle evolutionary swarm optimization algorithm (PESO). In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2005), New York, vol. 1, pp. 209–216. ACM Press, Washington, DC (2005) ISBN 1–59593-010–8
23. Gandomi, A.H., Yang, X.-S., Alavi, A.H.: Mixed variable structural optimization using Firefly Algorithm. Computers and Structures 89, 2325–2336 (2011)

Complex-Valued Neuro-Fuzzy Inference System Based Classifier

Kartick Subramanian[1], Ramaswamy Savitha[1], Sundaram Suresh[1], and B.S. Mahanand[2]

[1] School of Computer Engineering, Nanyang Technological University, Singapore
kartick1@e.ntu.edu.sg, {savi0001,ssundaram}@ntu.edu.sg
[2] Dept. of Information Science and Engineering, Sri Jayachamarajendra College of Engineering, Mysore, India
bsmahanand@sjce.ac.in

Abstract. In this paper, we propose a complex-valued Takagi-Sugeno-Kang type-0 neuro-fuzzy inference system (CNFIS) and develop for it, a gradient-descent based learning algorithm to solve classification problems. The gradient-descent based learning algorithm is derived based on Wirtinger calculus: which preserves the amplitude-phase correlation. The performance of the developed algorithm is evaluated on a set of four binary classification problems and three multi-category classification problems. Comparison with various real-valued and complex-valued classifiers show the improved performance of CNFIS.

Keywords: Complex-valued neuro-fuzzy system, wirtinger calculus, classification.

1 Introduction

Neuro-fuzzy inference systems [25, 28], which combine the optimal solution formulation ability of neural networks with the interpretability of fuzzy inference systems are increasingly being used to solve various problems in the domain of control, medicine, telecommunication, etc. Some of these problems, especially in the field of telecommunication [11] and image processing [13], operate on complex-valued signals, inherently. In addition, certain tasks such as wind speed and direction prediction [12] and tree representation of hand in hand gesture recognition system [7] employ complex-valued signals for better representability and solution formulation. The use of real-valued networks for these problems, would annul the advantages of complex-valued features [3]. Hence in literature, complex-valued networks have been proposed which operate on complex-valued signals. These networks also have better computational power than real-valued neural networks and better performance on real-valued classification tasks [14]. A brief review on existing complex-valued classifiers is provided in Section 2.

In order to combine the advantages of neuro-fuzzy inference system and complex-valued signal processing in solving classification tasks, we propose a complex-valued neuro-fuzzy inference system (CNFIS) based classifier. The proposed CNFIS is a four layered network which realizes Takagi-Sugeno-Kang type-0 fuzzy inference mechanism; with a real-valued Gaussian membership function. Although complex-valued neural networks employing real-valued Gaussian activation functions are available in

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 348–355, 2012.

literature [5], they do not use fully complex-valued gradients in their learning algorithm. Instead, they use the real and imaginary part of error to update the real and imaginary part of the network parameters, respectively, resulting in poor amplitude-phase correlation. Hence for CNFIS, we develop a gradient-descent based learning algorithm employing fully complex-valued gradients derived using Wirtinger Calculus [31]. In [24], such a CNFIS has been developed to forecast wind speed and direction, which employs mean squared error loss function. However, for classification tasks, it has been shown that hinge-loss functions are better error-loss function than mean squared error function [32, 29], as it could approximate the posterior probability more accurately. Hence in the current work, we employ hinge-loss function as error function and derive gradient descent based learning algorithm for classification tasks. The developed classifier is employed for solving four binary and three multi-category benchmark classification problems from UCI machine learning repository [4].

Rest of the paper is organized as follows: Section 2 presents a review on existing works on complex-valued neural network. Section 3 presents the architecture of CNFIS and develop its gradient-descent based learning algorithm for solving classification tasks derived using Wirtinger calculus. The performance of the proposed algorithm is evaluated in Section 4 and the paper concluded in Section 5.

2 A Brief Review on Complex-Valued Classifiers

Two major issues facing complex-valued neural networks are the selection of activation function and loss function so as to approximate amplitude and phase of the network accurately.

One of the basic requirements of artificial neural networks is the selection of an activation function which is non-constant, and entire and bounded everywhere. Whereas Lioville's theorem [15] states that an entire and bounded function is a constant function in complex domain. In the literature, two dominant approaches have been employed to overcome this impasse: (1) Split complex-valued neural networks and (2) Fully complex-valued neural networks.

Split complex-valued neural networks employ real-valued activation function with either real-valued weights [10, 8] or complex-valued weights [5, 11]. Networks with real-valued weights treat the complex-valued signals as pairs of independent real-valued signals and employ real-valued networks to process signals. While the networks with complex-valued weights map the complex-valued signals to real-valued feature space and is mapped back to complex-valued output space, by the virtue of output weights. Moreover, these algorithms rely on Cauchy Reimann conditions to update real and imaginary parts of parameters with real and imaginary parts of error, respectively. This will result in loss of cross-correlation between amplitude and phase during inference as well as learning. One of the first approaches to solve real-valued classification problems employing split complex-valued neural networks is [1]. In [1], the authors employ multi-valued neurons and use a derivative free global error correcting learning rule to update the network parameters. While in [2] a phase encoded complex-valued neural network has been proposed to perform classification tasks. This network uses a gradient-descent based batch learning algorithm. However, this networks do not employ fully complex-valued gradients in its learning.

Another approach to circumvent the restriction imposed by Lioville's theorem was proposed by Kim and Adali [9]. They relaxed the criterion for a fully complex-valued activation function to be analytic and bounded *almost everywhere* in complex plane. In [18, 22, 20, 27], a fully complex-valued Gaussian like *sech* activation function has been proposed. The use of this activation function has helped the network exploit the advantages of the orthogonal decision boundaries of complex-valued neural networks to provide better generalization performance. This activation function has also been employed successfully in extreme learning machines [21, 17]. In [21], two extreme learning machine classifiers, employing phase encoded as well as bilinear branch-cut complex-valued features, have been proposed. While in [17], a circular transformation with a translational/rotational bias that performs unique transformation of real-valued features to complex plane is employed.

Second issue related to complex-valued neural networks is the selection of appropriate loss function in order to approximate amplitude and phase of the network accurately. Most of the complex-valued neural networks found in literature employs well-known mean square error function as the loss function. However, this loss function does not include the error in phase directly and hence, does not approximate phase accurately [23]. In [23, 26, 19], authors have employed logarithmic error function, that directly minimizes both the errors in magnitude and phase.

Next, we shall propose a complex-valued neuro-fuzzy inference system.

3 Network Architecture and Learning Algorithm of CNFIS

Suppose we have N observations $\{\mathbf{x}^t, c^t\}_{t=1}^{N}$ where, $\mathbf{x}^t \in \mathbb{R}^m$ and $c^t \in \{1, 2, \cdots, n\}$ are the real-valued input and class labels of the network, respectively and n is the number of distinct classes.

Real-valued features are mapped to all the four quadrants of complex domain using a circular transformation as described in [17]. In addition, the real-valued class label c^t is converted to coded class label in complex domain $(\mathbf{y}^t \in \mathbb{C}^n)$ as

$$y_j^t = \begin{cases} 1 + 1i & if \quad c^t = j \\ -1 - 1i & otherwise \end{cases} \tag{1}$$

The aim of the CNFIS is to find the functional relationship between the inputs and outputs, so as to approximate the decision surface accurately. Next, we shall describe the inference mechanism of CNFIS and present its learning algorithm.

3.1 Network Architecture

The architecture of CNFIS is presented in Fig. 1. A detailed description of each layer is presented herein.

– Input layer: This layer is a linear layer with m nodes. There are as many nodes as the number of features for the given problem. The output of the l^{th} input node is given as

$$u_l = x_l, \;\; l \;=\; 1, \; 2, \; \cdots, \; m \tag{2}$$

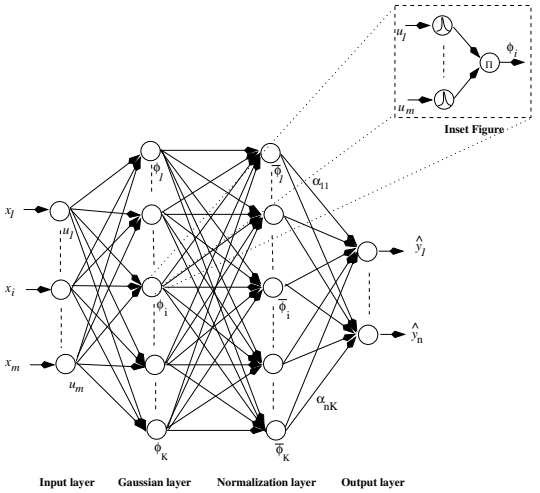


Fig. 1. Architecture of TSK type-0 complex-valued neuro-fuzzy inference system. The inset figure describes the internal structure of ith Gaussian rule antecedent.

– Gaussian layer: This layer consists of K nodes, where each node represents a hidden rule antecedent in the network. Response of the rule layer is real-valued due to the use of Gaussian membership function. The firing strength of k^{th} rule is given by

$$\phi_k(\mathbf{u}) = \exp\left(-(\mathbf{u}-\boldsymbol{\mu}_k)^H(\Sigma_{kk}^H\Sigma_{kk})^{-1}(\mathbf{u}-\boldsymbol{\mu}_k)\right), \;\; k \;=\; 1,\; 2,\; \cdots,\; K \quad (3)$$

where, $\mathbf{u}$ is the complex-valued input feature, $\boldsymbol{\mu}_k$ is the center of the Gaussian rule antecedent and Σ_{kk} is the width of the Gaussian rule.

– Normalization layer: This layer consists of as many nodes as the Gaussian layer. The output of the k^{th} normalization node is

$$\bar{\phi}_k = \frac{\phi_k}{\sum_{l=1}^{K}\phi_l}, \;\; k \;=\; 1,\; 2,\; \cdots,\; K \quad (4)$$

– Output layer: The output layer is a linear layer with n nodes representing output features. The predicted output is given by

$$\hat{y}_j = \sum_{k=1}^{K}\alpha_{jk}\bar{\phi}_k \quad j \;=\; 1, 2, \cdots, n \quad (5)$$

Here, α_{jk} is the complex-valued rule consequent between the k^{th} nomalization layer node and j^{th} output layer node.

Next, we shall describe the complex-valued gradient descent based learning algorithm for CNFIS.

3.2 Learning Algorithm

Let $\hat{\mathbf{y}}^t \in \mathbb{C}^n$ be the predicted output of the t^{th} input sample. The sum of squares error is calculated as

$$E = \frac{1}{2} \sum_t \left(\mathbf{e}^{tH} \mathbf{e}^t \right); \tag{6}$$

where H represents hermitian of a complex number and $\mathbf{e}$ is the hinge-loss error [29] which can be given as

$$e_j^t = \begin{cases} y_j^t - \hat{y}_j^t & \text{if} \quad y_j^t \hat{y}_j^t > 1 \\ 0 & \text{Otherwise} \end{cases} \tag{7}$$

The superscript t is dropped in rest of the discussion for convenience sake.

The gradient descent based learning algorithm proceeds by updating the parameters of the network depending on the gradient of error with respect to the parameters. The update rule for output weight, rule antecedent center and width, derived using Wirtinger calculus is provided herein. For detailed derivation, one should refer to [24].

Output Weight Update: The output weight of k^{th} rule, $\boldsymbol{\alpha}_k$ is updated according to the equation

$$\boldsymbol{\alpha}_k = \boldsymbol{\alpha}_k + \frac{1}{2} \cdot \eta_\alpha \cdot \mathbf{e} \cdot \bar{\phi}_k^{\,T} \tag{8}$$

where η_α is the predefined learning rate. T denotes transpose of the vector.

Rule Antecedent Center Update: The center of the j^{th} rule, $\boldsymbol{\mu}_j$ is updated according to the equation

$$\boldsymbol{\mu}_k = \boldsymbol{\mu}_k + \frac{1}{2} \cdot \eta_\mu \cdot (\boldsymbol{\alpha}_k^H \cdot \mathbf{e} + \boldsymbol{\alpha}_k \cdot \mathbf{e}^H) \cdot \bar{\phi}_k \cdot (1 - \bar{\phi}_k) \cdot (\Sigma_{kk}^H \Sigma_{kk})^{-1} \cdot (\mathbf{u} - \boldsymbol{\mu}_k) \tag{9}$$

Here, η_μ is the learning rate for rule center.

Rule Antecedent Width Update: The width of the k^{th} rule, Σ_{kk} is updated by

$$\Sigma_{kk}^* = \Sigma_{kk} + \eta_\Sigma \cdot (\boldsymbol{\alpha}_k^H \cdot \mathbf{e} + \boldsymbol{\alpha}_k \cdot \mathbf{e}^H) \cdot \bar{\phi}_k \cdot (1 - \bar{\phi}_k) \cdot (\mathbf{u} - \boldsymbol{\mu}_k) \cdot (\Sigma_{kk}^H \Sigma_{kk})^{-2} \cdot \Sigma_{kk} \tag{10}$$

Here, η_Σ is the learning rate for rule width.

Next, we shall compare the performance of the developed CNFIS classifier on benchmark classification problems.

4 Performance Analysis

For performance comparison of CNFIS, four binary and three multi-category benchmark data sets from UCI machine learning repository [4] are chosen. Details regarding the problems considered, including the number of features, classes, the number of training and testing samples is provided in Table 1. The table also provides the imbalance factor which is a measure of class-wise sample density. It is defined as

$$I.F. = 1 - \frac{n}{N} \min_{j=1,2,\cdots,n} n_j \tag{11}$$

Table 1. Binary and Multi-category data sets considered

Data set	# features	# classes	# Samples employed		I.F	
			training	testing	training	testing
Liver	6	2	200	145	0.17	0.14
ION	34	2	100	251	0.28	0.28
BC	9	2	300	383	0.26	0.33
PIMA	8	2	400	368	0.22	0.39
IS	19	7	210	2100	0	0
VC	18	4	424	422	0.1	0.12
GI	9	6	336	105	0.72	0.77

where, n is the number of classes, N is the total number of samples and $\mathrm{n_j}$ is the number of samples in class j.

The performance of the proposed algorithm is compared against other real-valued as well as complex-valued classifiers and neuro-fuzzy inference systems available in literature such as, Support Vector Machines (SVMs) [6], Extreme Learning Machines [30], Sequential adaptive fuzzy inference system (SAFIS) [16] and Meta-cognitive neuro-fuzzy inference system [25] in real domain and fully complex-valued radial basis function network (FC-RBF) in complex domain. Performance is compared with respect to the number of rules and overall testing efficiency. Overall testing efficiency is

$$\eta_o = 100 \times \frac{1}{N} \sum_{j=1}^{n} q_{jj} \tag{12}$$

where, q_{jj} is the number of correctly classified samples in class j.

Table 2. Performance Comparison of benchmark classification problems

Data set	SVM		ELM		SAFIS		McFIS		FC-RBF		CNFIS	
	K	η_o	K	η_o	K	η_o	K	η_o	K	η_o	K	η_o
Liver	158	68.24	132	71.79	133	68.28	66	71.7	20	74.6	20	76.2
ION	30	90.18	25	88.78	46	91.63	23	94.2	10	89.48	8	92.03
BC	190	94.20	65	96.28	55	97.39	33	97.39	10	97.12	8	97.12
PIMA	209	76.43	218	76.54	179	71.46	133	75.81	20	78.53	25	77.99
IS	127	91.38	49	90.23	156	90.38	78	91.33	38	92.33	35	92.9
VC	340	70.62	150	77.01	307	69.19	-	-	70	77.01	70	73.7
GI	183	70.47	80	81.31	134	75.24	100	82.9	90	83.76	70	81.76

From the Table 2, it could be noticed that CNFIS outperforms other real-valued networks including SVM, ELM, SAFIS and McFIS for almost all the problems considered. For certain problems, McFIS outperforms CNFIS due to the meta-cognitive nature of the learning algorithm. However, the complex-valued networks employ significantly lesser number of rules to achieve considerably better performance. This shows that the

use of complex-valued features as well as the use of learning algorithm which preserve the amplitude-phase correlation has helped the network achieve better performance.

5 Conclusion

In this paper, we have proposed a complex-valued neuro-fuzzy inference system and developed for it a gradient-descent based learning algorithm for solving classification problems. The learning algorithm employs fully complex-valued derivative derived employing Wirtinger calculus and uses hinge-loss error function. The performance of the developed algorithm is evaluated on a set of four binary classification problems and three multi-category classification problems. Comparison with various real-valued and complex-valued classifier shows the improved performance of the network.

References

[1] Aizenberg, I., Paliy, D., Zurada, J., Astola, J.: Multilayer feedforward neural network based on multi-valued neurons (mlmvn) and a backpropagation learning algorithm. Soft Computing 11(2), 169–183 (2007)
[2] Amin, M., Murase, K.: Single-layered complex-valued neural network for real-valued classification problems. Neurocomputing 72(4-6), 945–955 (2009)
[3] Benvenuto, N., Marchesi, M., Piazza, F., Uncini, A.: A comparison between real and complex-valued neural networks in communication applications. In: Proceedings of Intl. Conf. on Neural Networks, pp. 1771–1775 (1991)
[4] Blake, C., Merz, C.: UCI repository of machine learning databases. Department of Information and Computer Sciences. University of California, Irvine (1998)
[5] Chen, S., McLaughlin, S., Mulgrew, N.: Complex valued radial basis function network, part I: Network architecture and learning algorithms. EURASIP Singal Processing Journal 35(1), 19–31 (1994)
[6] Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(3), 272–297 (1995)
[7] Hafiz, A.R., Amin, M.F., Murase, K.: Real-Time Hand Gesture Recognition Using Complex-Valued Neural Network (CVNN). In: Lu, B.-L., Zhang, L., Kwok, J. (eds.) ICONIP 2011, Part I. LNCS, vol. 7062, pp. 541–549. Springer, Heidelberg (2011)
[8] Hirose, A.: Continuous complex-valued back-bropagation learning. Electronic Letters 28(20), 1854–1855 (1992)
[9] Kim, T., Adali, T.: Fully complex multilayer perceptron network for non-linear signal processing. Journal of VLSI, Signal Processing 32(1-2), 29–43 (2002)
[10] Leung, H., Haykin, S.: Complex backpropagation algorithm. IEEE T. Signal Processing 39(9), 2101–2104 (1991)
[11] Li, M., Huang, G., Saratchandran, P., Sundararajan, N.: Complex valued growing and pruning rbf neural networks for communication channel equalization. IEEE Proc. Vision, Image and Signal Processing 153(4), 411–418 (2006)
[12] Mandic, D., Javidi, S., Goh, S., Kuh, A., Aihara, K.: Complex valued prediction of wind profile using augmented complex statistics. Renewable Energy 34(1), 196–201 (2009)
[13] Muezzinoglu, M., Guzelis, C., Zuruda, J.: A new design method for the complex values multistate hopfield associative memory. IEEE Trans. on Neural Networks 14(4), 891–899 (2003)

[14] Nitta, T.: The Computational Power of Complex-Valued Neuron. In: Kaynak, O., Alpaydın, E., Oja, E., Xu, L. (eds.) ICANN/ICONIP 2003. LNCS, vol. 2714, pp. 993–1000. Springer, Heidelberg (2003)
[15] Remmert, R.: Theory of complex functions. Springer, Berlin (1991)
[16] Rong, H., Sundararajan, N., Huang, G., Saratchandran, P.: Sequential adaptive fuzzy inference system SAFIS for nonlinear system identification and prediction. Fuzzy Sets and Syst. 157(9), 1260–1275 (2006)
[17] Savitha, R., Suresh, S., Sundararajan, N.: Fast learning circular complex-valued extreme learning machine (CC-ELM) for real-valued classification problems. Information Sciences 187(1), 277–290 (2012)
[18] Savitha, R., Suresh, S., Sundararajan, N.: Fast learning complex-valued classifiers for real-valued classification problems. Intl. J. Machine Learning and Cyber. (accepted, 2012)
[19] Savitha, R., Suresh, S., Sundararajan, N.: A meta-cognitive learning algorithm for a fully complex-valued relaxation network. Neural Networks 32(1), 209–218 (2012)
[20] Savitha, R., Suresh, S., Sundararajan, N.: Metacognitive learning in a fully complex-valued radial basis function neural network. Neural Computation 24(5), 1297–1328 (2012)
[21] Savitha, R., Suresh, S., Sundararajan, N., Kim, H.J.: Fast Learning Fully Complex-Valued Classifiers for Real-Valued Classification Problems. In: Liu, D., Zhang, H., Polycarpou, M., Alippi, C., He, H. (eds.) ISNN 2011, Part I. LNCS, vol. 6675, pp. 602–609. Springer, Heidelberg (2011)
[22] Savitha, R., Suresh, S., Sundararajan, N., Kim, H.J.: A fully complex valued radial basis function classifier for real valued classification problems. Neurocomputing 78(1), 104–110 (2012)
[23] Savitha, R., Suresh, S., Sundararajan, N., Saratchandran, P.: A new learning algorithm with logarithmic performance index for complex-valued neural networks. Neurocomputing 72(16-18), 3771–3781 (2009)
[24] Subramanian, K., Savitha, R., Suresh, S.: Complex-valued neuro-fuzzy inference system for wind prediction. In: World Congress on Computational Intelligence, pp. 3217–3223 (2012)
[25] Subramanian, K., Suresh, S.: A meta-cognitive sequential learning algorithm for neuro-fuzzy inference system. Applied Soft Computing 12(11), 3603–3614 (2012)
[26] Suresh, S., Savitha, R., Sundararajan, N.: A fast learning fully complex-valued relaxation network (fcrn). In: IEEE Intl. Joint Conf. on Neural Networks, pp. 1372–1377 (2011)
[27] Suresh, S., Savitha, R., Sundararajan, N.: A sequential learning algorithm for complex-valued self-regulating resource allocation network-CSRAN. IEEE Transactions on Neural Networks 22(7), 1061–1072 (2011)
[28] Suresh, S., Subramanian, K.: A sequential learning algorithm for meta-cognitive neuro-fuzzy inference system for classification problems. In: Intl. Joint Conf. Neural Networks, pp. 2507–2512 (2011)
[29] Suresh, S., Sundararajan, N., Saratchandran, P.: Risk-sensitive loss functions for sparse multi-category classification problems. Information Sciences 178, 2621–2638 (2008)
[30] Suresh, S., Venkatesh Banu, R., Kim, H.J.: No-reference image quality assessment using modified extreme learning machine classifier. Applied Soft Computing 9(2), 541–552 (2009)
[31] Wirtinger, W.: Zur formalen theorie der funktionen von mehr komplexen veränderlichen. Annals of Mathematics 97, 357–375 (1927)
[32] Zhang, T.: Statistical behavior and consistency of classification methods based on convex risk minimization. Annals of Statistics 32(1), 56–85 (2003)

Neural Network Based Model Predictive Controller for Simplified Heave Model of an Unmanned Helicopter

Mahendra Kumar Samal, Sreenatha Anavatti, and Matthew Garratt

University of New South Wales, Australian Defence Force Academy,
Northcott Drive, Canberra ACT-2600, Australia
{m.samal,s.anavatti,m.garratt}@adfa.edu.au
http://www.unsw.adfa.edu.au

Abstract. Neural network (NN) based model predictive controller (NN-MPC) for height control of an unmanned helicopter is presented in this paper. The applicability of the NN-MPC scheme is evaluated on a simplified heave model of the helicopter in simulation. NN based system identification (NNID) technique is used to model the heave dynamics of the unmanned helicopter which is then used in the MPC algorithm to estimate the future control moves. To show the efficacy of the controller, controller results are provided. Results indicate that NN-MPC scheme is capable of handling external disturbances and parameter variations of the system.

Keywords: Neural Network, Model Predictive Controller, Unmanned Helicopter, Heave Dynamics.

1 Introduction

Rotary-wing Unmanned Aerial Vehicles (RUAVs) are a class of unmanned or pilot-less aerial vehicles which have aerodynamic and propulsion characteristics that enable it to hover, take-off and land vertically. This special feature of the RUAVs to hover, take-off and land on rough terrain to carry-out mission critical operations without endangering the life of human pilot has an added edge over the unmanned fixed-wing aircraft which require a runway for take-off and landing. RUAVs can be successfully used for real-time reconnaissance, surveillance, search and rescue missions, weather data collection, bush fire monitoring, agricultural crop dusting and different airborne operations [1–3].

Due to their aerodynamic and propulsion characteristics, RUAVs are highly complex and nonlinear systems. They are inherently unstable and quite responsive to small perturbations. In addition, their operating range and condition changes frequently with the changes in flight modes i.e. hover flight, cruise flight, forward flight, sideward flight and rearward flight. This necessitate an adaptive nonlinear controller.

System identification is the process of determining the input output relationship of a dynamic system from experimental data. Different conventional and nonconventional system identification techniques are available in literature. Conventional techniques such as Maximum Likelihood Estimation Method, Modified Maximum Likelihood Estimation Method, Kalman Filtering methods [4–7] have been used for flight vehicle system identification. These conventional methods of system identification require the

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 356–363, 2012.

structure of the mathematical model to be known a priori. Often, it may not be possible to come up with a structured mathematical model of a highly coupled nonlinear system like an RUAV. Hence non-conventional methods like Fuzzy Logic and Neural Network (NN), which do not explicitly require a mathematical model of a system, have gained popularity in recent times [8, 9].

Nonlinear Model Predictive Controller (NMPC) is a class of modern control technique which makes use of nonlinear process model to estimate the future control command to achieve optimum control performance. The nonlinear model is used to predict the plant behaviour over a specified period of time. Neural Network Model Predictive Controller (NN-MPC) uses NN model of the helicopter to compute the future control moves subject to input and output constraints. The present work deals with the implementation of NN-MPC algorithm for autonomous hover of a simplified heave model of the helicopter.

The Eagle autonomous helicopter platform shown in Fig. 1 is under development at UNSW at ADFA with an objective to develop AFCS for fully autonomous flight. The platform is equipped with avionics built in-house and instrumented with different sensors for measurement, processing and control. A simulation model of RUAV is developed to test and verify different identification and control algorithm and for conducting closed-loop experiments before real-flight implementation.

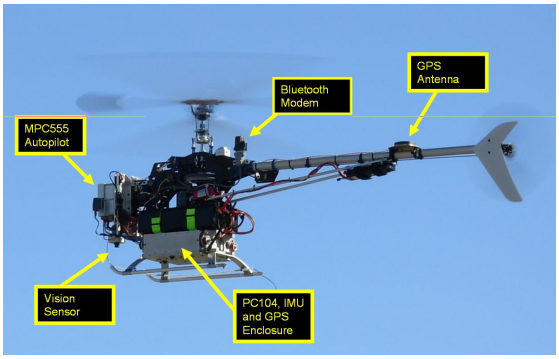


Fig. 1. Eagle helicopter

The rest of the paper is organised as follows. Section 2 briefly describes the simplified heave model of the helicopter platform. The controller performance with offline NN model is evaluated in Section 3. In Section 4, the controller performance with online NN model is analysed. The effect of parameter variations on the performance of the NN-MPC controller is studied in Section 5. The paper is concluded in Section 6.

2 Control: Heave Dynamics

The simplified Simulink® model of the helicopter used for validating the NN-MPC controller is shown in Fig. 2. In this simplified model, the dynamics corresponding to the collective input is isolated from longitudinal, lateral, flybar and tail rotor dynamics. Collective control channel is used with the rotor induced flow model to calculate the net acceleration generated by the helicopter.

This vertical acceleration (a_z) is integrated to obtain vertical velocity (V_z) which in turn is again integrated to obtain vertical height of the helicopter. The thrust (T) acting on the helicopter is calculated using the aerodynamics of the helicopter. Vertical

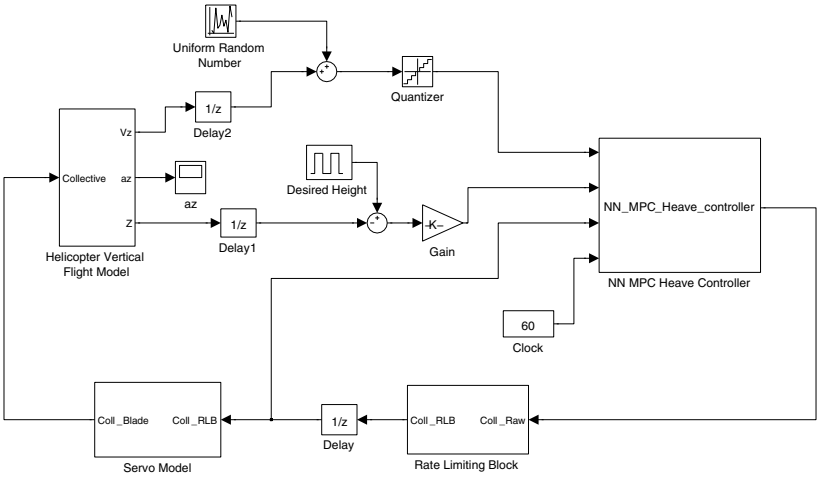


Fig. 2. Simplified helicopter model with NN-MPC controller

acceleration is obtained as the ratio of the total thrust acting on the helicopter to the helicopter weight M ($a_z = \frac{T}{M}$). The aerodynamic and physical parameters used for designing the simplified heave model of the helicopter are as follows: Total Weight is 8.2Kg; Main Rotor (MR) Radius is 0.76m; MR blade chord is 0.058m; MR Lift Curve Slope is 5.7; Main Rotor RPM is 1600.

The input to the heave dynamics is the collective (δ_{coll}). This input is generated by the NN-MPC controller based on V_z and Z. Thus the input and outputs which are of interest for NN-MPC algorithm are δ_{coll}, V_z and Z. The controller block (NN MPC Heave Controller) in Fig. 2 is implemented in Matlab® as a Simulink® "C MEX" S-function. The optimisation problem in NN-MPC algorithm is solved as sequential quadratic programming (SQP) problem and SQP algorithm inside the NN-MPC is executed once every sample time.

The control objective of this work is to: Study controller performance with offline and online generated NN models; Study controller performance with various levels of noise; Study controller performance with parameter (trim setting, effective lift curve slope and total weight) variations.

3 Controller Performance with Offline Model

The non-linear dynamics can be expressed by the AutoRegressive eXogenous inputs (ARX) model structure [10]

$$\begin{aligned} &y(t+1) + a_1 y(t) + \cdots + a_n y(t-n_y) \quad = b_1 u(t) + \cdots + b_m u(t-m_u) \\ &\text{or} \quad y(t+1) = f\left(\varphi, \theta\right) \end{aligned} \tag{1}$$

where the regressor vector is $\varphi(t) = [y(t-1), \cdots, y(t-n), \quad u(t-1), \cdots u(t-m)]^T$ and the parameter vector θ is given by $\theta = [a_1 \quad a_2 \cdots a_n \quad b_1 \quad b_2 \cdots b_m]^T$. This ARX

model can be approximated by Feed Forward Neural Network (FFNN) [11]. For controlling the collective channel, the dynamics is modeled with V_z and δ_{coll}. The regressor vector φ is given by $\varphi = [V_z(t-1), \cdots, V_z(t-n_y), \delta_{coll}(t-1), \cdots, \delta_{coll}(t-n_u)]$. Two past inputs ($n_u = 2$) and past outputs ($n_y = 2$) are used in the regressor vector. The input output data for NN training with NNID is obtained by simulating the heave model with a PID controller. A large training data set spanning over 1000sec is generated by using random step references ranging from 0.2sec to 5sec. The amplitude of the desired step is restricted to ensure that the vertical velocities of the helicopter is within $\pm 2m/s$. To enable the NN model learn the noisy dynamics in the presence of sensor noise and wind gust, external additive disturbance is included during simulation for data generation. This additive disturbance is in the form of white noise with an intensity of 0.3m/sec in V_z. This large set of data is then used for training the neural network. The network has 6 hidden neurons including a bias element. Once the network is trained, the weights are frozen and later used in the NN-MPC controller for predicting future response of the system over prediction horizon.

The desired velocity is calculated from the error between the actual height (Z_{act}) and the commanded height (Z_{ref}). This desired velocity is obtained by multiplying a suitable gain (K) with the error in the height. This gain K is taken as 0.75 for this study. This can be written as $V_{zref} = K(Z_{ref} - Z_{act})$. From the above equation, a desired vertical velocity V_{zref} is computed and used as an input for the NN-MPC controller.

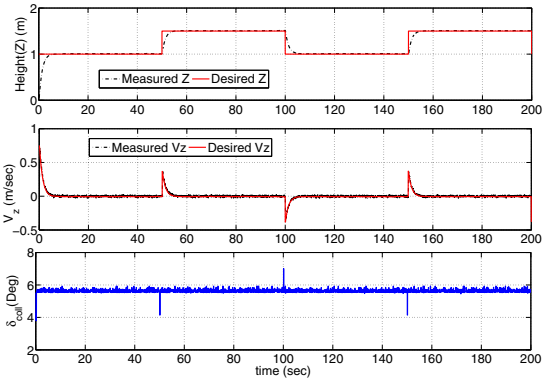


Fig. 3. Helicopter response: NN-MPC with offline model

Starting from the initial condition, the NN-MPC controller is required to drive the helicopter to track the repeating sequence stair of 1m and 1.5m heights. Fig. 3 shows the response of the helicopter with the NN-MPC controller while tracking the desired trajectory in the presence of small disturbance (white noise with an intensity of 0.005m/sec) in V_z. It can be seen that the helicopter tracks the commanded height very well. The controller is further tested with higher noise levels of intensity 0.1m/sec and 0.3m/sec. The same offline model is used for model prediction in NN-MPC controller. The responses of the helicopter are shown in Fig. 4. In the figure, Measured Z(a), Measured Z(b) and Measured Z(c) indicate the responses corresponding to the noise level of 0.005m/sec, 0.1m/sec and 0.3m/sec respectively. As can be seen from the Fig. 4, there is a deterioration in the performance as the level of noise increases. However, the helicopter is able to hover within $\pm$20cm height, even in the presence of sever noise, indicating that the performance of the helicopter with the NN-MPC is acceptable.

4 Controller Performance with Online Model

An offline generated model may not be able to represent the dynamics of the helicopter over entire flight regime due to its time varying dynamics. In such cases, an adaptive model of the helicopter is more suitable for predicting the dynamic behaviour over the prediction horizon. The adequacy of the NN-MPC algorithm in conjunction with the online identification scheme for tracking the reference trajectory is analyzed here. The NN model is generated every sample time using the most recent system information (measured input output data). At the first sample time, the NN weights are initialized to the offline trained weights. In subsequent sample times, the network training starts with weights corresponding to the previous sample time.

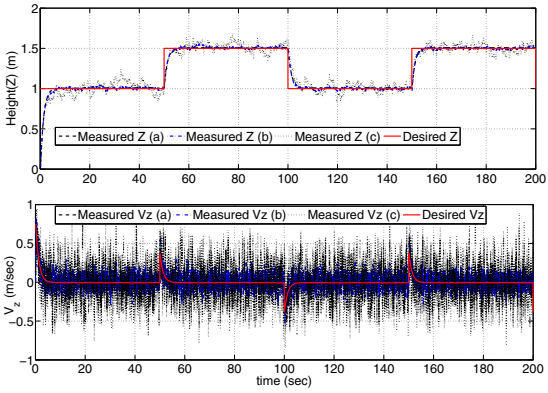


Fig. 4. Helicopter response in the presence of external disturbances: NN-MPC with offline model

The response of the helicopter with the online model in NN-MPC is shown in Fig. 5. A small additive disturbance (noise of intensity 0.005m/sec) is considered for this simulation. It can be seen that the helicopter hovers at the commanded height very well.

Fig. 6 shows the responses of the helicopter for higher levels of noise in V_z. It can be seen from the figure that there is a maximum absolute error of $\pm$16cm.

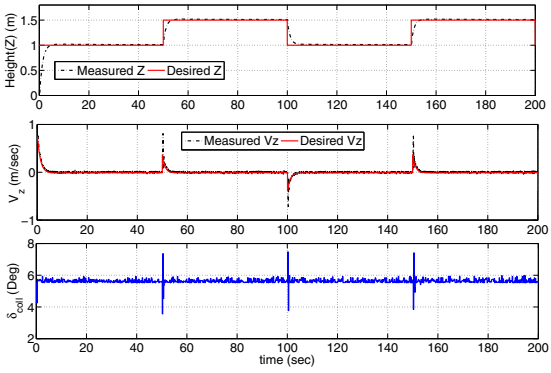


Fig. 5. Helicopter response: NN-MPC with online model

Next, the effect of parameter variations on the performance of the NN-MPC controller is studied. The performance of NN-MPC controller with offline trained model in the presence of parameter variations is compared against the performance of NN-MPC controller with online model of the plant. The offline NN model is generated for nominal plant without any parameter variations.

5 Controller Performance with Parameter Variation

Aerodynamics as well as physical parameters such as trim setting, effective lift curve slope (ELCS) and total weight change during flight. The change in the total weight is due to the fuel burn-off during flight or changes in the payload of the helicopter. A constant wind (head wind or tail wind) can have the effects, same as change in the trim setting during the flight. Due to the changes in flight conditions and environmental conditions, the net lift and drag acting on the helicopter rotor changes. These can be equivalently treated as change in the lift curve slope of the helicopter blade during flight. The controller is required to perform satisfactorily with these parameter variations. In this study, trim setting, ELCS and total weight of the helicopter are changed one at a time and the simulations are carried out.

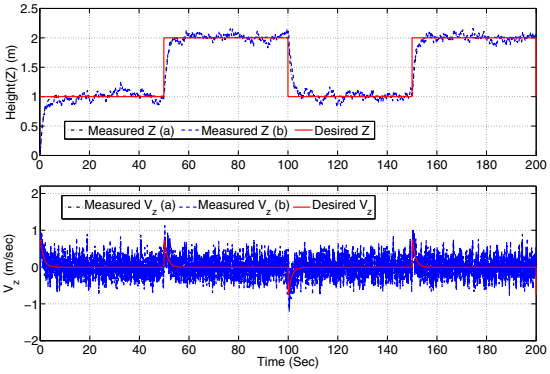


Fig. 6. Heave model in the presence of external disturbances: NN-MPC with online model (a) noise of 0.05m/sec and (b) noise of 0.3m/sec

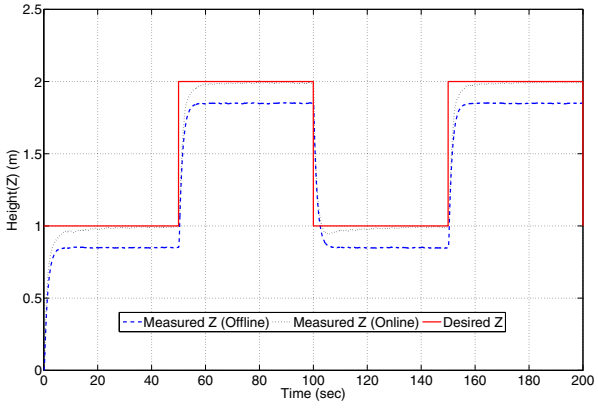


Fig. 7. NN-MPC with offline and online model (trim setting is changed from 5.52 to 6.52 degree)

Firstly, the trim setting is changed from 5.52 to 6.52 (about 18%) and the helicopter is commanded to track a repeating stair of 1m and 2m with a period of 100sec. The response of the plant with NN-MPC controller is presented in Fig. 7. As can be seen, NN-MPC controller with online model is able to give better system performance compared to controller with offline model. A steady state error of 16cm is noticed when

the offline trained model is used in controller. There is no noticeable steady state error when the online NN model is used. Thus the NN-MPC controller with online model performs better compared to the one with offline model and is capable of tracking the commanded altitude when there is a significant change in the trim setting.

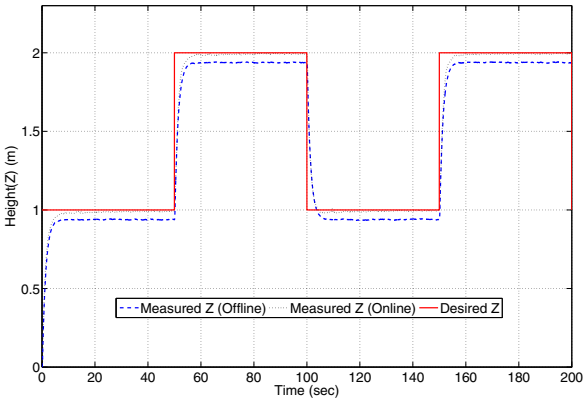


Fig. 8. NN-MPC with offline and online trained model (ELCS is changed from 5.7 to 6.7)

Secondly, the ELCS is changed from 5.7 to 6.7 (about 18%) and the helicopter is simulated with the changed ELCS. The helicopter responses with NN-MPC controller with offline as well as online model are plotted in Fig. 8. In this case also, it can be seen that the NN-MPC controller with online model outperforms the one with offline model. The steady state error of approximately 6cm with the offline model is eliminated with the use of online model.

Next, the performance of the NN-MPC controller is evaluated with variation in the total weight of the RUAV. For this, the total weight of the helicopter is changed from

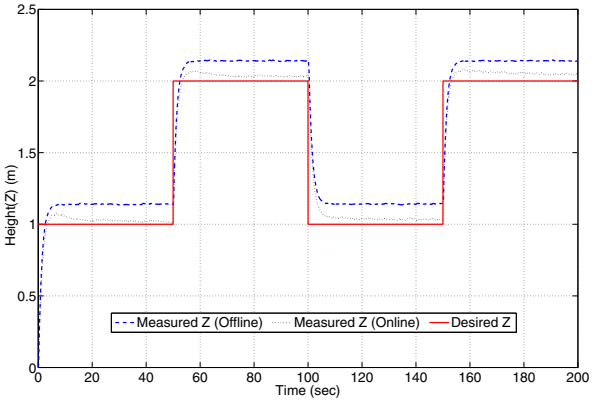


Fig. 9. NN-MPC with offline and online trained model (weight changed from 8.2kg to 10.2kg)

8.2kg to 10.2kg (approximately 25%). The helicopter with the new weight is driven by the NN-MPC controller to track the desired altitude. From Fig. 9, it can be seen that the controller with online model ensures that the desired altitude is closely matched by eliminating the steady state error seen with the offline model.

6 Conclusions

In the above section, the applicability and adequacy of the NN-MPC controller for heave model of the helicopter is validated for variety of cases in simulations. The robustness of the controller in the presence of noise and parameter variations is shown to be better with online model. NN-MPC controller with online model outperforms the controller with offline model and hence is suitable for control applications to handle variations in aerodynamic and physical parameters of the helicopter.

References

1. Sarris, Z.: Survey of UAV applications in civil markets. Technical report, STN ATLAS-3Sigma AE and Technical University of Crete, DPEM, 73100 Chania, Crete, Greece
2. http://www.uavm.com/uavapplications.html
3. Ollero, A., Maza, I.: Multiple Heterogeneous Unmanned Aerial Vehicles. STAR, vol. 37. Springer, Heidelberg (2007)
4. Mettler, B., Tischler, M.B., Kanade, T.: System identification modelling of a small-scale unmanned rotorcraft for flight control design. Journal of the American Helicopter Society, 50–63 (January 2002)
5. Jategaonkar, R., Fischenberg, D., von Gruenhagen, W.: Aerodynamic modeling and system identification from flight data-recent applications at dlr. Journal of Aircraft 41(4), 681–691 (2004)
6. Chowdhary, G., Jategaonkar, R.: Aerodynamic parameter estimation from flight data applying extended and unscented kalman filter. In: AIAA Atmospheric Flight Mechanics Conference (August 2006)
7. Kallapur, A., Ali, S., Anavatti, S.: Experiences using EMId and EKF for UAV online identification. In: Third International Conference on Autonomous Robots and Agents (ICARA 2006), Palmerston North, New Zealand, pp. 207–212. Springer (December 2006)
8. Hunt, K.J., Sbarbaro, D., Zbikowski, R., Gawthrop, P.: Neural networks for control systems-a survey. Automatica 28(6), 1083–1112 (1992)
9. McLoone, S.: Neural network identification: A survey of gradient based methods. IEEE (1998)
10. Ljung, L.: System Identification: Theory for the user. Prentice-Hall, Inc., Englewood Cliffs (1987)
11. Narendra, K.S., Parthasarathy, K.: Identification and control of dynamic systems using neural network. IEEE Transactions on Neural Networks 1(1), 4–27 (1990)

Hybrid Biogeography Based Simultaneous Feature Selection and Prediction of N-Myristoylation Substrate Proteins Using Support Vector Machines and Random Forest Classifiers

Shameek Ghosh[1], Nayana Ramachandran[2], C. Venkateshwari[3], and V.K. Jayaraman[1,*]

[1] Evolutionary Computing and Image Processing Group, Centre for Development of Advanced Computing (CDAC), Pune, Maharashtra
[2] Persistent Systems Limited, 'Aryabhata - Pingala', 9A / 12, Erandawane, Pune 411004, India
[3] Satyabhama University, Chennai, India
{shameekg,jayaramanv}@cdac.in, nayana_ramachandran@persistent.co.in, venkateswari.ch@gmail.com

Abstract. Majority of proteins undergo important post-translational modifications (PTM) that may alter physical and chemical properties of the protein and mainly their functions. Laboratory processes of determining PTM sites in proteins are laborious and expensive. On the contrary, computational approaches are far swifter and economical; and the models for prediction of PTMs can be quite accurate too. Among the PTMs, Protein N- terminal N-myristoylation by myristoyl-CoA protein N-myristoyltransferase (NMT) is an important lipid anchor modification of eukaryotic and viral proteins; occurring in about 0.5% encoded NMT substrates. Reliable recognition of myristoylation capability from the substrate amino acid sequence is useful for proteomic functional annotation projects as also in building therapeutics targeting the NMT. Using computational techniques, prediction-based models can be developed and new functions of protein substrates can be identified.

In this study, we employ Biogeography based Optimization (BBO) for feature selection along with Support Vector Machines (SVM) and Random Forest for classification of N-myristoylation sequences. The simulations indicate that N-myristoylation sites can be identified with high accuracy using hybrid BBO wrappers in combination with weighted filter methods.

Keywords: Post-translational modifications (PTM), N-myristoylation, Biogeography based Optimization (BBO), Support Vector Machines (SVM), Random Forest classifier, Amino Acid Indices, dbPTM, SwissProt.

* Corresponding author.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 364–371, 2012.

1 Introduction

Most proteins undergo covalent modifications during or after assembly of the polypeptide chain. This ensures proper protein conformation or folding [1], or directs the newly formed protein to distinct cellular apparatus. Studies indicate that some other protein translation modifications (PTMs) may occur after folding and localization activities are completed [2]; thus influencing the biological or catalytic activity of the protein (like protein degradation). Due to high data complexity, traditional techniques like mass spectroscopy, chemical proteomics, may turn out to be very costly. Thus, profile computations with computational methods are definitely the key to enhance future research in PTM substrate characterization. Protein N-myristoylation refers to the co-translational or post-translational covalent attachment of myristate, a 14-carbon saturated fatty acid, to the N-terminal glycine of eukaryotic and viral proteins. Myristoylation by the myristoyl-CoA protein *N-myristoyltransferase* (*NMT*) is an important lipid anchor modification of eukaryotic and viral proteins [3]. *NMT* recognizes the sequence motif of suitable substrate proteins at the N-terminus and attaches the lipid moiety to the absolutely essential N-terminal glycine residue [3]. N-Myristoyl proteins include proteins involved in different signal transduction cascades, especially intracellular and those enabling rapid, flexible cell responses [4]. Studies reveal that myristoylation and membrane-binding are known to regulate the activity of kinases and influence *c-Src* stability [5]. It is also known to be a vital growth and regulation component in development of the leukocytic lineage [6]. Recent scientific studies also indicate that HIV protein *Nef* is preferentially myristoylated by recombinant human isozyme *NMT2*. Thus, selective inhibition of *NMT2* may be termed as a novel means of blocking HIV virulence [4] and chemical proteomics has indicated the scope of developing therapeutics for *NMT* regulation [7].

In this regard, we have used the in-depth studies of the amino acid sequence variability of substrate proteins (on binding site analyses in X-ray structures or 3D homology models for NMTs from various taxa), biochemical data extracted from the scientific literature; and employed a hybrid BBO-based filter wrapper algorithm with SVM and RF for feature selection and prediction. It is discovered that, at least within a complete substrate protein, the N-terminal 17 amino-acid residues experience different types of variability restrictions that reveal a physical property pattern [8]. Accordingly, we have considered the relevant properties from N-terminal 17 amino-acid residues, where three motif regions may be identified as follows: Region 1 (positions 1–6) fitting the binding pocket, region 2 (positions 7–10) interacting with the NMT's surface at the mouth of the catalytic cavity and region 3 (positions 11–17) comprising a hydrophilic linker.

2 Materials and Methods

2.1 Dataset Generation

dbPTM [9] is a database that compiles information on protein post-translational modifications (PTMs). The database includes all of the experimentally validated PTM

sites from Swiss-Prot, PhosphoELM and O-GLYCBASE. The unified non-redundant positive and negative datasets were downloaded from dbPTM. Additionally, An SRS query was carried out for obtaining N-myristoylated sequences from SwissProt KnowledgeBase [10] (UniProtKB/Swiss-Prot Release 2010_11 of 02-November-2010). The query result retrieved entries from different eukaryotic species. This served as the Positive dataset. For the Negative dataset, random sequences (with Glycine as the second residue) which do not undergo N-myristoylation were selected. As the first 17 positions are the motif regions for these proteins, only the first 17 amino acid residues were considered for the analysis. For both positive and negative sequences, final datasets (containing combined entries from the dbPTM and the SWISS-PROT SRS search) were constructed after removing the redundant entries. The final positive dataset had 144 sequences and the negative dataset had 81 sequences.

2.2 Input Features for Classifier

Amino Acid Indices. We have used AAIndex1 section of the Amino Acid Index Database [11]. Each amino acid is assigned a value based on the different physicochemical and biological properties. For each sequence in the positive and negative datasets, we mapped the AAIndex1 values and created a 17 x 16 feature matrix programmatically, using PERL v5.12. The rows consisted of 16 different AAIndex1 indices and columns contained the AAIndex1 values at each of the 17 positions. Thus, there were 17 x 16 = 272 features for every sequence.

Thus, the final dataset comprises of 273 x (144 + 81) for positive sequences and the negative sequences together.

3 Biogeography Based Optimization

The field of biogeography attempts to study the distribution and dynamics of various species geographically and tracks their evolution over time. In 2008, D. Simon [12] first built upon the operational aspects of BBO to mimic its natural behavior for engineering optimization. Recently, the applications of BBO to various combinatorial optimization problems have reported encouraging results [12-17]. In this study, we present an application of BBO for simultaneous feature selection and protein function prediction as described.

3.1 Weighted Heuristic Ranking

To bring greater variability in our algorithm and help explore the feature search space efficiently, we have applied some statistical filter algorithms like Infogain (IG), Chi-Square (CS) and Correlation based Feature Selection (CFS) estimators [18], from the Weka software library [19], to obtain four different ranking of the descriptors for the concerned datasets. These are then combined by adopting a weighted ranking approach where weights are separately assigned to each filter and a weighted sum is generated. Equation (1) indicates the weighted statistical measure we used as heuristic, for classification.

$$Desc_{wt} = w_{ig} * IG_{(Desc)} + w_{cs} * CS_{(Desc)} + w_{cfs} * CFS_{(Desc)} \quad (1)$$

3.2 Hybrid Biogeography Based Feature Selection

In a BBO, the population consists of candidate solutions, otherwise also known as habitats. A habitat's suitability is influenced by a set of SIVs (Suitability Index Variables). We represent a feature in the dataset as an SIV. Therefore, each habitat is composed of a set of features. The suitability of a habitat is consequently related to the fitness function of the solution, also known the habitat suitability index. While selecting informative features, a candidate feature subset thus represents a habitat and its HSI is determined by the corresponding reduced dataset's 10 fold cross validated classification accuracy obtained from SVM and RF classifiers [12, 17]. Accordingly, our objective is to improve the overall HSI (fitness) landscape of the population over a number of iterations (generations).To do this, BBO employs the use of an operator called migration which mimics the natural form of geographical migration. A simple linear model of immigration and emigration of species across various habitats is shown in Figure 1.

Thus, obtaining the number of species for each habitat is paramount for computing the migration rates of each habitat. In our model, we map the species such that the HSI is directly proportional to the number of species in a habitat. The species count is later used to compute individual emigration (μ) and immigration (λ) rates of the habitats/candidate solutions as shown in equations (2) and (3) respectively.

$$\lambda_k = I(1 - \frac{k}{n}) \tag{2}$$

$$\mu_k = \frac{E * k}{n} \tag{3}$$

Here k denotes the number of species in the k^{th} habitat and n is the maximum count of species considered by the model. E and I are maximum emigration and immigration rates, which are both equal to 1.

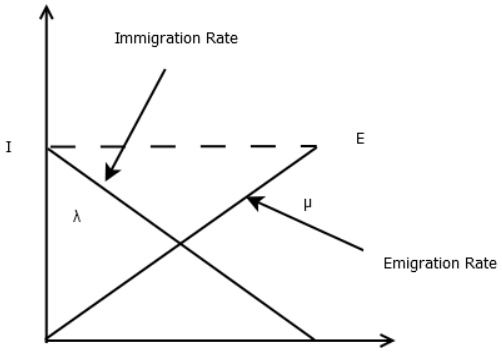


Fig. 1. Rate of Migration vs. Number of Species

As per BBO, good HSI solutions (with higher number of species) tend to have higher emigration rates and probabilistically share SIVs with poor HSI solutions which have higher immigration rates (refer Fig.1). For purposes of migration, a solution proportional to the immigration rate is selected and then correspondingly, a solution proportional to the emigration rate is selected too. Internally, for each of the emigrating habitat, which is selected, a random SIV is shared with the immigrating habitat. HSI of the modified habitats are recalculated after the process of migration ends. In addition, we perform mutations of probabilistically selected habitats to avoid problems due to local optima. During this process, we introduce the weighted ranking (refer equation (1)) of all the features as a heuristic to the BBO. The process of mutation iterates over all the habitats and based on their mutation probability, may be performed on a habitat. We set the mutation rate to 0.5 in order to allow sufficient exploration of the unvisited portions of the feature space with a good chance. BBO thus combines the weighted filter ranking (described in equation (1)) with the wrapper approach (i.e. BBO-SVM/RF), at this stage during mutation. A descriptor is thus selected with a probability proportional to the weighted ranking and is then used to replace an already existing SIV in the selected habitat.

Finally, elitism is implemented by carrying forward a certain number of best habitats of each iteration in the following stages. This way, BBO ensures that the best habitats are not corrupted in subsequent generations. This process is repeated across several generations until the best solution's HSI in the population starts giving consistent results.

3.3 Support Vector Machines

Based on statistical learning theory, Vapnik (1995) developed the class of algorithms currently known as Support Vector Machines (SVM) [20-21]. SVM makes use of a hyper-plane to divide a set of binary-labeled data, maximizing the margin between the nearest data points of each label, in the process. In case of linearly non-separable data, SVM transforms the input points into a higher dimensional feature space and then finds a suitable linear hyper-plane for separating the data points for classification. For implementation purposes, we have employed the libSVM software library [22].

3.4 Random Forests

Random Forests [23] comprise of a collection of randomly constructed decision trees. Random attributes are used for node splitting while growing a decision tree. The classification tree is constructed using an 'in bag' data (bootstrap set built using random sampling by replacement) by the CART algorithm. The overall accuracy of the Random Forest is assessed by the out-of-bag data (OOB-samples not selected for in-bag set). Thus, a statistical aggregation of the voted class label, which is not equal to the original class of a sample and averaged over all the cases in the training data, is called as the OOB error rate, which gives an estimate of the RF error rate. We have used the Weka Software suite for RF-specific implementation purposes [19].

4 Results and Discussion

Extensive simulations were carried out to maximize algorithmic performance by tuning the relevant algorithm parameters which include the weight parameters for each filter being used (IG, CS, CFS).An initial population of 100 has been considered to account for more variability in the migration process and for reaching an optimal result in lesser number of generations thereby minimizing the computational time too. Table 1 shows the values of various parameters used as part of BBO, SVM and RF.

Table 1. Algorithm Parameters

Algorithm Parameters	***Values***
cost(SVM), gamma(SVM),kernel	50, 0.02, radial basis function(rbf)
w_{ig}, w_{cs}, w_{cfs}	0.3, 0.3 , 0.4
trees(RF) , mtry(RF)	100, √features
Population, Generations, Elitism(BBO)	100,50,5

In Table 2, we list the 10 – fold cross validation accuracies obtained for the selected gene subsets using BBO. The BBO-SVM and BBO-RF algorithms reported the most optimal results for a subset size of 10 and 9 respectively when compared against the filter methods for the same feature subset sizes. The selected subsets were later used to build a model using 70% of the data (treated as a training set) and the subsequent training model was used to predict the rest 30% data(test set) for all the methods used. A test sensitivity comparison of BBO-SVM/RF against Infogain-SVM/RF is as shown in Table 3.

Table 2. 10-fold Cross Validation Classification Accuracies

SVM	**RF**	**Infogain-RF**	**Infogain-SVM**	**BBO-SVM**	**BBO-RF**
87.56%	92.4%	89.3%	89.7%	95.5%	97.7%

Table 3. Comparison of Infogain and BBO-SVM/RF sensitivities

Infogain-RF	**Infogain-SVM**	**BBO-SVM**	**BBO-RF**
0.92	0.92	1.0	0.96

So far, our results indicate that the weighted filter based BBO in combination with SVM and RF has performed well as a feature selection algorithm and improves upon, the results provided by filter based methods.

5 Conclusions

For this study, we employed a hybrid filter-wrapper approach employing weighted heuristics with BBO-SVM and BBO-RF algorithms for simultaneous feature selection

and subsequent model construction. The inclusion of weighted heuristics also provided more options for an effective search space exploration that seems to have helped in improvement of the overall population across generations. The BBO is also simple to implement and flexible based on the various possible alternatives in a problem domain and its related constraints. A significant speedup may also be achieved by parallel implementations where accuracies of individual solutions may be computed in parallel.

Acknowledgements. VKJ gratefully acknowledges the Council of Scientific and Industrial Research (CSIR) and Department of Science and Technology (DST), New Delhi, India for financial support.

References

1. Chou, P.Y., Fasman, G.D.: Empirical predictions of protein conformations. Annu. Rev. Biochem. 47, 251–276 (1978)
2. Frank, E., Birgit, E., Werner, K., Sebastian, M.-S., Georg, N., Georg, S., Michael, W.: Prediction of lipid posttranslational modifications and localization signals from protein sequences: big-Π, NMT and PTS1. Nucleic Acids Research 31, 3631–3634 (2003)
3. Gordon, J.I., Duronio, R.J., Rudnick, D.A., Adams, S.P., Gokel, G.W.: Protein N-Myristoylation. J. Biol. Chem. 266(14), 8647–8650 (1991)
4. Hayashi, N., Titani, K.: N-myristoylated proteins, key components in in-tracellular signal transduction systems enabling rapid and flexible cell responses. Proc. Jpn. Acad. Ser. B Phys. Biol. Sci. 86(5), 494–508 (2010)
5. Patwardhan, P., Resh, M.D.: Myristoylation and Membrane Binding Regulate c-Src Stability and Kinase Activity. J. Mol. Biol. 30(17), 4094–4107 (2010)
6. Kumar, S., Singh, B., Dimmock, J.R., Sharma, R.K.: N-myristoyltransferase in the leukocytic development processes. Cell Tissue Res. 345(2), 203–211 (2011)
7. Wright, M.H., Heal, W.P., Mann, D.J., Tate, E.W.: Protein myristoylation in health and disease. J. Chem. Biol. 3(1), 19–35 (2010)
8. Sebastian, M.-S., Birgit, E., Frank, E.: N-terminal N-Myristoylation of Proteins: Refinement of the Sequence Motif and its Taxon-specific Differences. J. Mol. Biol. 317, 523–540 (2002)
9. Lee, T.-Y., Huang, H.-D., Hung, J.-H., Huang, H.-Y., Yang, Y.-S., Wang, T.-H.: dbPTM: an information repository of protein post-translational modification. Nucleic Acids Res. 34 (Database issue), D622–D627 (2006)
10. Khoury, G.A., Baliban, R.C., Floudas, C.A.: Proteome-wide post-translational modification statistics: frequency analysis and curation of the swiss-prot database. Sci. Rep. 1, 90 (2011)
11. Kawashima, S., Pokarowski, P., Pokarowska, M., Kolinski, A., Katayama, T., Kanehisa, M.: AAindex: amino acid index database. Nucleic Acids Res. 36 (Database issue), D202–D205 (2008)
12. Simon, D.: Biogeography-Based Optimization. IEEE Transactions on Evolutionary Computation 12, 702–713 (2008)
13. Ma, H., Simon, D.: Blended biogeography-based optimization for constrained optimization. Engineering Applications of Artificial Intelligence 24(3), 517–525 (2011)
14. Mo, H., Xu, L.: Biogeography based optimization for traveling salesman problem. In: Sixth International Conference of Natural Computation, vol. 6, pp. 3143–3147 (2010)

15. Song, Y., Liu, M., Wang, Z.: Biogeography-based optimization for the traveling alesman problems. In: Third International Joint Conference on Computational Science and Optimization (CSO), vol. 1, pp. 295–299 (2010)
16. Panchal, V.K., Singh, P., Kaur, N., Harish, K.: Biogeography based satellite image classification. International Journal of Computer Science and Information Security 6, 269–274 (2009)
17. Nikumbh S., Ghosh S., Jayaraman V. K.: Biogeography-Based Informative Gene Selection and Cancer Classification Using SVM and Random Forests. In: IEEE World Congress on Computational Intelligence (IEEE WCCI 2012), Australia. In Press (2012)
18. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques - Information Gain. TheKaufmann Series in Data Management Systems. Morgan Kaufmann (2011)
19. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: An update. SIGKDD Explor. 11, 130–133 (2009)
20. Boser, B.E., Guyon, I.M., Vapnik, V.N.: Training algorithm for optimal margin classifiers. In: 5th Annual ACM Workshop on COLT, pp. 144–152. ACM Press, Pittsburgh (1992)
21. Cortes, C., Vapnik, V.N.: Support-Vector Networks. Machine Learning 20 (1995)
22. Chang, C.-C., Lin, C.-J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27 (2011)
23. Breiman, L.: Random forests. Machine Learning 45, 5–32 (2001)

Plant Leaf Disease Detection Using Gabor Wavelet Transform

Shitala Prasad[1], Piyush Kumar[2], Ranjay Hazra[1], and Ajay Kumar[3]

[1] Department of Electronics and Computer Engineering, IIT Roorkee, 247667, India
[2] Department of Information Technology, IIIT Allahabad, 211012, India
[3] Department of Electronics and Instrumentation, UCER Allahabad, 211010, India
godjidec@iitr.ernet.in,
{piyushkumariiita,raja.ranjay,kumar.ajay.style}@gmail.com

Abstract. This paper explores a new dimension of pattern recognition to detect crop diseases based on Gabor Wavelet Transform. The first proposed plant biometric system consist three modules: (1) spot detection using histogram based segmentation, (2) feature extraction using GWT and (3) feature matching with advance machine learning algorithm, SVM. The experimental results on different disease dataset shows that the GWT is effective and robust algorithm for plant disease detection. The accuracy is around 89% in all circumstances. The developed system is very helpful in biology and botanical studies and also used to guide and make aware the Indian farmers about the crop diseases and their natural and chemical controls to improve the production rate.

Keywords: Fourier Transform, Gabor Wavelet Transform, SVM, PCA, Plant Biometric System.

1 Introduction

Image processing and computer vision one of the future technologies and is becoming a ubiquitous part of human life today in all the fields with advance mobile devices. No fields are untouched by computer vision and one of the main focused area is agriculture. India being an agriculture country and maximum population depends on; require technological improvements to increase the production rate. Computer vision is playing a major role in this. Image processing algorithms such as edge detection, generalized Hough transform [1]; Marr-Hildreth algorithm [2]; scale-invariant transform [3]; and others are used for identification of plant leaves. In [4, 5] authors have used shape based plant leaf recognition to identify the plant species. Soille [6] used leaf vein information to identify plant species using mathematical morphological filters on digital images of plant leaves.

A relative sub-image based feature extraction was proposed by authors in [7] resulting an accuracy of 95% and in [8] Prasad el at., used Curvelet transform for the same purpose resulting a minor improvement of 95.36% but increases the computation and cost complexity. The advantage of using Curvelet is that it is

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 372–379, 2012.

multi-resolution and is multi-directional because of which over come all the shortcomings of other already proposed methods.

An automated plant biometric system is proposed in this paper to detect crop diseases from the day first it is attacked and to identify their proper name and chemical controls used for it. The plant leaves have patterns and are used by botanists to identify the species and disease it has. In similar manner, computer vision based pattern recognition is used in this paper to identify the leaf diseases such as bacterial, virus, and fungi diseases without any expert's presences. The plant biometric system is very much required for Indian farmers to know the status of crops day-to-day. And can be used for analytical studies by the botanical students performing their experiments. The system proposed uses simple Gabor Wavelet texture algorithm for disease identification.

This paper is divided into five sections. In the second section the overall plant biometric system is proposed and described followed by feature extraction algorithm, Gabor Wavelet transform. In the fourth section, the experimental result is shown and in the last section conclusion and future work.

2 Plant Biometric System

Plant leaves are the first where disease symptoms are seen and thus are used by the proposed system to identify the visual patterns and predict the disease. A healthy plant leaves are usually green but changes their color to light yellow, brown, red or mottled surface due to disease attack. All they have an uncommon complex pattern behind creating a wall for researchers to develop a single system to automatically detect all those diseases. A block flow diagram of proposed plant biometric system is shown in fig 1 with three main modules: spot detection, feature extraction and feature matching. Note that the system proposed here is to detect only plant leaf diseases.

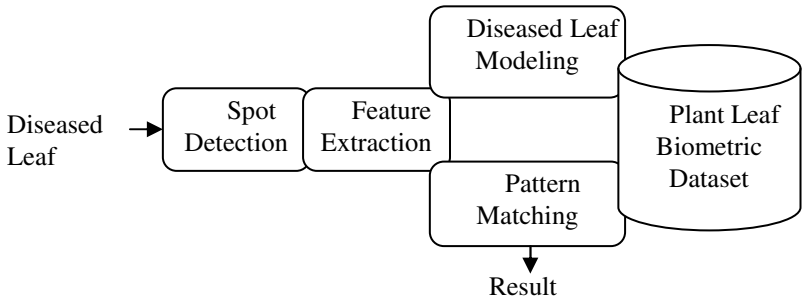


Fig. 1. Block flow diagram of Plant Biometric

In the first module, the spots on the leaves are detected and segmented using color thresholding in RGB color model. To segment the diseased portion from a color leaf image histogram based thresholding method [9] is used. A histogram in an image shows how many pixels are there with certain intensity value or pixel values in each

layer (R, G, and B) in case of color images as it is a 3 dimensional array. Using a threshold pixel value region of interest can be separated out from the image. Double thresholding is used in all the three layers in this paper (ρ_{red1}, ρ_{red2}, ρ_{green1}, ρ_{green2}, ρ_{blue1} and ρ_{blue2}). The processing done in first module is shown in fig 2 and the formula used is as below:

$$\begin{gathered} \textit{if Red} <= \rho_{red1} \textit{ and Red} >= \rho_{red2} \\ \textit{then if Green} <= \rho_{green1} \textit{ and Green} >= \rho_{green2} \\ \textit{then if Blue} <= \rho_{blue1} \textit{ and Blue} >= \rho_{blue2} \\ \textit{then Image(x, y, Red/Green/Blue)=0} \end{gathered}$$

The second method is feature extraction using Gabor Wavelet transform extract texture information from the segmented disease portion in plant leaf, discussed in details in the next section. Then for feature vector classification support vector machine (SVM) is used.

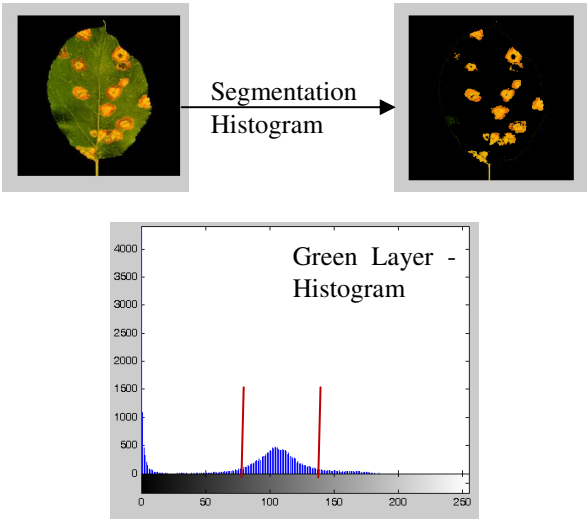


Fig. 2. Spot detection pre-processing

3 Feature Extraction Using Gabor Wavelet Transform

In signal analysis Fourier Transform was mainly used but since last two decades short time Fourier transform, STFT are used because it has time as well as frequency information of the signal simultaneously. The Gabor transform is like of STFT as it is a combination of Fourier transform kernel over a Gaussian function. The frequency resolution of a Gabor transform [10] is better than STFT as the Gaussian signal more concentrated than rectangular function in frequency domain. The Gabor transform is a 2D filter given by equation (1).

$$Gabor(t,w) = \int_{-\infty}^{\infty} e^{-\frac{(\tau-t)^2}{2}} e^{-jwt} x(\tau) d\tau \quad (1)$$

Wavelet based signal analysis is used to decompose a signal using an orthogonal basis functions as its energy is concentrated in time domain [11]. Gabor wavelet was introduced first by Dennis G. [10]. In image processing it is used because it covers both multiresolution and multidirectional properties and so is very much applicable for image analysis and its applications such as face and facial expression classification.

Gabor wavelet transform (GWT) is a time-frequency tool for identifying the rapidly varying characteristics of different wave signals. The Gabor wavelet [12] is defined as:

$$\Psi_{u,v}(x,y) = \frac{|\kappa_{u,v}|^2}{\sigma^2} e^{-|\kappa_{u,v}|^2 |(x,y)|^2 / 2\sigma^2} \left[e^{i\kappa_{u,v}(x,y)} - e^{-\sigma^2/2} \right] \quad (2)$$

Here, in this equation (2) *Ψ()* is a Gabor wavelet function with (x, y) co-ordinates of each pixels in Cartesian system where x ranges from 0 to height 'h' and y ranges from 0 to width 'w' of image and e is the function for oscillation having a Gauss window function defined by power of exponential in equation (2) specifying the localization of Gabor filter both in time and frequency domain. The main intention of using Gauss window function is to limit the range of oscillation function. *k* is the wave-vector of scale *v* and orientation *u*. if the values of *(u, v)* is changed then different Gabor filter is obtained. Similarly, a set of Gabor filter can be obtained with a constant $\sigma = 2n$. For experiment purpose five different scale values, *v* is used *{0, 1, 2, 3, 4}* and eight orientation, *u* is used *{0, 1, 2, 3, 4, 5, 6, 7}* resulting 40 morphology of Gabor filters as seen in the figure 3.

Pattern recognition and computer vision are most important application of Gabor wavelet transform. An image is represented using Gabor Wavelet describing both spatial frequency structure and spatial relations. Image by Gabor wavelet transform is same as a human mind perceives. The 40 different images of leaf are shown in fig 4. The advantages of using GWT are: it is invariant to some degree with respect to translation, rotation and dilation. Secondly it saves neighborhood relationships between pixels in image and it is a robust method against illumination and noise. GWT is a fast and low computational cost method to represent an image.

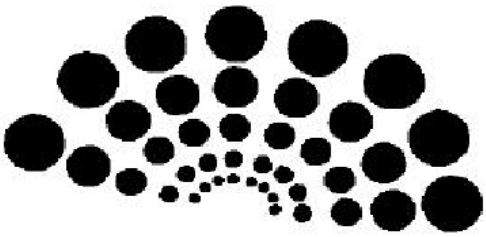


Fig. 3. 40 different morphology of Gabor filters applied on 2D images

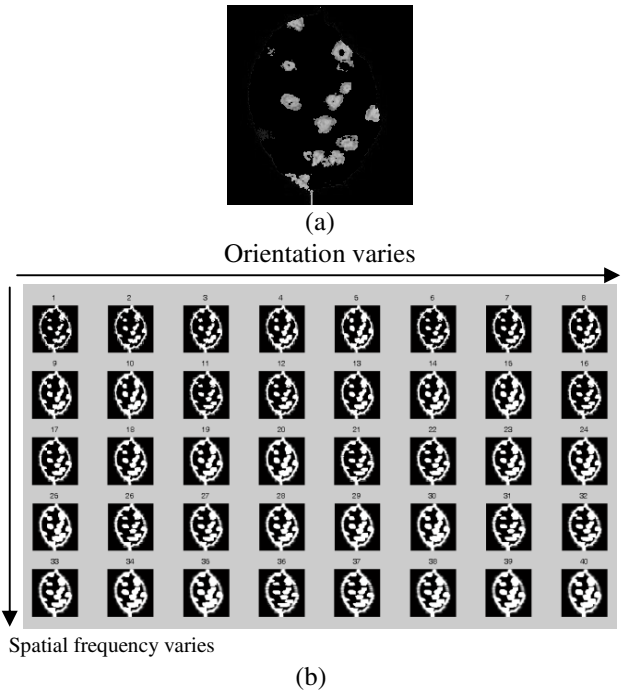


Fig. 4. (a) Original segmented diseased leaf in gray. (b) Gabor wavelet representation of diseased leaf.

The image of size *n x m* after applying GWT a feature vector, F_{vector} is generated of length *n x m x* 40 coefficients or features. This is the only drawback of GWT creating high dimension feature space. This feature vector is used to train and test the system. The classifier used in this paper is support vector machine.

4 Experimental Result

In this paper an automated plant biometric system is designed to test and identify the disease type in the plant leaf. The leaf samples used train and test support vector machine is self prepared home dataset. Each image in the plant leaf biometric dataset is of size 100x100 and so the feature vector, F_{vector} is of size 100x100x40, which comes out to be 400000 features. As the size of feature vector is too big and because of which SVM learning algorithm is used. For the experiment purpose Radial base function, RBT kernel SVM is used as it is the most popular kernel function defined as

$$k(x, x') = e^{-\gamma||x-x'||^2} \tag{4}$$

RBF kernel basically adds bumps to the low dimension data to represent it in a high dimension for proper and correct classification of test samples.

Before classification the dimension of the feature vector is reduced by applying a well known dimension reduction algorithm PCA. Principal component analysis reduces the dimension of new feature space and removes all unwanted and redundant features from the feature vector. PCA maps high dimension feature space to a same significant low dimension feature space. The redundancy in feature vector is common problem for Gabor wavelet transform. Then this new reduced feature vector, Γ_{vector_PCA} is used for training and testing.

The complete system is designed in Matlab 7.7 (2008b) tool with Window XP operating system having Core 2 DUO processor and 3GB RAM. Gabor wavelet transform is performed by using five scales and eight different orientations with $\sigma = 2n$ fixed. The performance is evaluated on home dataset containing 66 images (a very small dataset, $R=(R_{train}, R_{test})$) of six different leaf disease classes. The six leaf diseases are: Tikka a fungus disease in groundnut, Powdery Mildew, Downy Mildew in apple, graph, and mango plants, Late Blight and Early Blight in potato and tomato plants, and Rust in apple species. The threshold values for Powdery and Downy Mildew are different with other diseases as there diseased portion is bit similar to the leaf green because of which no histogram equalization technique is used as it reduces the algorithm accuracy.

The testing method used in this paper is one-versus-rest. In this one test image is tested with all other images in the dataset and so also known as one-versus-all test. All the possible parameters are considered while testing the sample images. Out of 66 images in dataset 48 are used for training and rest 18 are used for testing, 3 images from each disease class. Table 1, shows the accuracy result of testing.

Table 1. Comparison and accuracy result

Diseases	**Wavelet (db4)** Same Class (out of 3)	**Gabor Wavelet** Same Class (out of 3)	**Gabor Wavelet** One-Versus-all (out of all)
Tikka	3	3	0
Powdery Mildew	2	2	1
Downy Mildew	1	3	0
Late Blight	2	3	0
Early Blight	2	2	1
Rust	3	3	0

Table 1, shows that powdery mildew and downy mildew are diseases that are confused with other diseases. There are many other algorithms used for pattern recognition but Gabor wavelet perceives digital images as an individual human mind dose. Because of which it is very popular and used in this paper for natural plant leaf disease recognition and other natural images [13, 14, 15]. In Wavelet transform filtering and sub-samplings are involved and but lack orientation features. Ma and Manjunath (1995) worked on Wavelet and showed that the GWT is far better than Wavelet transform and has best accuracy rate, as in figure 5.

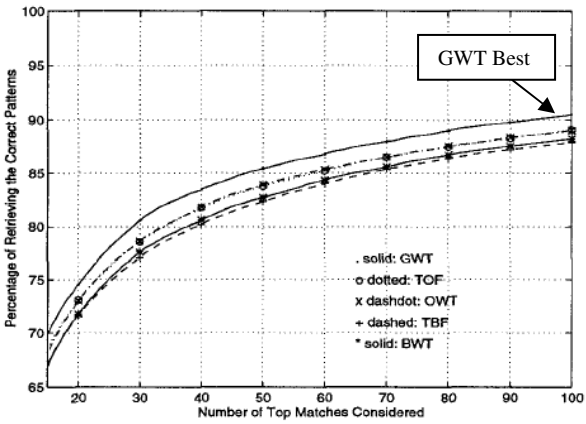


Fig. 5. Comparison of different pattern classification methods [16]. GWT: Gabor Wavelet transform; TOF: Tree-structure decomposition using orthogonal filter bank; OWT: Orthogonal Wavelet transform; TBF: Tree-structure decomposition using bi-orthogonal filter bank; BWT: Bi-orthogonal Wavelet transform.

5 Conclusion and Future Work

In this paper, a histogram thresholding technique is used to segment the diseased portion form plant leaf image and first time Gabor Wavelet is used to represent diseased plant leaf. The result shows that the implemented plant biometric system is a robust and is independent for plant leaf disease detection and identification. The accuracy rate is about 89% with homemade dataset which is better than other already existing methods. The GWT is 2D multiresolution and multidirectional feature extraction method used for gray level natural image. The system developed is also tested for incomplete diseased leaf images and results good.

GWT is restricted to some natural images such as face recognition, biometric system and others [13]. The other important advantage is that it is very simple and implement with a great high accuracy but need to fix threshold values for some diseases. To overcome this, unsupervised segmentation algorithm [17] with GWT can work fine and should be considered for future work. In future the dataset can also be expanded to some other leaf disease samples and can be used to identify herbal plants.

Acknowledgements. The research was obtained from **"Science and Technology Discovery Park"** project from **Department of Science and Technology,** New Delhi to promote the latest knowledge, tools and techniques for enhancing the productivity of fruits and vegetables in India. The present work was completed as one of the part in the project at **Indian Institute of Information Technology, Allahabad** and is continued at **IIT Roorkee.** We would specially like to thank Mrs. Shikha Prasad, Banaras Hindu University, Varanasi for their help full support.

References

1. Ballard, D.H.: Generalizing the Hough Transform to Detect Arbitrary Shapes. Pattern Recognition 13(2), 111–122 (1981)
2. Ponce, J.: Lecture 26: Edge Detection-II, http://wwwcvr.ai.uiuc.edu/~ponce/fall04/lect26.ppt (February 12, 2004)
3. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proceedings of the International Conference on Computer Vision, vol. 2, pp. 1150–1157 (1999)
4. Wang, Z., Chi, Z., Feng, D.: Leaf image retrieval using a two-step approach with shape features. In: Proc. 2000 IEEE Pacific-Rim Conf. on Multimedia (PCM 2000), Sydney, Australia, pp. 380–383 (December 2000)
5. Wang, Z., Chi, Z., Feng, D.: Shape based leaf image retrieval. IEE Proc., Vis. Image Signal Process. 150(1), 34–43 (2003)
6. Soille, P.: Morphological image analysis applied to crop field mapping. Image Vis., Comput. 8(13), 1025–1032 (2000)
7. Prasad, S., Kudiri, K.M., Tripathi, R.C.: Relative Sub-image Features for Leaf Recognition using Support Vector Machine. In: Processing of International Conference on Communication Computing and Security, pp. 343–346. ACM (February 2011)
8. Prasad, S., Kumar, P., Tripathi, R.C.: Plant leaf species Identification using Curvelet Transform. In: Int. Conference on Computer and Communication Technology, pp. 646–652. IEEE (2011)
9. Carlotto, M.J.: Histogram Analysis Using a Scale-Space Approach. IEEE Trans. Pattern Analysis and Machine Intelligence, PAMI-9, 121–129 (1997)
10. Gabor, D.: Theory of communication. Journal of Institute of Electrical Engineers 93, 429–457 (1946)
11. Prasad, S., Verma, G.K., Singh, B., Kumar, P.: Basic handwritten Character Recognition from Multi-lingual Image Dataset using Multiresolution and Multidirectional transform. International Journal of Wavelets, Multiresolution and Information Processing (2012)
12. Lee, T.S.: Image representation using 2D Gabor wavelets. IEEE Trans. Pattern Analysis and Machine Intelligence 18(10) (1996)
13. Fang, Y., Deyun, C., Rui, W.: Text Feature Extraction of Natural Scenes using Gabor Wavelet Transformation based on Scale Overlapping. In: 6th International Forum on Strategic Technology (IFOST 2012), vol. 2, pp. 1062–1064. IEEE (2011)
14. Zhang, C., Guo, K., Yu, G.: Analysis of Gabor Wavelet Algorithm for Tracking Driver's Feature Point. In: International Conference on Electrical and control Engineering, ICECE 2010, pp. 364–368. IEEE (2010)
15. Alimohamadi, H., Ahmadyfard, A., Shojaee, E.: Defect Detection in Textiles using Morphological Analysis of Optimal Gabor Wavelet Filter Response. In: International Conference on Computer and Automation Engineering (ICCAE 2009), pp. 26–30. IEEE (2009)
16. Ma, W.Y., Manjunath, B.S.: A comparison of Wavelet transform features for texture image annotation. In: International Conference on Image Processing, vol. 2, pp. 256–259. IEEE (1995)
17. Prasad, S., Kumar, P., Jain, A.: Detection of Disease Using Block-Based Unsupervised Natural Plant Leaf Color Image Segmentation. In: Panigrahi, B.K., Suganthan, P.N., Das, S., Satapathy, S.C. (eds.) SEMCCO 2011, Part I. LNCS, vol. 7076, pp. 399–406. Springer, Heidelberg (2011)

Analysis of Vasculature in Human Retinal Images Using Particle Swarm Optimization Based Tsallis Multi-level Thresholding and Similarity Measures

Nadaradjane Sri Madhava Raja[1], Ganesan Kavitha[2], and Swaminathan Ramakrishnan[3]

[1] Department of Electronics and Instrumentation Engg., St. Joseph's College of Engineering, Sholinganallur, Chennai, India
[2] Department of Electronics Engineering, M.I.T. Campus, Anna University, Chennai, India
[3] Biomedical Engineering Group, Department of Applied Mechanics, Indian Institute of Technology Madras, Chennai, India

Abstract. Retinal vasculature of the human circulatory system which can be visualized directly provides a number of systemic conditions and can be diagnosed by the detection of lesions. Changes in these structures are found to be correlated with pathological conditions and provide information on severity or state of various diseases. In this work, particle swarm optimization algorithm based multilevel thresholding is adopted for detecting the vasculature structures in retinal fundus images. Initially, adaptive histogram equalization is used for pre-processing of the original images. Tsallis multilevel thresholding is used for the segmentation of the blood vessels. Further, similarity measures are used to quantify the similarity between the segmented result and the corresponding ground truth. The optimal multi-threshold selection using particle swarm optimization seems to provide better results. Similarity measures analysis using dendrogram and box plot provide validation of the segmentation procedure attempted.

Keywords: Retinal vasculature, Particle swarm optimization, Tsallis multi-threshold, Similarity measures.

1 Introduction

Human eye is the most sophisticated organ with perfectly interrelated subsystems and retina is the inner most layer of the eye. The retina contains anatomic structures such as the vasculature, optic disc and macula [1]. The optic nerve is the region where the blood vessels and nerve fibers pass through the sclera. It is sensitive to the changes associated with intraocular pressure associated with retinal diseases [2].

The structure of retinal vessels plays an important role in revealing the state of diseases. Diabetic retinopathy and glaucoma are the major causes of blindness. The symptoms of glaucoma include the increase in pressure within the eye. The increase in pressure damages the optic nerve that carries the vital information from the retina

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 380–387, 2012.

to the brain. Diabetic retinopathy is a complication of diabetes mellitus. The tiny dilations of the blood vessels are the first unequivocal sign of diabetic retinopathy. Choroidal neovascular membrane involves the development of new and abnormal vessels below the retina [3].

Digital fundus images are used to assess the health condition of an eye. Colour fundus images provide anatomical information about the retina and are more suitable for automated analysis. The extraction of blood vessels from these images helps the ophthalmologists to diagnose eye diseases [4].

Segmentation is the process of partitioning an image to multiple segments. It is typically used to locate objects and boundaries. The features extracted from the segmented boundary can be used for the detection of abnormalities in the images [5]. Many segmentation methods such as graph search method, contourlet transform and morphological operations have been used for the segmentation of blood vessels [6-9].

Segmentation methods based on optimal thresholding have been attempted recently. The Otsu method is combined with modified Artificial Bee Colony (ABC) and Particle Swarm Optimization (PSO) algorithm for segmenting computed tomography lung images [10]. The Otsu thresholding and Ant Colony Optimization (ACO) method has been used to identify optic disc and to analyze the macula [11].

Recent developments of statistical mechanics on non-extensive entropy, called Tsallis entropy, intensified interest of investigating a possible extension of Shannon's entropy to Information Theory. The Tsallis entropy is a new proposal in order to generalize the Boltzmann/Gibb's traditional entropy to non-extensive physical systems. A multi-level Tsallis thresholding method with ABC optimization has also been used in improving image segmentation [12]. Maximum Kapur's and Otsu's entropy based multi-level thresholding using modified bacterial foraging algorithm have been applied to general images and results obtained have been compared with bacterial foraging algorithm, PSO and genetic algorithm [13]. Particle swarm adaptation operates on a population of search points that probe the search space simultaneously. It is highly effective and adaptable to diverse application, with potential for hybridization and integration into a range of intelligent systems. It is an optimization paradigm which models the exploration of a problem space by a population of individuals. Zhang Xue-Feng has proposed a method for segmentation of images based on PSO and rough entropy standards [14]. Yin proposed particle swarm algorithm based multilevel minimum cross entropy procedure for segmentation [15].

In this work, Tsallis multi thresholding combined with PSO is applied to segment the retinal blood vessels and the segmented result efficacy is validated using different similarity measures obtained by comparing with their corresponding ground truth.

2 Methodology

In this study, digital retinal images (N = 40) are obtained from public database and nearby hospitals. Images of similar sizes are considered for analysis. The retinal images are pre-processed using adaptive histogram equalization technique and further subjected to Tsallis multi thresholding technique with PSO based optimization.

PSO is a population based heuristic method discovered through simulation of social models of bird flocking, fish schooling and swarming. Let X and V denote the particle's position and its corresponding velocity in search space respectively. At iteration K, each particle 'i' has its position defined by $X_i^k = \left[X_{i,1}, X_{i,2}, ...X_{i,N}\right]$ and each velocity is defined as $V_i^k = \left[V_{i,1}, V_{i,2}, ...V_{i,N}\right]$ in search space N. Velocity and position of each particle in the next iteration can be calculated as:

$$V_{i,n}^{K+1} = W * V_{i,n}^{K} + C_1 * rand_1 * (pbest_{i,n} - X_{i,n}^{k}) + C_2 * rand_2 * (gbest_n - X_{i,n}^{k}) \quad (1)$$

Where i = 1, 2,…p and n = 1, 2,…m

$$X_{i.n}^{K+1} = \begin{cases} X_{i,n}^{k} + V_{i,n}^{k+1}; & if\ X_{mini,n} \le X_{max\,i,n} \\ X_{min_{i,n}} & ;\quad if\quad X_{i,n}^{k+1} \le X_{mini,n} \\ X_{max_{i,n}} & ;\quad if\quad X_{i,n}^{k+1} \le X_{maxi,n} \end{cases} \quad (2)$$

The inertia of weight W is an important factor for the PSO's convergence. It is used to control the impact of previous history of velocities on the current velocity. A large inertia weight factor facilitates global exploration (i.e., searching of new area) while small weight factor facilitates local exploration. Here larger weight factor are chosen for initial iterations and gradually reduce weight factor in successive iterations. Acceleration constant C_1 called cognitive parameter pulls each particle towards local best position where as constant C_2 called social parameter pulls the particle towards global best position. The process is repeated until stopping criterion is reached. This paper presents an approach to the multilevel image thresholding problems using the PSO algorithm [13].

In Tsallis multilevel thresholding, L represents gray levels in a given image and these gray levels are in the range {0, 1, 2…,(L-1)}. The probability of the pixel is defined as Pi = h(i)/N, (0 ≤ i ≤ (L-1)) , where h(i) denotes number of pixels for the corresponding gray-level L and N denotes total number of pixels in the image which is equal to $\sum_{i=0}^{L-1} h(i)$. Tsallis entropy criterion method [16] for multilevel thresholding is described as follows:

$$f(t) = \arg\max[S_q^A(t) + S_q^B(t) + ... + S_q^M(t) + (1-q)S_q^A(t)S_q^B(t)....S_q^M(t)] \quad (3)$$

Where,

$$S_q^A(t) = \frac{1 - \sum_{i=0}^{t_1-1} (\frac{P_i}{P^A})^q}{q-1}, P^A = \sum_{i=0}^{t_1-1} P_i \quad , \quad S_q^M(t) = \frac{1 - \sum_{i=0}^{t_1-1} (\frac{P_i}{P^M})^q}{q-1}, P^m = \sum_{i=t_m}^{t_m-1} P_i \quad (4)$$

Further, in this work, eleven similarity measures are used to quantify the similarity between segmentation results and the ground truth. Hierarchical clustering is conducted to estimate the similarity among the measures collected. A hierarchical cluster tree is created using ward inner squared distance or min variance algorithm [17]. Segmentation under test is the segmentation that has an unknown evaluation and reference segmentation is the segmentation with a known evaluation.

Table 1. Binary similarity measure

Sl. No	Similarity measure	Formula
1	Simple matching	$\frac{a+d}{a+b+c+d}$
2	Rogers and Tanimoto	$\frac{a+d}{a+d+2(b+c)}$
3	Hamann measure	$\frac{(a+d)-(b+c)}{a+b+c+d}$
4	Sokal and Sneath-II	$\frac{2(a+d)}{2(a+d)+b+c}$
5	Jaccard measure	$\frac{a}{a+b+c}$
6	Anderberg measure	$\frac{t_1-t_2}{2(a+b+c+d)}$ $t_1=\max(a,b)+\max(c,d)+\max(a,c)+\max(b,d)$ $t_2=\max(a+c,b+d)+\max(a+b,c+d)$
7	Czekanowsky	$\frac{2a}{2a+b+c}$
8	Braun and Banquet	$\frac{a}{\max(a+b,a+c)}$
9	Pearson and Heron-II	$\frac{ad-bc}{\sqrt{(a+b)(a+c)(b+d)(c+d)}}$
10	Baroni-Urbani and Buser-I	$\frac{\sqrt{ad}+a}{\sqrt{ad}+a+b+c}$
11	Baroni-Urbani and Buser-II	$\frac{\sqrt{ad}+a-(b+c)}{\sqrt{ad}+a+b+c}$

Table 1 shows the various similarity measures, used for the validation of the segmentation methods in this paper. Simple matching similarity measure is the ratio of the number of matches to the total number of characteristics. Jaccard similarity measure is also known as the similarity ratio. Hamann similarity measure gives the probability that a characteristic has the same state in both items (present in both or absent from both) minus the probability that a characteristic has different states in the two items (present in one and absent from the other). All the similarity measures have the range of 0 to 1 except Hamann, which has the range of -1 to +1 [18, 19].

Let $X_{i,j}$ be the ground truth and $Y_{i,j}$ be the segmentation under test then, a = number of times $X_{i,j}=1$ and $Y_{i,j}=1$, b = number of times $X_{i,j}=0$ and $Y_{i,j}=1$, c = number of times $X_{i,j}=1$ and $Y_{i,j}=0$, d = number of times $X_{i,j}=0$ and $Y_{i,j}=0$.

The value 'a' is the number of pixels with positive matches, b is the number of pixels in $X_{i,j}$ absence mismatches, c is the number of pixels in $Y_{i,j}$ absence mismatches and d is the number of pixels with negative matches. The diagonal sum a+d represents the total number of matches between $X_{i,j}$ and $Y_{i,j}$, the other diagonal sum b+c represents the total number of mismatches between $X_{i,j}$ and $Y_{i,j}$. The total sum of the table, a+b+c+d is always equal to total number of pixels i×j. [20]. Analysis of the considered similarity measures is performed with dendrogram plot. The plot shows dependence of various similarity measures with cluster formations between the interdependent data.

3 Results and Discussion

In this work, normal and abnormal retinal samples from age matched, healthy and adult volunteers are obtained from DRIVE database and nearby hospitals for study. The retinal images acquired through a high sensitive colour fundus camera with constant illumination, resolution, field of view, magnification and dilation procedures are used for the study. Input images of same size (256 x 256) are considered for analysis.

Fig. 1. Retinal fundus image 1 (a), Ground truth (b) , PSO based multilevel Tsallis thresholding (c)

Fig. 2. Retinal fundus image 2 (a), Ground truth (b) , PSO based multilevel Tsallis thresholding (c)

Typical retinal images and their corresponding ground truth are shown in figures 1(a), 2(a) and 1(b), 2(b) respectively. The pre-processed images after subjecting to PSO shown in figures 1(c) and 2(c) demonstrate enhanced visual quality of the blood

vessels. PSO based method provide better performance in extraction of vascular networks and in detection of smaller vessels. The numbers of disconnected vessel segments are also found to be less. Some lesions are also detected in addition to blood vessels in the abnormal retinal images. The high degree of visual quality in the images obtained can be attributed to the ability of the hybrid optimization based thresholding method to retain more features in the given images.

The dendrogram shown in figure 3 gives three distinct clusters of similarity measures for detection of retinal blood vessels using PSO based optimal multi thresholding technique. The cluster 1 includes the Rogers and Tanimoto, Anderberg measure, Hamann measure, Baroni-Urbani and Buser-II, Pearson and Heron. These similarity measures range from 0.80 to 0.87. The similarity measures bundled in cluster 2 has the range of 0.90 to 0.93 and the measures are simple matching, Jaccard measure, Baroni-Urbani and Buser-I. The similarity measures corresponding to cluster 3 are Sokal and Sneath-II, Czekanowsky measure, Braun and Banquet measure. These clustered similarity measures have almost the same range varying from 0.92 to 0.96.

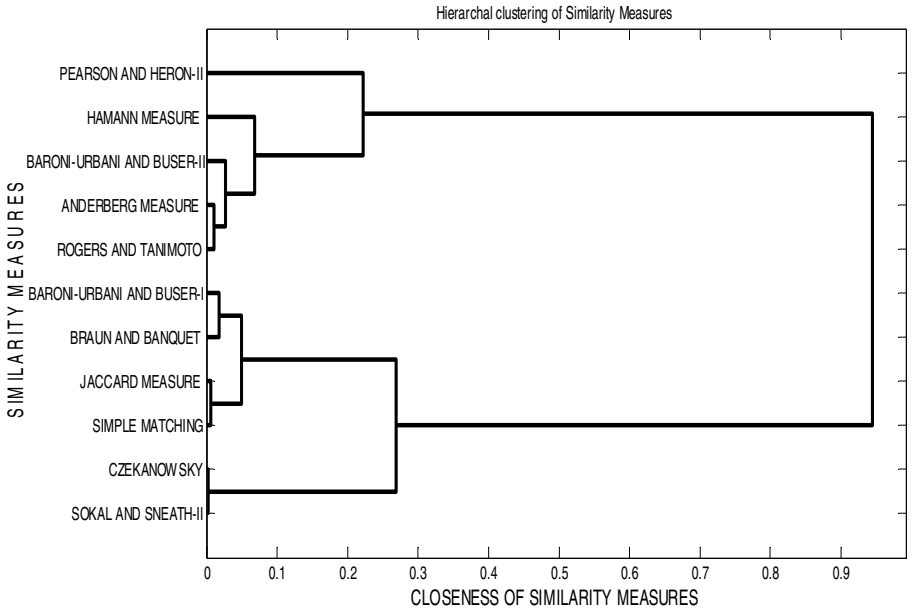


Fig. 3. Dendrogram of similarity measures obtained from PSO based Tsallis multilevel thresholding method for normal images

Figure 4 represents the box plot of the various similarity measures obtained from comparing segmented vessels with their corresponding ground truth. The mean values of similarity measures belonging to first cluster which include Rogers and Tanimoto, Anderberg measure, Hamann measure, Baroni-Urbani and Buser-II, Pearson and Heron-II measures are 0.85, 0.84, 0.83, 0.84, and 0.84 respectively.

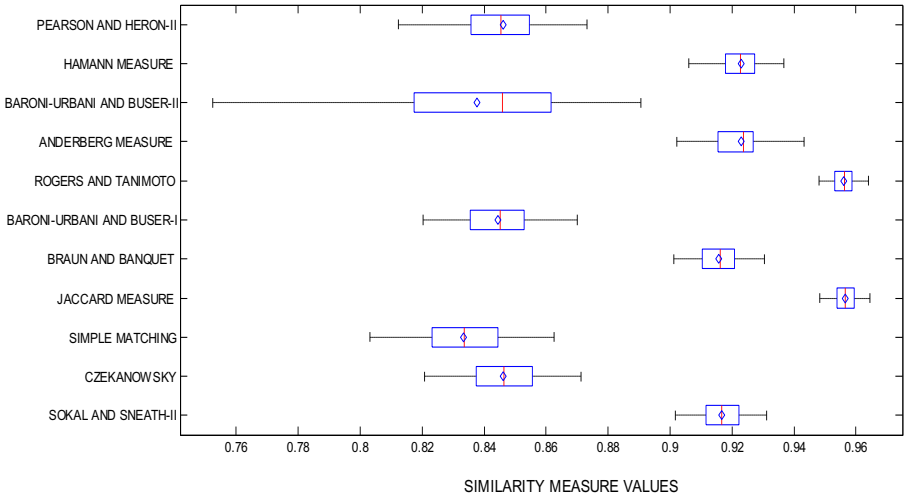


Fig. 4. Box plot representation of the different similarity measures

The similarity values of simple matching, Jaccard, Baroni-Urbani and Buser-I measures in second cluster have mean values of 0.91, 0.92 and 0.92 respectively. Measures of third cluster include Sokal and Sneath-II, Czekanowsky and Braun & Banquet measures. Their mean values are 0.95, 0.95 and 0.93 respectively.

4 Conclusion

In this work, an attempt has been made to detect and analyze retinal vasculature using PSO based multilevel thresholding method. The pre-processed PSO based Tsallis multi thresholding results in better detection of vasculature both in normal and abnormal images. Similarity measures show better validation of the segmentation algorithm when compared to their ground truth. Based on the dendrogram analysis of the similarity measures, it was observed that the clustered similarity measures such as Sokal & Sneath-II, Czekanowsky and Braun & Banquet have high and almost same range (0.92 to 0.96) of similarity values. Futher, the box plot also shows high mean value (approximately 0.95) for the same cluster. In addition, the results obtained could be used to address disorders such as diabetic retinopathy and conditions such as leaky blood vessels. Since the similarity measures are more relevant to their ground truth, the method seems to be clinically relevant and could be used in differentiating normal and pathological subjects.

References

1. Muthu, R.K.M., Rajendra, A.U., Chua, K.C., Lim, C.M., Ng, E.Y.K., Milind, M.M., Augustinus, L.: Application of Intuitionistic Fuzzy Histon Segmentation for the Automated Detection of Optic Disc in Digital Fundus Images. In: IEEE-EMBS International Conference on Biomedical and Health Informatics, Hong Kong and Shenzhen, China (2012)
2. Radhika, C.: A Morphological approach for discrimination between Glaucomatous and Non glaucomatous Eyes by Optic Disc Localization. Int. J. Adv. Research in Comput. Engg.&Tech. 1(4), 643–648 (2012)
3. Kumar, K.V.V., Ravi, T., Rani, B.M.S., Habibullah, K.: Detection of Retinal Diseases by Tracing Vessel Trees and Accurately Segmenting Vessels. Int. J. of Engg. and Adv. Tech. 1(4), 182–187 (2012)
4. Osareh, A.: Automated identification of diabetic retinal exudates and the optic disc. PhD Thesis, Department of Computer Science, Faculty of Engineering, University of Bristol, Bristol (2004)
5. Gayathri, D.B., Balasubramanian, R.: Segmentation of Optic Disk from Fundus Images of the Human Eye. In: International Conference on Computing and Control Engineering, Chennai (2012)
6. Rattathanapad, S., Mittrapiyanuruk, P., Kaewtrakulpong, P., Uyyanonvara, B., Sinthanayothin, C.: Vessel Extraction in Retinal Images using Multilevel Line Detection. In: IEEE-EMBS International Conference on Biomedical and Health Informatics, Hong Kong and Shenzhen, China (2012)
7. Xiayu, X., Meindert, N., Qi, S., Milan, S., Mona, K.G.: Vessel Boundary Delineation on Fundus Images Using Graph-Based Approach. IEEE Trans. Med. Imaging 30(6), 1184–1191 (2011)
8. Farnoosh, G., Seyed, M.Z., Hamid, R.P., Touka, B.: A Novel Method for Vessel Detection Using Contourlet Transform. In: IEEE Conference on Communications (2012)
9. Usman, A.M., Ibaa, J., Anam, T.: Blood Vessel Enhancement and Segmentation for Screening of Diabetic Retinopathy. TELKOMNIKA 10(2), 327–334 (2012)
10. Sushil, K., Tarun, K.S., Millie, P., Ray, A.K.: Adaptive Artificial Bee Colony for Segmentation of CT lung Images. Int. J. Comput. Appl. iRAFIT, 1–5 (2012)
11. Kavitha, G., Ramakrishnan, S.: Identification and Analysis of Macula in Retinal Images using Ant Colony Optimization based Hybrid Method. In: World Congress on Nature & Biologically Inspired Computing, NaBIC 2009, pp. 1174–1177 (2009)
12. Zhang, Y., Wu, L.: Optimal Multi-Level Thresholding Based on Maximum Tsallis Entropy via an Artificial Bee Colony Approach. Entropy 13, 841–859 (2011), doi:10.3390/e13040841
13. Sathya, P.D., Kayalvizhi, R.: Modified bacterial foraging algorithm based multilevel thresholding for image segmentation. Engg. Appns. of Arti. Intelli. 24, 595–615 (2011)
14. Zhang, X.F., Shang, J.K.: Application of Image Segmentation Algorithm Based on Particle Swarm Optimization and Rough Entropy Standard. In: Proc. of IEEE Conf. on Chinese Control and Decision Conference, pp. 2905–2909 (2009)
15. Yin, P.Y.: Multilevel minimum cross entropy threshold selection based on particle swarm optimization algorithm. Appl. Math. Comput. 184(2), 503–513 (2007)
16. Sathya, P.D., Kayalvizhi, R.: Optimum Multilevel Image Thresholding Based on Tsallis Entropy Method with Bacterial Foraging Algorithm. Int. J. Comp. Sci. Issues (IJCSI) 7(5), 336–343 (2010)
17. Everitt, B.S., Landau, S., Leese, M.: Cluster Analysis, 4th edn. Oxford University Press, Inc., New York (2001)
18. Anderberg, M.R.: Cluster analysis for applications. Academic Press, New York (1973)
19. Rosenburg, H.C.: Cluster analysis for researchers. Lifetime Learning Publications, Belmont (1984)
20. Choi, S.S., Cha, S.H., Tappert, C.: A Survey of Binary Similarity and Distance Measures. Journal on Systemics, Cybernetics and Informatics 8(1), 43–48 (2010)

Optimal Reactive Power Compensation for Improvement of Steady State Voltage Stability Limit under Stressed System Condition Using BF Algorithm

Santi Behera and Manish Tripathy

Department of Electrical Engineering, Veer Surendra Sai University of Technology (VSSUT), Burla, Odisha 768018, India
bsanti.uce@gmail.com, manish_tripathy@yahoo.co.in

Abstract. Power system operation under stressed condition, can lead to voltage instability problem if subjected to additional load increase or contingency. Suitable amount of reactive power compensation at proper location in the system improves the loadability. This work aims at obtaining optimal size and location of compensation in the 39- bus New England system with the help of Bacteria Foraging and Genetic algorithms. To reduce the computational time the work first, identifies weak candidate buses in the system, and then picks only few of them to take part in the optimization. The objective function is based on a recently proposed voltage stability index which takes into account the local equivalent network, which is a simpler and faster approach than the conventional CPF algorithm. BFA has been found to give better results compared to GA.

Keywords: Voltage stability, Bacteria Foraging Algorithm Genetic Algorithm.

1 Introduction

In power system voltage instability problem [1] is caused by the stressed load condition and contingencies in the form of outage of lines, generators etc. Research in this area has tried to determine effective voltage stability indices so that a voltage collapse scenario may be well predicted either in off-line or on-line manners. Various works have established different kinds of such indices which can foretell either steady or dynamic state voltage stability conditions[2-4] when the system is subjected to rise of load or face contingencies. To improve upon the solvability issues of system particularly near the Saddle Node Bifurcation(SNB) point of the system research community have resorted to more stable Continuation Power Flow (CPF) algorithm[5] and their variants. However, to do away with the demerits of computational difficulty of CPF, which restricts its use for an online evaluation, several works have focussed on detection of voltage instability with the help of local bus voltage phasor [6] or line based indices. Recently, a new voltage stability index termed as Equivalent Node Voltage Collapse Index(ENVCI) [7], has been proposed, which has the advantage of representing the effect of rest of the system outside local

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 388–397, 2012.

network into it. The simplicity and speed of computation of this index, has encouraged this work to consider it in formulating the problem. Moreover, recent advances in reactive power compensation devices and technologies have helped systems to improve upon voltage stability margins by suitably sizing and placing VAR compensating devices [8]. The key issue of suitable size and location has been determined by several research works after formulating the problem as an optimization problem. In this regard, conventional algorithms like Sequential Quadratic Programming(SQP), Linear Programming (LP) and Interior point based methods [9-11]and intelligent search algorithms like Genetic Algorithm(GA), Particle Swarm Optimization (PSO)[12,13], have been applied to solve the problem based on their merits over others. Recently a new intelligent technique based method, known as Bactria Foraging Algorithm (BFA), which duplicates the principles of foraging behaviour of *e-coli* bacteria present in human intestine, has drawn attention of research community with its wide applicability in power systems problems [14-15]. BFA has shown unique features of handling functions which are highly nonlinear and epistatic. This work aims at optimizing the suitable location and capacity of VAR Compensation (VC) in the IEEE 39-Bus New England power system so as to maximize the system loadability when operating under a stressed conditions. The paper is organized in the following sections. In Section 2, the basic concepts of the ENVCI is introduced and section 3 informs about BFA and its algorithm steps. Similarly, in section 4 the basic approach in the problem formulation is discussed, where as Section 5 highlights about the Test system. In the same section, some preliminary work is carried out to determine the weak buses in the system. Section 6 probes into the optimization problem and methodologies adopted for simulations Section 7 concludes the findings of the work.

2 Equivalent Node Voltage Collapse Index (ENVCI)

Traditionally, the point of voltage instability has been predicted with the help number of methods as discussed above. In this work, a node based index proposed recently[7], known as Equivalent Node Voltage Collapse Index (ENVCI) has been utilized to predict the instability point and thus find the static margin of loadability in any system. As highlighted in the paper, the ENVCI has some obvious advantages over other methods which can be summarized as below.

I. By modelling an equivalent local network, it takes into account the effect of entire system, including the part of the system which is outside the local network. Therefore it should provide a more accurate approximation of system loadability in the event of load increase or contingency.
II. As this method avoids the use of continuation method for the purpose of voltage stability margin determination, so it can quickly take decisions regarding suitable compensation required in the system taking the data provided by Phasor Measuring Units (PMUs).

In this model, initially an Equivalent Local Network Model(ELNM) for any node N is evaluated, which consists of two parts. The first part contains the total of real and reactive powers leaving the node N and the second part consists of all the powers entering N through different lines connected to it. The effects of all the charging currents feeding N through numbers of line and their impedances are included to form an equivalent impedance Z_{eq} and the connecting lines sending end voltages are considered to find an equivalent V_{eq} of ELNM. A dummy voltage source E_k with impedance Z_{km} is added to the ELNM to include the effect of the system outside the local network and thus form a single line equivalent model of the entire system. The reader is advised to follow [7] for detail modeling and derivation required to evaluate E_k and Z_{km} . Finally, the ENVCI values of different Buses can be determined by using the following equation

$$ENVCI = 2(e_k e_n + f_k f_n) - (e^2{}_k + e^2{}_n) \tag{1}$$

Where, $\vec{E}_k = e_k + jf_k$ and $\overrightarrow{V_n} = e_n + jf_n$ $\vec{V}_n$ is the node N voltage. $\vec{E}_k$ is the voltage of the external system. The effect of the inclusion of external circuit can result better control actions to be taken .

3 Bacteria Foraging Algorithm: An Overview

Bacteria Foraging Optimization Algorithm (BFOA), recently proposed by Passino [15], is a unique parallel search algorithm based on the foraging attributes of a type of bacterial species present in human intestine, known as *E.coli*. The foraging process of each of these bacterium is defined by four processes known as Chemotaxis, Swarming, Reproduction and Elimination & Dispersal.

The details of the algorithm steps of the original version as proposed by Passino[15] and the modified version[14] are explained in the respective works.

4 Optimisation Problem Formulation

The objective of this work is to maximize the system loadability limit at key identified weak load buses, with the help of suitably locating and sizing reactive power compensation in the system. The system loadability limit known as the steady state Voltage Stability Limit (VSL) of the system. To evaluate the steady state VSL of the system with a particular pattern of load increase, the load at the i^{th} load bus is increased in steps (with equivalent increase in generation as well) till the unstable point is reached. This is illustrated in (2).

$$\begin{aligned} P_{L_i} &= P_{L_{io}}(1+\lambda) \\ Q_{L_i} &= Q_{L_{io}}(1+\lambda) \\ P_{G_i} &= P_{G_{io}}(1+\lambda) \end{aligned} \tag{2}$$

Where, λ is known as the *Load Parameter*, used to increase the real and reactive powers from their respective nominal values. The system VSL in this work is evaluated with the help of ENVC Index as introduced in Section 2. With the increase in load ENVCI value reduces, reaching towards zero near the VSL. Hence the corresponding λ value is the VSL of the system. The problem is formulated as a static constrained non-linear optimization problem, as illustrated below

$$\textit{Minimize} \quad F(x,u,\lambda) \tag{3}$$
$$\textit{Subject to} \quad g(x,u,\lambda) = 0 \tag{4}$$
$$h(x,u,\lambda) \leq 0 \tag{5}$$

Equations (4) and (5) respectively denote the set of equality and inequality constraints. Where *x* is set of all the node voltages and their respective angles and *u* the control variables are the location and amount of reactive power compensation in the system. Moreover, in power system problem, equation (4) defines the real and reactive power balance equations at all the system nodes, whereas (5) denotes the optimum limits of magnitudes of bus voltages, line and transformer power flows . To account for the inclusion of all these constraints the objective function *F(x,u)* is designed as in (6) below.

$$F = \left(1/min\{a_{ij}\}\right) + P_{loss} + Pf_1 + Pf_2 + Pf_3, \tag{6}$$

Where, $\lambda max = \{a_{ij}\}$ is a set of maximum values of load parameters (λ_{max}) each for *i* numbers of critical load bus power increase scenarios.

P_{loss} = Average values of Real Power Loss in the system for all *i* numbers of critical load bus power increase scenarios

$$Pf_1 = 10 * abs(sign(V_{min} - 0.9) - 1) + 10 * abs(sign(V_{max} - 1.1) + 1) \tag{7}$$
$$Pf_2 = 10 * abs(sign(trans_{max} - 10) + 1) \tag{8}$$
$$Pf_3 = 10 * abs(sign(line_{max} - 10) + 1) \tag{9}$$

Similarly, pf_1, pf_2 and pf_3 respectively denote the average values of penalty factors related to minimum and maximum voltages , transformer capacity and line capacity limits respectively. The first and second parts of the objective functions formulated in (6) take into account the voltage stability limit maximization and real power loss minimization. Each part of the objective function is suitably scaled. The penalty factors pf_1, pf_2, pf_3 for most of the cases return zero output, unless violation of limits.

5 Test System and Its Weakness

In this paper the studies are carried out on 10-machine, 39-bus New England power system[14]. The system has 46 numbers of lines. The 1st generator (G_1) is an equivalent representation of the U.S.-Canadian interconnection system.. In order to gain an insight into possible weak zones present in the system, some indices were evaluated, so that their values could throw some light on the choice of suitable location and size of compensation required by the system in the event of load increase and contingencies.

A number of sensitivity based analysis are carried out to determine the vulnerable buses where the voltage instability problem may be initiated much earlier than others. For this purpose four different indices known as P-loss sensitivity index, Q-loss sensitivity index, L-index, and V-Q eigen value sensitivity index are enumerated separately for all the buses present in the system. A weighted index (*WASI*) comprising the effects of all these indices is formulated, based on whose values a weakness bus ranking could be done. This approach helps to determine a worse case loading scenario in the system. The reader may please be referred for detailed mathematical equations for P-Loss, Q-Loss, L-index, and V-Q sensitivity indices in literatures [1,16]. The Weighted Average Sensitivity Index is defined as

$$WASI = (SI_1 * w + SI_2 * w + SI_3 * w + \cdots . SI_n * w)/n * w. \quad (10)$$

Where, w is weighing factor. Though any value may be chosen, in this work it is taken as 0.5. SI_1 SI_2 $SI_{3....}$ SI_n are the indices. The detail enumeration and results for the considered test system are discussed in Section-6.

6 Results and Discussions

Initially, the weak buses in the test system are evaluated with the help of various sensitivity indices discussed in section 5. The weighted average value of all such indices is then evaluated to determine the weak buses in the system. The weighted average sensitivity index (*WASI*) obtained for all the buses except PV buses under nominal condition of loading are then sorted out to decide as to which one of them are candidate buses those shall be loaded to simulate a stressed condition of power systems. Similarly, to take decision for the location of reactive power compensation among all the buses, 17 numbers of weak buses are chosen which may not be necessarily a load (PQ) bus. For the purpose of optimization, loads in top five weakest buses in the system mentioned in Table 2 are subjected to load increase till their respective ENVCI values reach the unstable mark of zero. Load increase scenario in all these buses are simulated first by keeping the reactive power (Q) constant and then by keeping their nominal value Power Factor (PF) constant. To determine the maximum loadability of the system, the minimum value of $\{a_{ij}\}$ as defined in section 4 obtained by loading each of the five weakest buses is maximized. A brief discussion for parameter selection and methodology adopted with both GA and BFA techniques is summarized in section 6.

Table 2. Weak Bus Ranking

INDEX VALUE(SI_{avg})	0.0136	0.0187	0.0448	0.0503	0.0562
BUS NO .	20	8	29	27	28
Rank	1	2	3	4	5

6.1 Optimization with GA

Simple conventional GA is adopted for optimization. The crossover and mutation probabilities are assumed to be 0.85 and 0.05 respectively. The location and amount of reactive power compensation at two locations are randomly initialized. All the top 17 numbers of weak buses are considered as candidates for compensation and two out of them are randomly chosen with randomly generated compensation amount in the range of -40% to 40%. The optimized results are shown in Table 3.

6.2 Optimization with BFA

The same philosophy as discussed above, was again adopted with BFA. The control parameters for BFA are judiciously chosen. 4 numbers of bacteria evolve in the optimization process consisting of 4 chemotactic stages. The values of *runlengthunit* and *elimination probability* are considered as 0.06 and 0.25 respectively .Simulations were carried out with both the above algorithms. The convergence characteristics obtained with these algorithms for both constant Q (*Case1*) and constant power factor types (*Case2*) of load shown in Fig. 1 and Fig.4 respectively, depicts the supremacy of BFA over GA.

Table 3. Optimized Compensation Results with GA and BFA

Compensation Specifications	Case1(Constant Q-Load)		Case2(Constant *pf*-Load)	
	GA	BFA	GA	BFA
1st Location	Bus No. 28	Bus No. 24	Bus No. 27	BusNo. 15
1st Amount	43.35%	43.78%	33.55%	17.82%
2nd Location	Bus No. 20	Bus No. 15	Bus No. 18	BusNo. 27
2nd Amount	31.57%	08.68%	12.63%	40.05%
Obj.FunctionValue	0.7106	0.5058	0.7273	0.7196
Constraint	48.52	28.72	49.31	47.63
QQ	2.1402	2.3162	2.3078	2.2428

Though, the objective function values are obtained with both GA and BFA are almost similar to each other, but GA has fared quite badly in terms of limiting constraints violation. The optimization results are depicted in Table-3. The ENVCI values obtained at each step of load increase till the point of instability, are illustrated in Fig.3 and fig. 6 respectively for *Case1* and *Case2*. Similarly, figures of all the bus voltages profiles at the last stable loading point are depicted in Fig .2 and Fig.5 for both the cases.

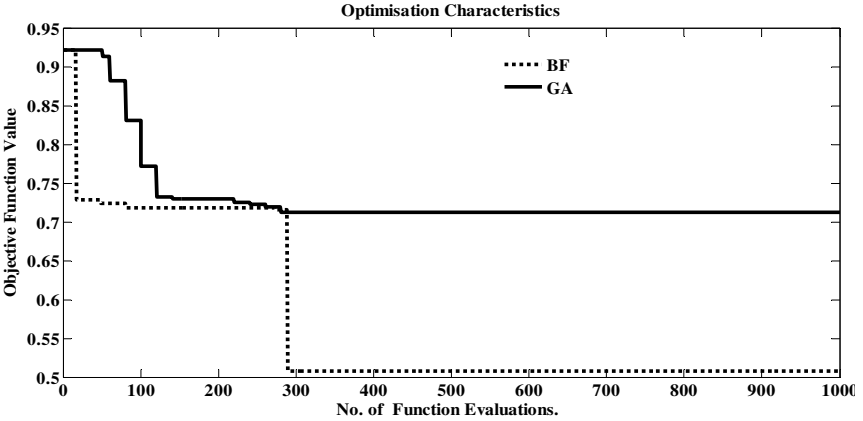


Fig. 1. Convergence Characteristics for Case-1

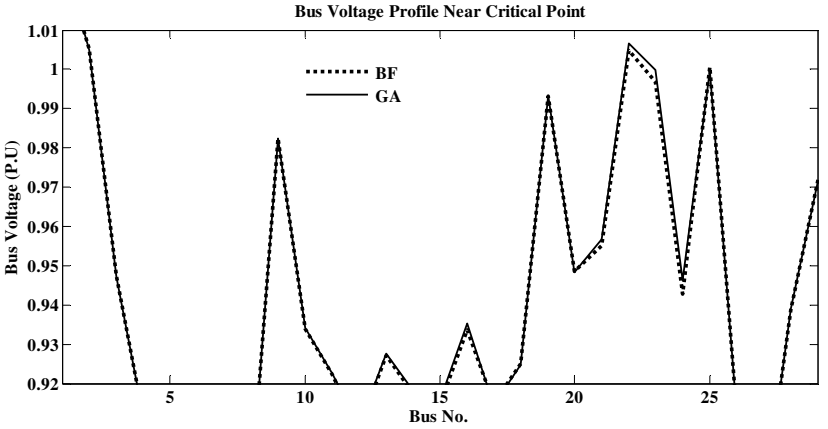


Fig. 2. Bus Voltage Profiles at VSL in Case-1

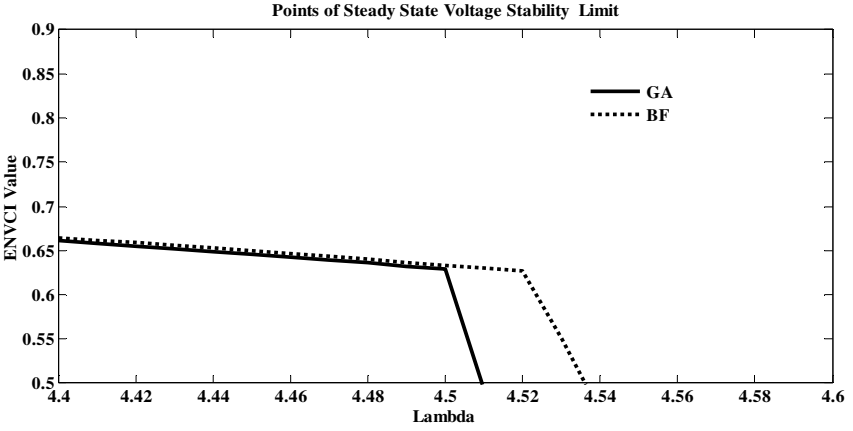


Fig. 3. Voltage Stability Limits for Case-1

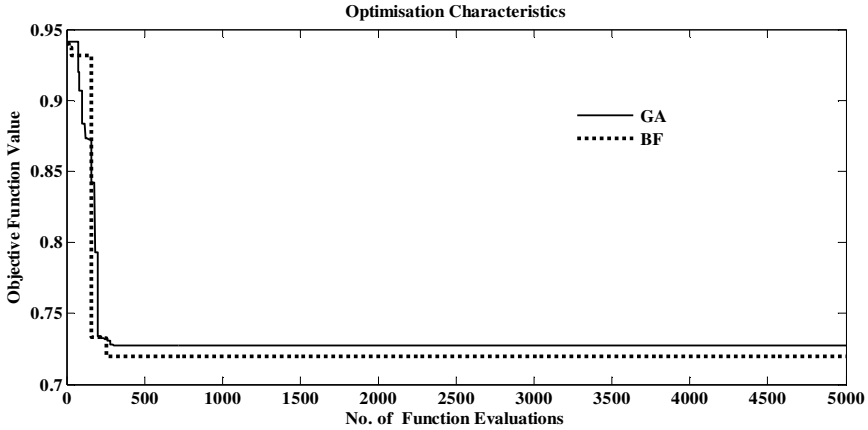


Fig. 4. Convergence Characteristics for Case-2

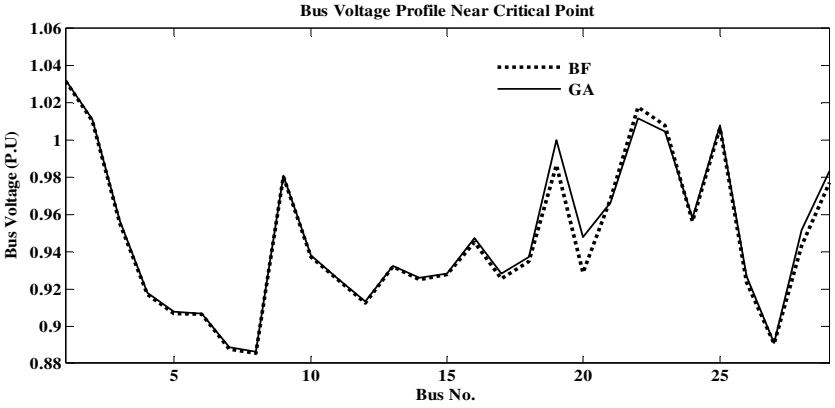


Fig. 5. Bus Voltage Profiles at VSL in Case-2

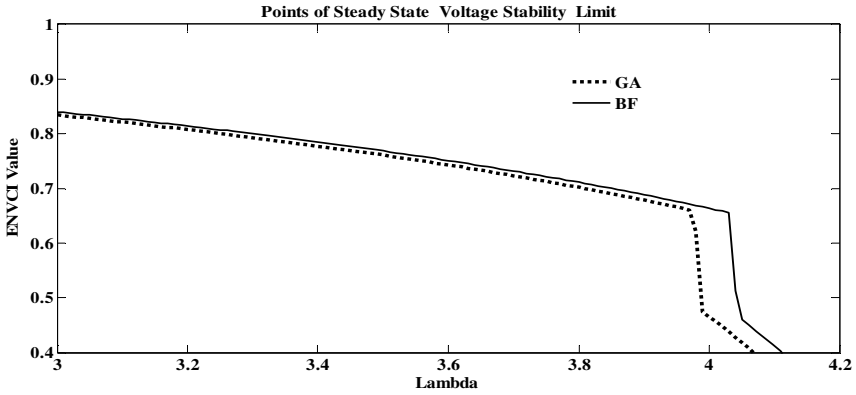


Fig. 6. Voltage Stability Limits for Case-2

7 Conclusion

In this work two effective locations and sizes of reactive power compensation was evaluated to improve the steady state VSL of 39-Bus New England system. To determine the VSL an equivalent network based system model is represented and it can be utilized for online implementation. The problem was formulated as a non linear optimization problem with a suitably designed objective function, which takes into account real power loss minimization in the system as well. The objective function is optimized by using GA and BFA. Results show the supremacy of BFA over GA not only in reducing the cost function but also in handling the desired constraints. To reduce the number of control variables, initially weak buses in the system are identified which helps in reducing the computational time.

References

1. Kundur, P.: Power System Stability and Control. McGraw-Hill (1994)
2. Feng, Z., Ajjarapu, V., Maratukulam, D.J.: A Comprehensive Approach for Preventive and Corrective Control to Mitigate Voltage Collapse. IEEE Trans. Power Systems 15(2), 791–797 (2000)
3. Alvarado, F.L., Meng, J., De Marco, C.L., Mota, W.S.: Stability Analysis of Interconnected Power Systems Coupled With Market Dynamics. IEEE Trans. Power Systems 16(4), 695–701 (2001)
4. Reddy, M.V., Pradeep, Y., Murthy Balijepalli, V.S.K., Khaparde, S.A., Dobaria, C.V.: Dobaria: Improvement of Voltage Stability Based on Static and Dynamic Criteria. In: 16th National Power Systems Conference, December 15-17, p. 715 (2010)
5. Ajjarapu, V., Christy, C.: The continuation power flow: A tool for steady state voltage stability analysis. IEEE Trans. Power Systems 7(1), 416–423
6. Verbič, G., Gubina, F.: A New Concept of Voltage-Collapse Protection Based on Local Phasors. IEEE Trans. Power Delivery 19(2) (April 2004)
7. Wang, Y., Li, W., Lu, J.: A new node voltage stability index based on local voltage phasors. Electrical Power System Research 79, 265–271 (2009)
8. Udgir, S., Varshney, S., Srivastava, L.: Optimal Placement and Sizing of SVC for Voltage Security Enhancement. International Journal of Computer Applications 32(6), 44–51 (2011)
9. Feng, Z., Ajjarapu, V., Maratukulam, D.J.: A Comprehensive Approach for Preventive and Corrective Control to Mitigate Voltage Collapse. IEEE Trans. Power Systems 15(2), 791–797 (2000)
10. Nanba, M., Huang, Y., Kai, T., Iwamoto, S.: Studies on VIPI based control methods for improving voltage stability. Electrical Power & Energy Systems 20(2), 141–146 (1998)
11. Yan, W., Yu, J., Yu, D.C., Bhattarai, K.: A New Optimal Reactive Power Flow Model in Rectangular Form and its Solution by Predictor Corrector Primal Dual Interior Point Method. IEEE Trans. Power Systems 21(1), 61–67 (2006)
12. Iba, K.: Reactive Power Optimization by Genetic Algorithm. IEEE Trans. Power Systems 9(2), 685–691 (1994)
13. Kowsalya, M., Ray, K.K., Kothari, D.P.: Loss Optimization for Voltage Stability Enhancement Incorporating UPFC Using Particle Swarm Optimization. Journal of Electrical Engineering & Technology 4, 492–498 (2009)

14. Tripathy, M., Mishra, S.: Bacteria Foraging based solution to optimize both real power loss and voltage stability limit. IEEE Trans. Power Systems 22(1), 240–248 (2007)
15. Passino, K.M.: Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control System, 52–67 (June 2002)
16. Thukaram, D., Lomi, A.: Selection of static VAR compensator location and size for systemvoltage stability improvement. Electric Power Systems Research 54, 139–150 (2000)

Optimal Placement of Capacitors in Distribution Networks Using a Modified Teaching-Learning Based Algorithm

Ankita Mohapatra[1], Bijaya Ketan Panigrahi[1], Bhim Singh[1], and Ramesh Bansal[2]

[1] Department of Electrical Engineering, Indian Institute of Technology, Delhi, India
[2] School of Information Tech. and Electrical Engg, University of Queensland, Australia

Abstract. This paper presents an efficient approach to determine the size of the capacitor bank and its placement in a radial distribution system. The objective of the above optimization is to place a proper size of the capacitor bank at a particular bus so as to reduce the power loss in the distribution system. Although the prime objective of the work in reduction of loss, during optimization other operational constraints of the system like, voltage limits at the distribution buses and the current limit of the lines have been considered. Teaching Learning Based Optimization (TLBO) algorithm is adopted in this work to find the optimal size of capacitors and its location in an existing radial distribution system. The proposed method is applied to 10 and 85-bus radial distribution systems and the obtained results are compared with other existing methods.

1 Introduction

Power loss in a distribution system is a major concern for the utilities. Along with the power loss, it is also equally important to maintain a consistent voltage profile in the distribution system within the prescribed voltage limit. Shunt capacitors are installed in the distribution system to serve various purposes, like power factor improvement, voltage profile improvement and to reduce the power loss. Hence in an existing system, this is a prime concern to find a proper site for the installation of the capacitor bank to achieve overall reduction in power loss and improvement of the voltage profile. Similarly, determining the optimal size of the capacitor bank is also a concern. Therefore, finding the optimal size of the capacitor bank amongst the available sizes and its placement at a specified bus or may be at different buses depends on the load scenario and load distribution along the whole radial distribution network.

The optimal capacitor sizing and placement becomes a combinatorial optimization problem. Many different optimization techniques have been applied successfully to solve this problem. Optimization techniques like nonlinear programming [1], dynamic programming [2], mixed integer programming [3] have been applied to solve this problem. Modern, evolutionary computing techniques have also been tried to solve the problem over the few years. Genetic Algorithm (GA) [4,5], Particle Swarm

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 398–405, 2012.

Optimization (PSO) [6], Differential Evolution (DE) [7], Ant colony optimization [8], Plant Growth Simulation Algorithm (PGSA) [9] have been applied to solve the capacitor sizing and placement problem. In this work we have adopted the recent developed Teaching Learning Based Optimization (TLBO) algorithm to solve the capacitor placement problem.

2 Capacitor Placement Problem Formulation

The objective of the capacitor placement problem is to minimize the annual cost of the distribution system. Since the capacitor placement causes a reduction in power losses, this leads to a gain to the distribution company. On the other hand the placement of the capacitor banks of appropriate size as available in different discrete sizes are having different cost. So the distribution company has to bear the installation cost of the capacitors and in return the benefit comes from the power loss reduction. The objective function is formulated to minimize the cost over a period of one year. As the power loss depends on the load pattern, we have assumed different load patterns for one year horizon. To make the problem formulation simpler three load patterns for different durations in a year has been assumed. The light load condition for 1000 hours, medium load for 6760 hours and heavy load for 1000 hours is assumed for the analysis purpose in a year [10]. We have considered only the fixed cost and the capacitor bank installation cost into account, neglecting the operation and maintenance cost and depreciation cost. Whereas these costs can be considered to make the problem more realistic and presently under investigation. The radial distribution system is considered to be balanced.

$$\text{Min Cost} = \mathrm{K_P}\, P_{Loss} + \sum_{\mathrm{j}=1}^{Nc} C_{FC} + K_{IC}^{j} Q_{FC}^{j} \tag{1}$$

The minimization of the total cost consists of two components. K_P is a proportionality factor, which represents the equivalent annual cost per unit of power loss expressed in \$/kWh. P_{Loss} is calculated for the radial distribution system for a period of one year through three different time varying load conditions. C_{FC} is the fixed cost for the capacitor placement. K_{IC}^{j} is the annual capacitor installation cost for the j^{th} capacitor bank and Q_{FC}^{j} is the j^{th} capacitor bank rating. N_c is the number of capacitor banks. It is assumed that a capacitor bank is installed at one bus. hence N_c is the number of candidate buses in the radial distribution system, where the capacitor bank is to be installed.

The above optimization is carried out subjected to the following operational constraints of the distribution system.

The bus voltage magnitude should be maintained between the limits V_{min} and V_{max}

$$V_{\min} \le |Vi| \le V_{\max}\,;\ \forall\, \mathrm{i} \in N_b \tag{2}$$

N_b is the number of bus in the system.

The current limit of branches

$$|I_{ij}| \leq I_{ij}^{max} \; ; \forall i, j \in N_b \tag{3}$$

I_{ij} is the current flow in the branch between bus i and bus j.

The total power loss in the radial distribution system is the summation of power losses in all the branches. The power loss in any branch 'k', between two buses 'i' and 'j' is given by

$$P_k = I_k^2 R_k$$

where, I_k is the current in branch 'k' and R_k is the resistance of branch 'k'.
The total power loss in the system is given as

$$P_L = \sum_{k=1}^{nl} P_k \tag{4}$$

where, 'nl' is the total number of lines in the radial network

3 Teaching Learning Based Optimization

The Teaching Learning Based Optimization algorithm mimics the behavior and performance of the student as well as teacher in a class. It is primarily based on the influence of a teacher (leader) in a class having a group of learners (students). The performance of the learners is completely influenced by the performance and quality of the teacher [11]. Like other nature-inspired algorithms, TLBO is also a population based method that uses a population of learners in a class. Each learner is considered as a probable solution of the problem. The teacher among the group of learners, is considered as the best individual among the population of learners. Hence this mimics the global best in PSO, and can be considered as the best solution obtained so far. The complete process of TLBO can be divided into two parts. The first part consists of the 'Teacher Phase' and the second part consists of the 'Learner Phase'. The 'Teacher Phase' means learning from the teacher and the 'Learner Phase' means learning through the interaction between learners [11].

3.1 Teacher Phase

The initial phase of the algorithm is to create a class or a group of learners in a class opting for various different subjects. So at the starting of the algorithm, a matrix of *N* rows and *D* columns is randomly generated in the search space. *N* represents the population size or the "class size" and *D* represents the number of "subjects or courses offered" in the class, which is analogous to the dimensionality of the problem considered. Each learner at k^{th} generation cycle is represented by

$$X_{(i)}^{k} = \left[x_{(i,1)}^{k}, x_{(i,2)}^{k}, \ldots, x_{(i,j)}^{k}, \ldots, x_{(i,D)}^{k} \right] \tag{5}$$

where $i = 1.....N$ and $j = 1.....D$ and each dimension of the learner is presented by

$$x_{(i,j)}^{k} = x_{j}^{\min} + rand_{(i,j)} \times \left(x_{j}^{\max} - x_{j}^{\min} \right) \tag{6}$$

where $rand_{(i,j)}$ represents a uniformly distributed random variable within the range (0,1).

In a group of learner, the best learner is considered as a teacher having maximum learning level, which in turn evaluates the best for the objective function. The teacher tries to share its knowledge among the other learners, which in turn will increase the knowledge level of the whole class and help learners to improve their performance. So a teacher increases the mean fitness of the class according to his or her capability. The teacher puts its effort to increase the quality of the students.

So, the mean vector containing the mean of the learners in the class for each subject is computed. The mean vector M^k at k^{th} iteration is given as

$$M^{k} = \left[\sum_{i=1}^{N} \left(\frac{x_{(i,1)}^{k}}{N} \right) \quad \sum_{i=1}^{N} \left(\frac{x_{(i,2)}^{k}}{N} \right) \quad \cdots \quad \sum_{i=1}^{N} \left(\frac{x_{(i,j)}^{k}}{N} \right) \quad \cdots \quad \sum_{i=1}^{N} \left(\frac{x_{(i,D)}^{k}}{N} \right) \right] \tag{7}$$

A new set of learners is obtained by

$$L_{(i)}^{k} = X_{(i)}^{k} + rand^{k} \left(Teacher^{k} - F.M^{k} \right) \tag{8}$$

where $Teacher^k$ is the vector with the maximum fitness value and F is a teaching factor which is randomly chosen as 1 or 2 for each iteration k. The better learners found in this phase replace the inferior ones in the population.

3.2 Learner Phase

Learners try to increase their knowledge, which is analogous to the increase in their performance for the evaluation of the objective function by two different ways. This enhancement in the knowledge level is achieved either through input from the teacher or through interaction between themselves. This step mimics the concept of particle swarm optimization as the learning through social interaction and self cognition. A learner interacts randomly with other learners A learner learns something new if the other randomly selected learner has more knowledge than him or her. In this algorithm we have proposed a new learner stage based on the mutation strategy adopted in Differential evolution algorithm. A new set of learners are produced in this stage using differential random vectors and also the difference vector with respect to the best vector at that point. Learner modification can be expressed as

$$L_{(i)}^{k} = X_{(i)}^{k} + F_1 \left(X_{(i)}^{k} - X_{(best)}^{k} \right) + F_2 \left(X_{(\alpha)}^{k} - X_{(\beta)}^{k} \right) \tag{9}$$

where F_1 and F_2 are constants which indicate the extent of contribution of the difference vectors to the learning process. The values α and β are random indices which are mutually exclusive and also different that the current index. The procedure of replacing the inferior solutions is repeated and the algorithm is continued till termination.

4 Case Study: Results and Discussion

In this section, the 85 bus radial distribution system has been considered for capacitor placement. The algorithm is implemented using MATLAB V7 on a PC Pentium IV, 2-GHz computer with 2 GB RAM.

The 85 bus radial distribution system with one supply point is a single feeder distribution system having no laterals and the system voltage is 11kV. Details of the feeder and the loads are adopted from [12]. The load data at different buses as reported in [9] and are assumed to be medium loading condition for the test case in our simulation. The second test case, consists of the same 85 bus system but with different loading conditions of various duration throughout a year. 80% of the loading is assumed as light loading and 120 % of the loading is assumed to be heavy loading condition for the simulation over one year for a time varying load pattern. For this test case , K_P is assumed to have a value of 168 $/ kWhr . The available capacitor sizes are in the discrete steps of 150 kVAR and the range of capacitors available are from 150 kVAR to 4050 kVAR. The cost coefficient K_{IC} for different capacitor bank size in multiple of 150kVAR, are reported in [9]. The fixed cost of the capacitor, C_{FC} is selected as 1000 $. Bus 1 is treated as the substation and the voltage at bus 1 is assumed as 1 p.u.. The voltage magnitude limit during the optimization is assumed to be in between 0.9 pu to 1.1 pu.

As the test case, the load throughout the year is assumed to be constant and treated as base case load (medium loading). The algorithm is run for this case and the results are reported in Table 1. It is observed that the proposed TLBO provides better result as reported in ref [13]. However the result by the proposed approach is more than that of reported result of [9] because of the discrete size of the capacitor bank and multiples of 150 kVAR. The percentage loss reduction is better than both the approaches as mentioned in ref [9] and [13].

We have also carried out the simulation assuming the same load pattern for a year and tried to run the algorithm for different number of capacitor placement. It is observed that minimum loss is obtained for 4 capacitor placement, but it increases the annual cost. Whereas 3 capacitor placement yields lowest annual cost. The results are reported in Table 2.

The third set of simulation is carried out for a time varying loading pattern as mentioned earlier. The results are reported in Table 3.

Table 1. Results for 85 bus radial distribution system, constant load throughout year, comparison

	Uncompensated	Compensated		
		PSO[13]	Plant Growth [9]	Proposed TLBO
Total hourly Loss (kW)	315.714	163.32	161.4	150.98
Loss reduction(%)		48.27	48.88	52.18
Capacitor Size With Bus No	–	8 796 58 453 7 324 27 901	8 1200 58 908 7 200	9 1200 34 600 68 450
Total installed kVAr	–	2473	2308	2005.7
Annual cost($/year)	53040	29051	28585	28816

Table 2. Results for 85 bus radial distribution system, with constant load throughout year, for 1-4 number(s) of capacitor placement

	Uncompensated	Compensated			
No of Cap.		1 cap.	2 cap.	3 cap.	4 cap.
Total yearly loss (MW)	2768.3	1581.8	1399.1	1322.6	1307.9
Loss reduction(%)		42.86	49.46	52.22	52.75
Capacitor size with bus no	--	8 2400	32 900 60 1050	9 1200 34 600 68 450	9 1200 31 450 48 300 68 450
Min Voltage	0.8713	0.9118	0.9164	0.92	0.9243
Max Voltage	0.9957	0.9973	0.9971	0.9973	0.9974
Annual Cost ($/year)	53091	31744	29237	28816	29620

Table 3. Results for 85 bus radial distribution system, with variable load for a year

	Uncompensated	Compensated		
		2 capacitor	3 capacitor	4 capacitor
Total yearly loss (MW)	2809.2	1684.4	1520.5	1470.7
Loss reduction (%)		40.04	45.87	47.65
Capacitor size with bus no	--	34 1500 59 1050	29 1200 53 450 63 900	32 900 53 450 60 750 80 450
Min Voltage (light load)	0.8999	0.9563	0.9604	0.9596
Max Voltage (light load)	0.9966	0.9983	0.9983	0.9983
Min Voltage (normal load)	0.8713	0.9343	0.9370	0.9372
Max Voltage(normal load)	0.9957	0.9974	0.9974	0.9974
Min Voltage (heavy load)	0.8409	0.9106	0.9120	0.9122
Max Voltage(heavy load)	0.9947	0.9966	0.9966	0.9966
Annual Cost ($/year)	53876	34845	32643	32805

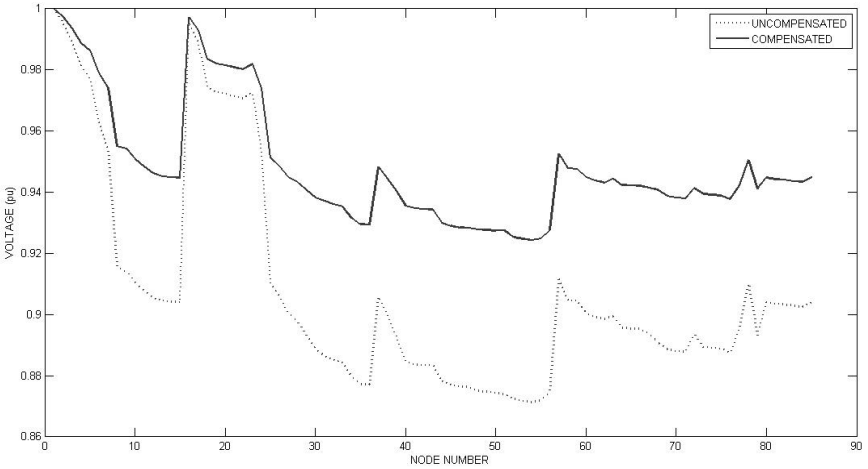


Fig. 1. Voltage profile comparison of uncompensated and 4 capacitor compensated 85 bus Radial Distribution System with constant load throughout the year

5 Conclusion

This paper presents a novel optimization algorithm with a modification to solve the capacitor sizing and placement problem in distribution networks. The random search procedure allows the TLBO to explore the whole search space enhancing the algorithm`s capability of reaching the optimal solution in a reasonable time.

References

1. Lee, S.H., Grainger, J.J.: Optimum size and location of shunt capacitors for reduction of losses on distribution feeders. IEEE Trans. Power Appartus System 100(3), 1105–1118 (1981)
2. Duran, H.: Optimum number, location and size of shunt capacitors in radial distribution feeders: a dynamic programming approach. IEEE Trans. Power Apparat. Syst. 87, 1769–1774 (1968)
3. Baran, M.E., Wu, F.F.: Optimal capacitor placement on radial distribution system. IEEE Trans. Power Deliv. 4(1), 725–734 (1989)
4. Boone, G., Chiang, H.-D.: Optimal capacitor placement in distribution systems by genetic algorithm. Electrical Power & Energy Systems 15(3), 155–162 (1993)
5. Sundhararajan, S., Pahwa, A.: Optimal selection of capacitors for radial distribution systems using a genetic algorithm. IEEE Trans. Power System 9, 1499–1507 (1994)
6. Prakash, K., Sydulu, M.: Particle swarm optimization based capacitor placement on radial distribution systems. In: IEEE Power Engineering Society General Meeting, pp. 1–5 (2007)
7. Chiou, J.-P., Chang, C.-F., Su, C.-T.: Capacitor placement in large-scale distribution systems using variable scaling hybrid differential evolution. Electrical Power and Energy Systems 28, 739–745 (2006)
8. Annaluru, R., Das, S., Pahwa, A.: Multi-level ant colony algorithm for optimal placement of capacitors in distribution systems. In: Congress on Evolutionary Computation, CEC 2004, June 19-23, vol. 2, pp. 1932–1937 (2004)
9. Srinivasas Rao, R., Narasimham, S.V.L., Ramalingaraju, M.: Optimal capacitor placement in a radial distribution system using Plant Growth Simulation Algorithm. Electrical Power and Energy Systems 33, 1133–1139 (2011)
10. Mohkami, H., Hooshmand, R., Khodabakhshian, A.: Fuzzy optimal placement of capacitors in the presence of nonlinear loads in unbalanced distribution networks using BF-PSO algorithm. Applied Soft Computing 11, 3634–3642 (2011)
11. Rao, R.V., Savsani, V.J., Vakharia, D.P.: Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. Computer Aided Design 43, 303–315 (2011)
12. Das, D., Kothari, D.P., Kalam, A.: Simple and efficient method for load flow solution of radial distribution network. Electrical Power Energy Syst. 17(5), 335–346 (1995)
13. Prakash, K., Sydulu, M.: Particle swarm optimization based capacitor placement on radial distribution systems. In: IEEE Power Engineering Society General Meeting, pp. 1–5 (2007)

Gene Expression Programming Algorithm for Transient Security Classification

Almoataz Y. Abdelaziz[1], S.F. Mekhamer[1], H.M. Khattab[2], M.L.A. Badr[1], and Bijaya Ketan Panigrahi[3]

[1] Department of Electrical Power & Machines, Faculty of Engineering, Ain Shams University, Cairo, Egypt
[2] Engineering for the Petroleum and Process Industries, (ENPPI), Cairo, Egypt
[3] Department of Electrical Engineering, Indian Institute of Technology, Delhi, India

Abstract. In this paper, a gene expression programming (GEP) based algorithm is implemented for power system transient security classification. The GEP algorithms as evolutionary algorithms for pattern classification have recently received attention for classification problems because they can perform global searches. The proposed methodology applies the GEP for the first time in transient security assessment and classification problems of power systems. The proposed algorithm is examined using different IEEE standard test systems. Power system three phase short circuit contingency has been used to test the proposed algorithm. The algorithm checks the static security status of the power system then classifies the transient security of the power system as secure or not secure. Performance of the algorithm is compared with other neural network based classification algorithms to show its superiority for transient security classification.

1 Introduction

The growth of large interconnected power systems demands a high degree of security during normal and abnormal operation. Power systems design and operation must ensure that all operating variables are controlled and fall within acceptable ranges at all operating conditions. Power system security main goal is to ensure proper system operation within acceptable limits. Conversely, failure of power system to operate securely can lead to outages which may have wide ranging consequences on power system, users and equipment including power interruption to customers, damage of equipment, huge financial losses, or even loss of life. Technical and economic outcomes can be maintained by ensuring power system security [1, 2].

With the recent advent of the deregulated electricity market, modern utilities have been forced to operate power systems closer to their security boundaries. This trend has encouraged the need for fast and accurate assessment of power system security [3]. Constraints are imposed on power system equipment, for instance each generator has limited active and reactive power capacity, each line or transformer has limits on the flow through it. Constraints are also imposed on power system, bus voltage magnitude levels have to be within acceptable limits, bus voltages angles have limits

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 406–416, 2012.

across the buses for stability. The power system is considered to be in the normal operating state when all the above constraints are satisfied and fulfilled. It is the main role and purpose of power system security to maintain system operation in the normal state [4].

Power system security assessment is classified into transient security, dynamic security and static security [5-7].

In order to assess system static and transient security, conventional analytical methods have been applied. They demand a huge amount of computation in order to solve the power system equations, simulate each contingency. In order to overcome their huge computational requirements, applications based on other types of techniques have evolved. Application of artificial intelligence based algorithms like decision trees, pattern recognition, fuzzy logic, artificial neural networks and expert systems have been explored for static and transient security assessment problems [8, 9].

The authors in reference [10] presented a probabilistic neural network (PNN) based classifier to judge the static security of the power system. The proposed classifier classifies the security of the power system based on the voltage profile of each bus in reference to changes in the generation and load profile in the system. The probabilistic neural network is used and compared with the radial basis function neural network (RBFNN) and the backpropagation neural network (BPNN) with superior results. The proposed methodology has been examined using three IEEE standard test systems, where the input to the neural network is the voltage profile at each bus, the output of the PNN classifies the security of the power system into three classes, normal, alert and emergency.

Power system transient stability classification has been investigated using support vector machine in [11], the transient security evaluation problem is presented as a two class pattern recognition problem. In [12], transient stability assessment of a large actual 87-bus system and the IEEE 39-bus system using the probabilistic neural network (PNN) with enhanced feature selection and extraction methods is presented. The investigated power systems are divided into smaller areas depending on the coherency of the areas when subjected to disturbances. An enhanced feature selection and extraction methods are then incorporated to reduce the input features to the PNN used as a classifier to determine whether the power system is stable or not. It is concluded that the PNN with enhanced feature selection and extraction methods reduces the time taken to train the PNN without affecting the accuracy of the classification results.

Reference [13] presents the application of different neural network models for classifying the power system states as secure/insecure. Pattern recognition approach is recognized as the classification tool. The neural network models adopted for classification includes multilayer perceptron, learning vector quantization, probabilistic neural network and adaptive resonance theory mapping. The neural network models designed are tested on IEEE standard test systems. The performance of various neural network models are studied in training and testing phases and the results are compared. A methodology to analyze transient stability for electric energy systems using artificial neural networks based on fuzzy ARTMAP architecture is presented in [14].

Gene expression programming (GEP) presented in [15] is a new technique of evolutionary algorithm for data analysis. GEP uses fixed length, linear strings of

chromosomes to represent computer programs in the form of expression trees of different shapes and sizes, and implements a genetic algorithm to find the best program. GEP combines the advantages of both genetic algorithm (GA) and genetic programming (GP), while overcoming some of their individual limitations. GEP has been used for solving classification problems of power systems since its evolution.

In this paper, a novel GEP based classification algorithm is used for transient security of power system; the GEP based algorithm checks the system static security then classifies the statically secured status for transient security. Results are compared to different neural network based classifiers. The output of each algorithm classifies the transient security of the power system under study into two classes secure or insecure. The designed models have been applied 30-bus and 57-bus IEEE standard test systems and the classification results are compared to show superiority of the proposed GEP approach.

2 Gene Expression Programming (GEP)

GEP belongs to the Evolutionary Algorithms (EA's) and uses populations of individuals (models or solutions), selects and reproduces them according to fitness, and introduces genetic variation using one or more genetic operators thus creating the next generation of new models. By repeating this process for a certain number of generations, one is bound to get evolution, which in this case means better solutions to achieve the best solution.

Of special interest to the GEP technique are the Genetic Algorithms (GAs) and Genetic Programming (GP), for they serve to illustrate some of the fundamental characteristics of the GEP technique and why GEP surpasses the old GP technique.

First of all, both GAs and GP are simple replicator systems, with the latter considerably more complex than the former. The GEP system is a full-fledged genotype/phenotype system with expression trees of different sizes and shapes encoded in linear chromosomes of fixed length.

- **The Architecture of GEP Programs**

There are two main players in GEP; the chromosomes and the expression trees (ETs). The structural organization of GEP genes is better understood in terms of open reading frames (ORFs).

Consider, for example, the algebraic expression:

$$\sqrt{(a-b)(c+d)} \tag{1}$$

It can also be represented as a diagram or an expression tree of Fig. (1):
Where “Q” represents the square root function.

This kind of diagram representation is what is called the phenotype in gene expression programming. And the genotype can be easily inferred from the phenotype as follows:

$$\begin{array}{l} 01234567 \\ Q*-+abcd \end{array} \tag{2}$$

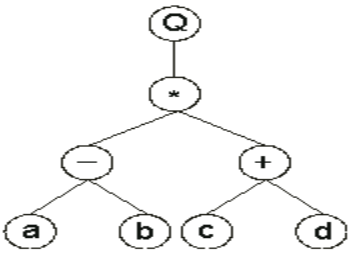


Fig. 1. Tree presentation for equation

The expression (2) is an ORF, starting at "Q" (position 0) and terminating at "d" (position 7). These ORFs were named K-expressions from Karva notation. By looking at the structure of GEP K-expressions, it is difficult or even impossible to see the advantages of such a representation, except perhaps for its simplicity and elegance. However, when K-expressions are analyzed in the context of a gene, the advantages of this representation become obvious. GEP chromosomes have fixed length, and they are composed of one or more genes of equal length.

Thus, in GEP, what varies is not the length of genes which is constant, but the length of the K-expressions. In the former case, the termination point coincides with the end of the gene, and in the latter, the termination point is somewhere upstream of the end of the gene. The genes of GEP are composed of a head and a tail. The head contains symbols that represent both functions and terminals, whereas the tail contains only terminals. For each problem, the length of the head h is chosen, whereas the length of the tail t is a function of h and the number of arguments n of the function with more arguments and is evaluated by the equation:

$$t = h(n-1)+1 \tag{3}$$

In GEP, chromosomes are usually composed of more than one gene of equal length. For each problem or run, the number of genes, as well as the length of the head, is chosen.

3 Transient Security Assessment

Transient security is the ability of a power system to operate consistently within the limits imposed by the stability phenomena. A set of most probable contingencies needs to be first specified for security evaluation. This set may include single line outage, generator outage, sudden increase in load, a three phase fault in the system. In this paper, the system operating condition is disturbed and the static security status is evaluated. Each operating condition declared as static secure is then tested for transient security status by applying three phase faults on all transmission lines, one at a time and monitoring the rotor oscillations using time domain simulation (TDS). Transient security assessment consists of determining whether the system oscillations, following the occurrence of a fault or a large disturbance from a contingency set, will cause loss of synchronism among the system generators. The goal of transient security assessment (TSA) is to solve equations describing the transient behavior of the system under a set of credible contingencies. The motion of the generators is governed by a set of non-linear differential equations called 'Swing Equation' given by:

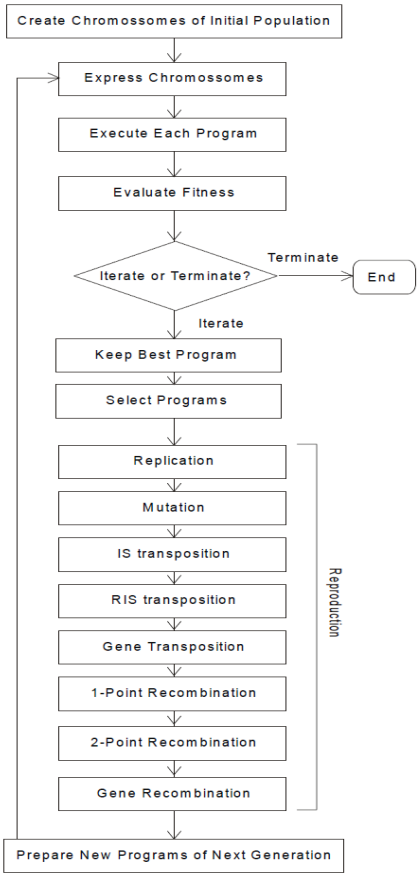


Fig. 2. Flowchart of gene expression programming

$$M_i \frac{d^2\delta_i}{dt^2} = P_{mi} - P_{ei} \; ; \tag{4}$$

$$\frac{d\delta_i}{dt} = \omega_i - \omega_s \; ; i - 1,2,3,..n_G \tag{5}$$

The expression for Pei is given by:

$$P_{ei} = E_i^2 G_{ii} + \sum_{\substack{j=1 \\ j \neq i}}^{N_G} \left[C_{ij} \sin\delta_{ij} + D_{ij} \cos\delta_{ij} \right] \tag{6}$$

where, $C_{ij} = E_i E_j B_{ij}$; $D_{ij} = E_i E_j G_{ij}$; E_i is the internal voltage of i^{th} machine. The network parameters G_{ii}, C_{ij}, D_{ij} have different numerical values, depending on the fault. If the fault is cleared at $t = t_c$, then we have:

$$M_i \frac{d^2\delta_i}{dt^2} = P_{mi} - P_{ei0}^f \; ; 0 < t < t_c \tag{7}$$

$$M_i \frac{d^2\delta_i}{dt^2} = P_{mi} - P_{ei_c}^{pf}; t \geq t_c \tag{8}$$

where, subscripts f and pf denote the faulted and post-fault state respectively [11].

4 Implementation of Classification Algorithms

The GEP algorithm and different neural network models have been applied to judge the transient security state of different test systems previously judged for static security by the same authors in [16]. The different classifier algorithms have been applied to the IEEE 30-bus and 57-bus test systems. Fig. (3) shows a flowchart for the application of the different algorithms to the test systems. Each static secure case is subjected to transient security analysis by simulating a set of transient disturbances of three phase fault on all lines, one at a time. The faults are applied at 0 sec and cleared after 15 cycles by opening the faulted line. The numerical integration technique adopted for solving system dynamic equations is the 4th order Runge Kutta method.

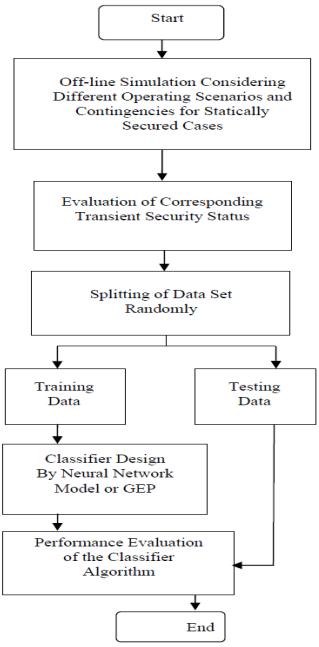


Fig. 3. Flowchart for application of different classifiers models

A. Classification Algorithm Architecture

Classification Algorithm Input

The system is first assessed for static security, for load variation contingencies, the input consists of the values of the voltage magnitude of the system buses. For single and double line outages contingencies, line status in service (binary "1") or line

outage/out of service (binary "0") is the input to the algorithm [16]. The statically secured cases are subjected to another classifier algorithm using the three phase fault contingency case on each system line to judge the system transient security.

Classification Algorithm Output
The output of the algorithm determines the transient security state of the system; 1 (Transiently secure state) and 0 (Transiently insecure state).

B. Data Generation

B1. First Test Case (Load Variation Contingencies)
The real time security assessment requires continuous monitoring of the input signals. Normalized voltage level of the buses in the system is fed to the classification algorithm. The application of classification algorithm for classifying the patterns requires considerable amount of data for training and testing of the network. The training and testing data for the present work are generated by varying the active and reactive load at the buses. The active power generated at the generator bus is incremented with reference to changes in the active load. Three different data patterns are generated for training the algorithm by running the load flow as follows:

- Increments of only active load at all the buses.
- Increments of only reactive load at all the buses.
- Simultaneous increments of active and reactive load at all the buses.

The statically secured cases are then subjected to another transient security classifier algorithm using the three phase fault contingency case on each system line to judge the system transient security.

B2. Second Test Case (Line Outage Contingencies)
The status of the lines in each power system line topology is fed to the classification algorithm as an input pattern for static security assessment. A line outage/out of service is input as "0" while a line in service is input as "1", this binary coding for the line status simplifies the input and accordingly enhances the classifier performance. Single and double line outage contingencies have been studied. The statically secured cases are then subjected to transient security assessment using three phase fault simulation at each of the system transmission lines.

C. Performance Evaluation of the Classifier
Performance of the static security assessment is found in [16]. The performance of the different transient security classifiers are judged by evaluating measures similar to those of [13] after implementing required modifications to account for the problem under study.

Mean Squared Error (MSE)

$$MSE = \frac{1}{n}\sum_{k=1}^{n}(E_k)^2; E_k = | DO_k - AO_k | \tag{9}$$

Where n is equal to the number of samples in the data set, DOk is the desired output obtained from off-line simulations and AOk is the actual output obtained from the trained classifier.

Classification Accuracy (CA)

$$CA = \frac{No.of\ samples\ classified\ correctly}{Total\ number\ of\ samples\ in\ data\ set} \tag{10}$$

Misclassification Rate (MC)

(i) Secure Misclassification (SMC)

$$SMC = \frac{No.of\ samples\ classified\ as\ 0}{Total\ number\ of\ \text{sec}ure\ states} \tag{11}$$

(ii) Insecure Misclassification (IMC)

$$IMC = \frac{No.of\ samples\ classified\ as\ 1}{Total\ number\ of\ un\text{sec}ure\ states} \tag{12}$$

5 Simulation Results

First Test Case (Load Variation Contingencies)

The classifiers for transient security assessment were trained using a group of loading patterns, then tested using group of loading patterns they have never seen. The BPNN gave best results with 23 hidden nodes and was trained using the standard gradient descent algorithm. Learning rate of BPNN was assumed to be 0.15 and momentum constant was taken as 0.96. A spread constant of 1.23 was chosen for the RBFNN. A smoothing parameter of 0.67 was chosen for the PNN. GEP parameters (number of chromosomes 30, head size 8, mutation rate 0.048, inversion 0.13, one-point recombination 0.36, two-point recombination 0.32). Table (1) shows the data generation for the transient security assessment. Operating scenarios have been chosen to cover a wide range of operating conditions in order to adequately present all possible configurations. Time domain simulation results were fed to the different classifiers models after splitting the data randomly into training set and test set. Tables (2) and (3) show the classification results. It is shown that the GEP is very superior in classification better than other classifier models. The GEP and PNN classifier algorithms yield zero error or 100% accuracy for classifying the data patterns they have been previously trained to, other neural network models fail to achieve this property. Moreover, in terms of test set data the GEP provides the best results in terms of least error, best accuracy classification and least misclassification.

Table 1. Data generation results

Test Case	IEEE 30 Bus	IEEE 57 Bus
Operating Scenarios	29440	66560
Secure Cases	10080	28450
Insecure Cases	19360	38110

Table 2. Classification results of transient security assessment for the IEEE 30-bus test system

	TRAIN SET				TEST SET			
Samples	**BPNN**	**RBFNN**	**PNN**	**GEP**	**BPNN**	**RBFNN**	**PNN**	**GEP**
MSE%	01.55	01.23	0.00	0.00	02.31	02.21	01.41	01.03
CA%	81.21	95.24	100.00	100.00	80.31	90.64	94.05	95.38
SMC%	25.34	21.67	00.00	00.00	50.37	40.39	06.97	04.27
IMC%	18.41	12.04	00.00	00.00	20.87	13.58	05.80	05.74

Table 3. Classification results of transient security assessment for the IEEE 57-bus test system

	TRAIN SET				TEST SET			
Samples	**BPNN**	**RBFNN**	**PNN**	**GEP**	**BPNN**	**RBFNN**	**PNN**	**GEP**
MSE%	01.75	01.98	0.00	0.00	03.25	02.55	01.64	01.32
CA%	77.65	90.62	100.00	100.00	71.50	85.54	89.84	92.60
SMC%	32.70	28.14	00.00	00.00	52.54	42.89	06.50	06.07
IMC%	25.56	20.68	00.00	00.00	25.87	21.36	06.47	05.55

Second Test Case (Line Outage Contingencies)

The secure states of [16] were then checked for transient security. The BPNN gave best results with 21 hidden nodes and was trained using the standard gradient descent algorithm. Learning rate of BPNN was assumed to be 0.12 and momentum constant was taken as 0.95. A spread constant of 1.13 was chosen for the RBFNN. A smoothing parameter of 0.60 was chosen for the PNN. GEP parameters (number of chromosomes 36, head size 10, mutation rate 0.051, inversion 0.14, one-point recombination 0.34, two-point recombination 0.23). Table (4) shows the data generation. Tables (5) and (6) compare the classification results in terms of performance

Table 4. Data generation results

Test Case	**IEEE 30 Bus**	**IEEE 57 Bus**
Operating Scenarios	10240	84320
Secure Cases	3368	39587
Insecure Cases	6872	44733

Table 5. Classification results of transient security assessment for the IEEE 30-bus test system

	TRAIN SET				TEST SET			
Samples	**BPNN**	**RBFNN**	**PNN**	**GEP**	**BPNN**	**RBFNN**	**PNN**	**GEP**
MSE%	08.71	03.25	0.00	0.00	08.64	05.33	02.54	01.90
CA%	78.31	84.70	100.00	100.00	75.21	80.87	90.25	91.36
SMC%	26.56	11.24	00.00	00.00	29.28	17.31	09.20	08.46
IMC%	16.24	08.39	00.00	00.00	20.97	09.14	08.38	07.15

Table 6. Classification results of transient security assessment for the IEEE 57-bus test system

	TRAIN SET				TEST SET			
Samples	**BPNN**	**RBFNN**	**PNN**	**GEP**	**BPNN**	**RBFNN**	**PNN**	**GEP**
MSE%	03.65	03.36	0.00	0.00	05.69	04.64	03.25	02.98
CA%	85.19	87.50	100.00	100.00	81.25	83.29	90.98	92.88
SMC%	17.30	14.66	00.00	00.00	26.65	19.54	12.30	10.47
IMC%	15.95	13.25	00.00	00.00	19.20	18.36	15.36	08.68

measures which show the leading performance of the GEP and PNN algorithms, also yield zero error or supreme accuracy for classifying the test data patterns, this advantage is not achieved by other neural network models. Performance is guaranteed with increasing the test patterns.

6 Conclusion

This paper presented a novel classifier based on GEP algorithm used for the first time in literature for transient security of power systems. The algorithm classifies the transient security of the power system which is proofed to be statically secure. For each statically secured state the system is subject to a three phase fault on one each of the system lines. The GEP classification algorithm is compared to the probabilistic, radial basis function and the back-propagation neural networks. The GEP algorithm shows superior results in comparison to other techniques. The GEP and PNN algorithms solely has the intrinsic property of achieving zero error on patterns they have been trained to, which compels classification accuracy of 100%. It is clearly evident that the GEP classification algorithm is leading in transient security assessment of power system and very promising for application in future assessment researches.

References

[1] Stott, B., Alsac, O., Monticelli, A.J.: Security Analysis and Optimization. Proceedings of the IEEE 75(12), 1623–1644 (1987)

[2] Moshref, A., Provencher, M., Sun, J.: An Intelligent System for Advanced Dynamic Security Assessment. In: Proceedings of the IEEE PowerCon 2002, International Conference on Power System Technology, October 13-17, vol. 1, pp. 220–224 (2002)

[3] Santo, M.D., Vaccaro, A.: A Distributed Architecture for Online Power System Security Analysis. IEEE Trans. on Industrial Electronics 51(6), 1238–1248 (2004)

[4] Pang, C., Prabhakara, F., Al-Abiad, A., Koivo, A.: Security Evaluation in Power Systems Using Pattern Recognition. IEEE Trans. on PAS 93(2), 969–976 (1974)

[5] Sterpu, S., Lu, W., Basenger, Y., Hadjisaid, N.: Power System Security Analysis. In: Proceedings of the IEEE Power Engineering Society General Meeting (2006)

[6] Bizjak, G., Kerin, U., Kerbs, S.R., Lerch, E., Ruhle, O.: Vision 2020 Dynamic Security Assessment in Real time Environment. Proceedings of the IEEE (2008)

[7] Mansour, Y., Vaahedi, E., El-Sharkawi, M.A.: Large Scale Dynamic Security Screening and Ranking using Neural Networks. IEEE Trans. on Power Systems 12(2), 954–958 (1997)
[8] Wardwick, K., Ekwue, A., Aggarwal, A.: Artificial Intelligence Techniques in Power Systems. IEE, London (1997)
[9] Bansal, R.C.: Overview and Literature Survey of Artificial Neural Networks Applications to Power Systems (1992-2004). IE (I) Journal-EL 86, 282–296 (2006)
[10] Khattab, H.M., Abdelaziz, A.Y., Mekhamer, S.F., Badr, M.A.L.: Static Security Assessment using a Probabilistic Neural Network Based Classifier. The Online Journal on Electronics and Electrical Engineering (OJEEE) 3(4), 454–461 (2011)
[11] Kalyani, S., Swarp, K.S.: Transient Security Assessment and Classification using Support Vector Machine. Journal of Electrical Systems (2009)
[12] Abdul Wahab, N.I., Mohamed, A.: Transient Stability Assessment of Power Systems Using Probabilistic Neural Network with Enhanced Feature Selection and Extraction. International Journal on Electrical Engineering and Informatics 1(2), 103–114 (2009)
[13] Kalyani, S., Swarp, K.S.: Study of Neural Network Models for Security Assessment in Power Systems. International Journal of Research and Reviews in Applied Sciences 1(2), 104–117 (2009)
[14] Silveria, M.C., Lutofo, A.D., Minussi, C.R.: Transient Stability Analysis of Electrical Power Systems Using a Neural Network Based on Fuzzy ARTMAP. In: IEEE Bologna Power Tech. Conference, Bologna, Italy, June 23-26 (2003)
[15] Ferreira, C.: Gene Expression Programming: A New Adaptive Algorithm for Solving Problems. Complex Systems 13(2), 87–129 (2001)
[16] Abdelaziz, A.Y., Mekhamer, S.F., Badr, M.A.L., Khattab, H.M.: Probabilistic Neural Network Classifier for Static Voltage Security Assessment of Power Systems. Electric Power Components and System 40(2), 147–160 (2012)

PSO-Tuned Control Parameter in Differential Evolution Algorithm

Tapas Si[1], Nanda Dulal Jana[2], and Jaya Sil[3]

[1] Department of Computer Science and Engineering,
Bankura Unnayani Institute of Engineering,Bankura,West Bengal,India
[2] Department of Information Technology,National Institute of Technology, Durgapur,
West Bengal, India
[3] Department of Computer Science and Technology, BESU, West Bengal, India
{c2.tapas,nanda.jana}@gmail.com,
js@cs.becs.ac.in

Abstract. In this work, a method to control the parameters of Differential Evolution (DE) algorithm is proposed. Here the control parameters of DE are co-evolved by Particle Swarm Optimization (PSO) algorithm. The classical DE algorithm has two main control parameters: **Scale Factor (F)** and **Cross-over Rate (CR).** These are selected on *trial-and-error* basis for solving optimization problems. Several optimization problems lead to optimal or sub-optimal solution by proper selection of control parameters of the DE algorithm. In this proposed method, PSO algorithm is used to tune the scale factor and cross-over rate in DE algorithm. Basically PSO algorithm is used as a meta-optimizer for DE algorithm. The proposed method is termed as **mPSO-DE** in this paper. The mPSO-DE algorithm is applied on 12 benchmark unconstrained optimization problems. The obtained results are compared with that of classical DE algorithm. From the experimental studies, it has been found that the proposed mPSO-DE algorithm performed better than DE algorithm.

1 Introduction

The Differential Evolutionary (DE) algorithm is proposed by Storn and Price [4].It is real value encoded population based global optimization algorithm having stochastic nature. It has three control parameters: (a)***scale factor (F)*** (b)***cross-over rate (CR)*** and (c) ***population size***. In classical DE algorithm, all these parameters are kept fixed during search process. The DE algorithm suffers from stagnation and premature convergence. The premature convergence arises due to lack in diversity for multi-modal function having too many local optima. Therefore, DE algorithm's effectiveness, efficiency and robustness depend on these parameters. There are three control strategies for these parameters: (a) Deterministic [1,3](b) Adaptive [6-8](c) Self-Adaptive approaches [9-12]. In deterministic control strategy, parameters are chosen on

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 417–424, 2012.

trial-and-error basis and it is time consuming task. In adaptive control strategies, parameters are dynamically adjusted by getting feedback from the search process. In Self-adaptive strategies, parameters are encoded at the end of the individual vectors and co-evolved with the individual vectors. Brest *et al.*[8] proposed a self-adaptive DE algorithm in which each individual is extended with scale factor and cross-over rate. They used two new parameters τ_1 and τ_2 to control the evolution of *F* and *CR*. τ_1 and τ_2 are used to control the re-initialization of *F* and *CR* for each individual. Qin *et al.*[9] proposed a self-adaptive (SaDE) algorithm in which suitable learning strategy and parameter settings are self-adapted during evolution process. Yang *et al.*[12] introduced a self-adaptive differential evolution algorithm with neighborhood search (SaNSDE).

In this work, the main objective is to evolve the control parameters scale factor and cross-over rate for each individual using PSO algorithm in swarm space.

The organization of this paper is as follows: section 2 and section 3 describe the classical DE and PSO algorithm respectively. The proposed algorithm with flow chart is presented in Section 4. The experimental studies are carried out in Section 5. Finally a conclusion including future work is given in Section 6.

2 Differential Evolution Algorithm

In Classical DE [3] algorithm, a donor vector for an individual is created by perturbing a vector by the weighted difference of any other two mutual exclusive randomly selected vectors. Then the cross-over is performed between an individual and its donor vector to create a trial vector and finally if the trial solution is better than the current individual then it is replaced by the trial vector. In the DE/best/1/bin algorithm, the mutation is done by perturbing current best using weighted the difference of two mutually exclusive random vectors.

$$V_i = X_{best} + F.\ (X_{r1} - X_{r2})$$

Where V_i is the donor vector and X_{best} is the current best solution, X_{r1} and X_{r2} are two other vectors where r1≠ r2≠i. here i is the index of current individual vector. The binomial crossover is performed using the following:

```
for j=1:D
if rand(0,1)< CR || J_rand == j
Ui(j)=Vi(j)
else
Ui(j)=Xi(j)
End
```

Here U_i is the trial vector, D is the dimension of the problem, CR is the cross-over rate. J_{rand} is random index in [1, D].

3 Particle Swarm Optimization

PSO [5] is a kind of algorithm to search for the best solution by simulating the movement of flocking birds. PSO algorithms use a population of individual called particles. Each particle has its own position and velocity to move around the search space. Particles have memory and each particle keep track of previous best position and corresponding fitness. The previous best value is called as *pbest*. Thus *pbest* is related only to a particular particle. It also has another value called gbest, which is the best value of all the particles *pbest* in the swarm. The basic concept of PSO technique lies in accelerating each particle towards its *pbest* and the locations at each time step. Global best solution *gbest* is the best of the *pbest* of all particles in the swarm space.

$$V_{ij} = w \times V_{ij} + C1 \times R1 \times \left(X_{pbest} - X_{ij}\right) + C2 \times R2 \times \left(X_{gbest} - X_{ij}\right) \tag{7}$$

$$X_{ij} = X_{ij} + V_{ij} \tag{8}$$

C1 and C2 are personal and social cognizance of particles respectively and R1 and R2 are two uniformly distributed random numbers in the interval (0, 1). *w* is the inertia weight.

4 Proposed Algorithm

In the proposed algorithm, PSO co-evolves the scale factor and cross-over rate of DE algorithm. These two parameters are evolved in swarm space in parallel with DE algorithm. The particles are represented as $X_i = <F_i, CR_i>$ in swarm space and dimension of particle is 2. The particle X_i (F_i, CR_i) represents the parameter set for i^{th} DE individual. The F_i is used to scale the difference of two mutually exclusive randomly selected individuals in mutation operation and CR_i is used in cross-over between i^{th} individual with its donor vector. In this work, PSO is used to tune the parameters of a DE version DE/best/1/bin. F_i and CR_i are initialized in the range (0, 2) and (0, 1) respectively. In the co-evolution process, when these parameters violate those boundary conditions, they are reinitialized in the range (0.1, 2) and (0.1, 1). F_i and CR_i are obtained from particle X_i before mutation and cross-over operation. The flowchart of mPSO-DE algorithm is given in the Fig. 1.

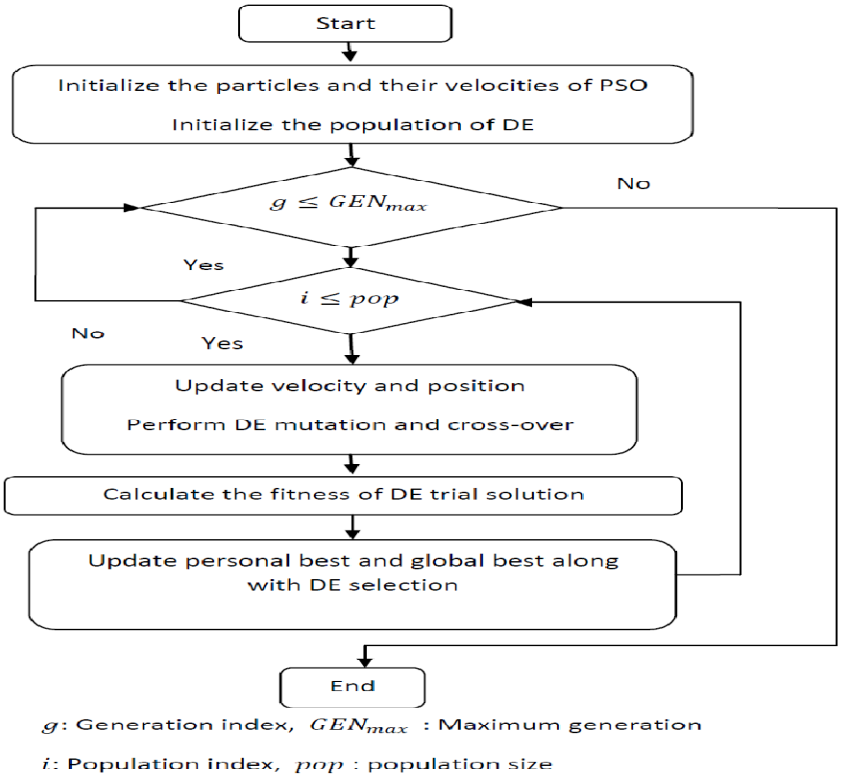


Fig. 1. Flow chart of mPSO-DE algorithm

5 Experimental Studies

5.1 Benchmark Functions

There are 12 different global optimization problems, including 6 uni-modal functions (f_1-f_4) and (f_9, f_{11}), 4 multi-modal functions (f_5-f_8), one discontinuous step function f_{10} and one quadratic noisy function f_{12} chosen in our experimental studies. These functions were used in an early study by X. Yao et al. [14].The function f_4 has platue like region. The function f_5 is highly multimodal i.e. it has too many local optima. The functions (f_6-f_8) have few local optima. The description of these benchmark functions and their global optima are given in Table 1, where D is the dimension of the problem, $f_{min}$is the minimum values of the functions, and $S \subseteq R^D$ in the search space.

Table 1. The 12 benchmark functions

Test Function	S	fmin
$f_1(x) = \sum_{i=1}^{D} x_i^2$	[-100,100]	0
$f_2(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	[-100,100]	0
$f_3(x) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{n-1}} x_i^2$	[-100,100]	0
$f_4(x) = [100(x_{i+1} - x_i{}^2)^2 - (1 - x_i^2)^2]$	[-100,100]	
$f_5(x) = -x_i * \sin(\sqrt{\lvert x_i \rvert})$	[-500,500]	-12569.50
$f_6(x) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1$	[-600,600]	0
$f_7(x) = -20 * \exp\left(-0.2 * \sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D} \cos(2\pi x_i)\right) + 20 + e$	[-32,32]	0
$f_8(x) = \sum_{i=1}^{D} [x_i - 10\cos(2\pi x_i) + 10]$	[-5.12,5.12]	0
$f_9(x) = max_i(x_i, 1 \le i \le D)$	[-100,100]	0
$f_{10}(x) = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$	[-100,100]	0
$f_{11}(x) = \sum_{i=1}^{D} x_i + \prod_{i=1}^{D} x_i$	[-10,10]	0
$f_{12}(x) = \sum_{i=1}^{D} i x_i^4 + random[0,1)$	[-1.28,1.28]	0

5.2 Parameter Settings

In this work, experiment is carried out for 12 benchmarks problems having dimension 30. Population size=150 for both DE and mPSO-DE. For DE, scale factor (F)=0.5 and cross-over rate (CR) =0.9.For mPSO-DE, the swarm size is same as population size in DE. Inertia weight (W) =0.729, C1=C2= 1.49445,Vmax=0.5x (Xmax-Xmin) where [Xmin, Xmax] is the search space range. The same population is used in a single run for DE and mPSO-DE algorithms.In this work, the following termination criteria are considered:

1. Maximum number of function evaluations (FES=3, 00,000)
2. $E = | f(X) - f(X^*) | <= e$ where f(X) is the current best and $f(X^*)$ is the global optimum is the best-error of a run of the algorithm and e is the threshold error.

e=0.001.

Total 30 runs are carried out for each problems considered in this work.

5.3 PC Configuration

1. System: Fedora 17
2. CPU: AMD FX -8150 Eight-Core
3. RAM: 4 GB
4. Software: Matlab 2010b

5.4 Results and Discussions

The best, mean and std. dev of run-error (absolute value of the difference between algorithm's best and global optimum) are given in Table 2. The Success Rate (SR) is the ratio of achieving threshold error and number of runs. The mean, std. dev. of function evaluations and success rate are given in Table 3. The convergence graph for function f_5 is given in Fig. 2. To compare the performance of DE and mPSO-DE algorithms with statistical significance, t-test [13] has been carried out for sample size (number of runs) = 30 and degrees of freedom=98. The last column of Table 2 signifies whether the null hypothesis that the means of the two data are equal is accepted or rejected. The value "0" indicates that the null hypothesis is accepted and the value "1" indicates that the null hypothesis is rejected. From the Table 2, it is seen that the mPSO-DE performed better than DE for functions f_5, f_7, f_8, f_{10} and f_{12}. There is no significance difference in performance between DE and mPSO-DE for remaining functions. Function f_5 is the highly multi-modal function having too many local optima. mPSO-DE performed better but there is no improvement in success rate and convergence speed. For function f_6, mPSO-DE has higher success rate and higher convergence speed though there is no difference in the obtained solution. For function f_7, mPSO-DE produced better solution but no improvement in success rate as well as convergence speed. For function f_8, mPSO-DE has better solution,higher success rate and higher convergence speed. f_{10} is a step function and mPSO-DE performed better than DE. mPSO-DE performed better than DE for noisy function f_{12} but no improvement in success rate as well as convergence speed. From the obtained results, it can be said that there is no difference in performance for uni-modal functions (f_1-f_4), $f_9$and f_{11}.But overall performance of mPSO-DE is better than DE algorithm.

Table 2. Run-error values achieved using DE and mPSO-DE

Problems	DE			mPSO-DE			h
	Best	Mean	Std. Dev.	Best	Mean	Std. Dev.	
f1	0.000543	0.000755	0.000126	0.000434	0.000792	0.000128	0
f2	2.69E-06	0.000399	0.000298	8.21E-07	0.000329	0.000278	0
f3	0.000426	0.000784	0.00014	0.000438	0.00083	0.000134	0
f4	0.000688	1.063765	1.792681	0.000707	1.595198	1.985971	0
f5	2487.255	4068.831	619.9052	136.5442	1394.927	1054.64	1
f6	0.00079	0.029249	0.025806	0.000634	0.019254	0.022284	0
f7	1.155149	2.713141	1.226568	1.155149	4.550661	4.52532	1
f8	33.82859	70.01176	19.33262	0.000952	42.78322	25.20573	1
f9	0.000907	0.000975	2.44E-05	0.000942	0.000983	1.47E-05	0
f10	6	19.33333	10.97437	0	5.666667	5.689272	1
f11	5.3E-05	0.000724	0.000295	7.73E-05	0.000726	0.000337	0
f12	0.001461	0.004956	0.003531	0.001136	0.003514	0.001558	1

Table 3. Mean, standard deviation of number of function evaluations, success rate of DE and mPSO-DE

	DE			mPSO-DE		
Problems	**Mean**	**Std. Dev.**	**SR (%)**	**Mean**	**Std. Dev.**	**SR (%)**
f1	7375	302.79	100	10985	690.97	100
f2	2240	1508.22	100	8515	10757.65	100
f3	13620	1856.19	100	19255	1488.83	100
f4	109780	116759.41	73.33	161245	115623.22	60
f5	300000	0	0	300000	0	0
f6	251335	110678.58	16.67	194200	141425.99	36.67
f7	300000	0	0	300000	0	0
f8	300000	0	0	292330	29231.62	6.67
f9	75170	8580.91	100	112590	15038.31	100
f10	300000	0	0	291090	48802.08	3.33
f11	13245	1533.33	100	18430	2687.25	100
f12	300000	0	0	300000	0	0

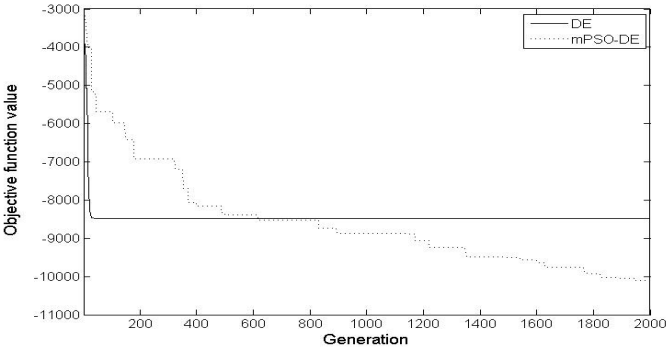


Fig. 2. Convergence graph of DE and mPSO-DE for function f_5

6 Conclusions

In this work, the scale factor and cross-over rate are co-evolved with DE individual vectors using particle swarm optimization algorithm. The proposed algorithm is applied on 12 unconstrained optimization problems and experimental results are compared with classical DE algorithm. From the experimental studies, it is found that the proposed algorithm performed better than classical DE algorithm in optimization of specifically multi-modal, step and noisy functions. A comparative study with different self-adaptive DE algorithms is considered as a future enhancement of this work.

References

1. Das, S., Abraham, A., Konar, A.: Particle Swarm Optimization and Differential Evolution Algorithms–Technical Analysis, Applications and Hybridization Perspectives. In: Liu, Y., Sun, A., Loh, H.T., Lu, W.F., Lim, E.-P. (eds.) Advances of Computational Intelligence in Industrial Systems. SCI, vol. 116, pp. 1–38. Springer, Heidelberg (2008)
2. Das, S., Suganthan, P.N.: Differential Evolution: A Survey of the State-of-the-Art. IEEE Transaction on Evolutionary Computation 15(1) (2011)
3. Storn, R., Price, K.: Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization (11), 341–359 (1997)
4. Shi, Y., Eberhart, R.C.: A modified particle swarm optimizer. In: Proc. IEEE World Congr. Comput. Intell., pp. 69–73 (1998)
5. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: 6th Symposium on Micro Machine and Human Science, Nagoya, Japan, pp. 39–43 (1995)
6. Liu, J., Lampinen, J.: A Fuzzy Adaptive Differential Evolution Algorithm. Softcomputing (9), 448–462 (2005)
7. Islam, S.M., Das, S., Ghose, S., Roy, S., Suganthan, P.N.: An Adaptive Differential Evolution Algorithm With Novel Mutation and Crossover Strategies for Global Numerical Optimization. IEEE Transaction on Systems, Man and Cybernetics-Part B: Cybernetics (2011)
8. Thangaraj, R., Pant, M., Abraham, A.: A Simple Adaptive Differential Evolution Algorithm. In: World Congress on Nature and Biologically Inspired Computing (NaBIC 2009), pp. 457–462 (2009)
9. Brest, J., Greiner, S., Boškovic, B., Mernik, M., Žumer, V.: Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems. IEEE Transaction on Evolutionary Computation 10(6), 646–657 (2006)
10. Qin, A.K., Huang, V.L., Suganthan, P.N.: Differential Evolution Algorithm With Strategy Adaptation for Global Numerical Optimization. IEEE Transaction on Evolutionary Computation 13(2), 398–417 (2009)
11. Jin, W., Gao, L., Ge, Y., Zhang, Y.: An Improved Self-Adaptive Differential Evolution Algorithm. In: International Conference on Computer Design and Applications, pp. 341–344 (2010)
12. Yang, Z., Tang, K., Yao, X.: Self-Adaptive Differential Evolution with Neighborhood Search. In: IEEE Congress on Evolutionary Computation, Hong-Kong, pp. 1110–1116 (2008)
13. Das, N.G.: Statistical Methods (Combined Vol). Tata Mcgraw Hill Education Private Limited (2008)
14. Yao, X., Liu, Y., Lin, G.: Evolutionary programming made faster. IEEE Transactions on Evolutionary Computation 3, 82–102 (1999)

On the Non Linear Dynamics of the Global Best Particle in Particle Swarm Optimization

Dipankar Maity[1], Udit Halder[1], Swagatam Das[2], and Bijaya Ketan Panigrahi[3]

[1] Dept. of Electronics and Telecomm. Engineering, Jadavpur University, Kolkata, India
[2] Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata, India
[3] Dept. of Electrical Engineering, Indian Institute of Technology, Delhi, India
{dipankarmaity1991,udithalder99,bijayaketan.panigrahi}@gmail.com,
swagatam.das@isical.ac.in

Abstract. The dynamics of Particle swarm optimization has been developed from the collective behavior of the social creatures like fish schooling and gradually has become a powerful global optimization technique. In this paper we do the analysis on a continuous variant of PSO. The non linear dynamics of the global best particle is studied here and the exponential convergence is ensured. The effects of the different control parameters on the convergence of the global best particle are also studied.

Keywords: Particle swarm optimization, global optimization, stability, convergence, dynamics.

1 Introduction

Kennedy and Eberhart [1] proposed the concept of function optimization by means of Particle swarm optimization. The velocity and position update equation governing the dynamics can be written as:

$$\boldsymbol{v}_i(t+1) = \omega.\boldsymbol{v}_i(t) + a \cdot rand_1(\boldsymbol{l}_i(t) - \boldsymbol{x}_i(t)) + b \cdot rand_2(\boldsymbol{g}(t) - \boldsymbol{x}_i(t)), \tag{1}$$

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1). \tag{2}$$

The ω is called as 'inertia factor' as it is related to the inertial velocity. Here, $\boldsymbol{l}_i(t)$ is the best previous position found by the i^{th} particle and $\boldsymbol{g}(t)$ is the best position discovered by the whole population at t^{th} iteration. '*a*' and '*b*' are the acceleration coefficients reflecting the weighting of stochastic acceleration terms that pull each particle toward the time variant *pbest* and *gbest* positions. An analysis [2] can be found considering fixed attractors instead of time varying. Instead of gbest technique lbest PSO also developed to maintain the diversity of the swarm. In spite of all these things the basic PSO algorithm still attracts researchers to explore the beauty of this algorithm. Ozcan and Mohan [3] considered single particle moving towards fixed attracters which was extended to a multi particle and multidimensional space in [4]. Trelea [2] also considered fixed attractors and analyzed the system convergence property. Clerc and Kennedy did a stability analysis in [5]. Brandstätter and

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 425–432, 2012.

Baumgartner [6] related the PSO model of Clerc and Kennedy [5] with a damped mass-spring oscillator, making it possible to rewrite the model using the notions of damping factor and natural vibrational frequency. A number of similar studies have been done, some of them can be found in [7-9]. Recently Fernández-Martínez and García Gonzalo [9] found a continuous form of *gbest* PSO.

In this paper we did the analysis of PSO in a continuous time domain where we generalize the dynamics as [9]. The notion of critical points [10] and its' natures of stabilities are introduced here. Various types of stable behaviors along with limit cyclic nature of the system is done under various conditions and justified empirically.

2 PSO Dynamics

The PSO dynamics discussed so far is for discrete time domain but for the analysis of the dynamics it can be generalized [9]. The dynamics is given by:

$$\boldsymbol{v}_i(t+\Delta t) = \left(1-(1-\omega)\Delta t\right)\cdot \boldsymbol{v}_i(t) + a\cdot rand_1 \cdot \Delta t(\boldsymbol{l}_i(t) - \boldsymbol{x}_i(t)) + b\cdot rand_2 \cdot \Delta t(\boldsymbol{g}(t) - \boldsymbol{x}_i(t)) \cdot \quad (3)$$

$$\boldsymbol{x}_i(t+\Delta t) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+\Delta t)\cdot \Delta t \quad . \quad (4)$$

From equation (4) we have, $\boldsymbol{v}_i(t+\Delta t) = \left(\boldsymbol{x}_i(t+\Delta t) - \boldsymbol{x}_i(t)\right)/\Delta t$ and similarly $\boldsymbol{v}_i(t) = \left(\boldsymbol{x}_i(t) - \boldsymbol{x}_i(t-\Delta t)\right)/\Delta t$. Substituting these expressions in equation (3) we get:

$$\ddot{\boldsymbol{x}}_i(t) + (1-\omega)\dot{\boldsymbol{x}}_i(t) + c\boldsymbol{x}_i(t) = a\cdot rand_1 \cdot \boldsymbol{l}_i(t) + b\cdot rand_2 \cdot \boldsymbol{g}(t), \quad (5)$$

Where $c = a\cdot rand_1 + b\cdot rand_2$, $\boldsymbol{v}_i(t) = \dot{\boldsymbol{x}}_i(t) = \lim_{\Delta t\to 0}\left(\boldsymbol{x}_i(t) - \boldsymbol{x}_i(t-\Delta t)\right)/\Delta t$,

and $\dot{\boldsymbol{v}}_i(t) = \ddot{\boldsymbol{x}}_i(t) = \lim_{\Delta t\to 0}\left(\boldsymbol{x}_i(t+\Delta t) - 2\boldsymbol{x}_i(t) + \boldsymbol{x}_i(t-\Delta t)\right)/(\Delta t)^2$.

So we can write from equation (5)

$$\dot{\boldsymbol{v}}_i(t) + (1-\omega)\boldsymbol{v}_i(t) + c\boldsymbol{x}_i(t) = a\cdot rand_1 \cdot \boldsymbol{l}_i(t) + b\cdot rand_2 \cdot \boldsymbol{g}(t), \quad (6)$$

$$\dot{\boldsymbol{x}}_i(t) = \boldsymbol{v}_i(t)\cdot \quad (7)$$

By letting $E(\boldsymbol{x}_i(t)) = \boldsymbol{\mu}_x(t)$ and $E(\boldsymbol{v}_i(t)) = \boldsymbol{\mu}_v(t)$ and applying expectation operator on both sides of equations (6) and (7) and interchanging the order of differentiation and expectation, we obtain:

$$\dot{\boldsymbol{\mu}}_v(t) = -(1-\omega)\boldsymbol{\mu}_v(t) - c\boldsymbol{\mu}_x(t) + (a\cdot \boldsymbol{L}(t) + b\cdot \boldsymbol{g}(t))/2. \quad (8)$$

$$\dot{\boldsymbol{\mu}}_x(t) = \boldsymbol{\mu}_v(t). \quad (9)$$

Now we define a new system variable $\boldsymbol{y}(t)$ defined as, $\boldsymbol{y}(t) = \left[\boldsymbol{\mu}_x(t) \quad \boldsymbol{\mu}_v(t)\right]^T$.

So, $$\dot{\boldsymbol{y}}(t) = \mathbf{A}\cdot \boldsymbol{y}(t) + \mathbf{B}\cdot \boldsymbol{r}(t), \quad (10)$$

where $\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -c & -(1-\omega) \end{bmatrix}$, $\mathbf{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ and $\boldsymbol{r}(t)$ follows from equation (8).

$\boldsymbol{r}(t) = \left(a \cdot \boldsymbol{L}(t) + b \cdot \boldsymbol{g}(t)\right)/2$, because $\boldsymbol{g}(t)$ is same for all the particles so it directly follows that $E(b \cdot rand_1 \cdot \boldsymbol{g}(t)) = b \cdot \boldsymbol{g}(t)/2$ and $E(a \cdot rand_1 \cdot \boldsymbol{l}_i(t)) = a \cdot \boldsymbol{L}(t)/2$. For a global best particle $x(t) = l(t) = g(t)$. So we can re-write the equation (8) and (9) as follows:

$\dot{\boldsymbol{\mu}}_v(t) = -(1-\omega)\boldsymbol{\mu}_v(t)$, and $\dot{\boldsymbol{\mu}}_x(t) = \boldsymbol{\mu}_v(t)$, where $\boldsymbol{\mu}_x(t) = g(t)$ and $\boldsymbol{\mu}_v(t) = v_g(t)$

Now the dynamics is of the form ,

$$\dot{\boldsymbol{y}}(t) = \mathbf{A} \cdot \boldsymbol{y}(t). \tag{11}$$

For the system (11) the critical point of the system is obtained by assigning $\boldsymbol{y}(t) = \mathbf{0}$ and this leads to $(\boldsymbol{\mu}_x, \boldsymbol{\mu}_v) \equiv (0,0)$. This is an isolated critical point as there exists no other critical points for the system under consideration. For the system described in equation (11) the Eigen values of the system is given by: $\lambda_1, \lambda_2 = \left(-(1-\omega) \pm \sqrt{(1-\omega)^2 - 4c}\right)/2$. The corresponding Eigen vectors are $\boldsymbol{\alpha}_1 = \begin{bmatrix} 1 & \lambda_1 \end{bmatrix}^T$, $\boldsymbol{\alpha}_2 = \begin{bmatrix} 1 & \lambda_2 \end{bmatrix}^T$. As both $\boldsymbol{y}_1(t) = \boldsymbol{\alpha}_1 . e^{\lambda_1 t}$ and $\boldsymbol{y}_2(t) = \boldsymbol{\alpha}_2 . e^{\lambda_2 t}$ are the solutions of the system (11), we can write the solution of system (11) as a linear combination of $\boldsymbol{y}_1(t)$ and $\boldsymbol{y}_2(t)$, and is given as: $\boldsymbol{y}(t) = c_1 \cdot \boldsymbol{y}_1(t) + c_2 \cdot \boldsymbol{y}_1(t)$.

$$\begin{bmatrix} \boldsymbol{\mu}_x(t) & \boldsymbol{\mu}_v(t) \end{bmatrix}^T = \begin{bmatrix} c_1 \cdot e^{\lambda_1 t} + c_2 \cdot e^{\lambda_2 t} & \lambda_1 \cdot c_1 \cdot e^{\lambda_1 t} + \lambda_2 \cdot c_2 \cdot e^{\lambda_2 t} \end{bmatrix}^T. \tag{12}$$

c_1 and c_2 depends on the initial conditions of the dynamical system, given by:

$c_1 = (\lambda_2 \cdot \boldsymbol{\mu}_x(0) - \boldsymbol{\mu}_v(0))/(\lambda_2 - \lambda_1)$, and $c_2 = (\boldsymbol{\mu}_v(0) - \lambda_1 \cdot \boldsymbol{\mu}_x(0))/(\lambda_2 - \lambda_1)$. The nature of the Eigen values determines the dynamics, for example simultaneously positive values for both eigen values leads to an instability. The Eigen values are completely determined by the system parameters e.g. ω, '*a*' and '*b*'. Now we will discuss different cases and conclude about the stability of the system.

2.1 Case 1: λ_1, λ_2 Are Both Real, Unequal and of Same Sign

The condition can be obtained iff $(1-\omega)^2/4 > c > 0$. We have, $\boldsymbol{\mu}_x(t)/\boldsymbol{\mu}_v(t) = \left(c_1 \cdot e^{\lambda_1 t} + c_2 \cdot e^{\lambda_2 t}\right)/\left(\lambda_1 \cdot c_1 \cdot e^{\lambda_1 t} + \lambda_2 \cdot c_2 \cdot e^{\lambda_2 t}\right)$. If $c_1 = 0$ then $\boldsymbol{\mu}_x(t)/\boldsymbol{\mu}_v(t) = 1/\lambda_2$ and if $c_2 = 0$ then $\boldsymbol{\mu}_x(t)/\boldsymbol{\mu}_v(t) = 1/\lambda_1$. These two special cases results in a linear variation of $\boldsymbol{\mu}_x(t)$ with $\boldsymbol{\mu}_v(t)$ with slope $1/\lambda_1$ or $1/\lambda_2$. Otherwise

$$\boldsymbol{\mu}_x(t)/\boldsymbol{\mu}_v(t) = \left((c_1/c_2) \cdot e^{(\lambda_1 - \lambda_2)t} + 1\right)/\left((\lambda_1 \cdot c_1/c_2) \cdot e^{(\lambda_1 - \lambda_2)t} + \lambda_2\right). \tag{13}$$

Equation (13) describes a rectilinear path. $\lambda_1 < \lambda_2 < 0$ leads to $\lim_{t \to \infty} \boldsymbol{\mu}_x(t)/\boldsymbol{\mu}_v(t) = 1/\lambda_2$, so all the rectilinear paths enters the critical point with limiting slope of $1/\lambda_2$. In case $\lambda_2 < \lambda_1 < 0$ all the paths enters with limiting slope of

$1/\lambda_1$. If $0 < \lambda_2 < \lambda_1$ or $0 < \lambda_1 < \lambda_2$ then all the paths diverges from the critical point with initial slopes $1/\lambda_1$ and $1/\lambda_2$ respectively and the system is unstable.

a) Now if $\omega < 1$, both λ_1 and λ_2 are negative. Hence, the critical point is a node and this condition results into an asymptotically stable behavior of the system.

b) If $\omega > 1$, both λ_1 and λ_2 are positive and hence this condition will lead to an unstable dynamics.

A typical plot is given in figure 1. In figure 1 each half of the straight line represents the cases either $c_1 = 0$ or $c_2 = 0$. Other lines follow a rectilinear path, with limiting slope equal to the slope of the straight line.

2.2 Case 2: λ_1, λ_2 Are Both Real, Unequal and of Opposite Sign

The condition can be written as $c < 0$. Here also if $c_1 = 0$ then the dynamics follows a linear path with slope $1/\lambda_2$ and if $c_2 = 0$ then the slope of the line is $1/\lambda_1$. Now consider none of c_1 and c_2 is zero and say $\lambda_1 < 0 < \lambda_2$. From equation (13) we have $\lim_{t\to\infty} \mu_x(t)/\mu_v(t) = 1/\lambda_2$, but the path does not approach to the critical point (0,0) as $t \to \infty$ or $t \to -\infty$ and also does not pass through the critical point for any $-\infty < t_0 < \infty$. This follows that the rectilinear path is asymptotic to the linear paths. For any value of ω if $c < 0$ then the system is always unstable and its' nature becomes independent of ω. Thus we can say that any negative value of c always leads to an unstable situation of the dynamics. In this case the critical point is called a saddle point.

A typical plot is given in figure 2. As discussed above the asymptotic lines are special case of either $c_1 = 0$ or $c_2 = 0$. Other lines follow this asymptotic path as can be seen readily from figure 2.

2.3 Case 3: λ_1, λ_2 Are Both Real and Equal

The condition is obtained iff $(1-\omega)^2/4 = c$. In this case the characteristics equation has double root and the solution of the system (11) is given as:

$$\begin{bmatrix}\mu_x(t) & \mu_v(t)\end{bmatrix}^T = c_1 \cdot \begin{bmatrix}1 & \lambda\end{bmatrix}^T \cdot e^{\lambda t} + c_2 \cdot \left(\begin{bmatrix}1 & \lambda\end{bmatrix}^T \cdot t + \begin{bmatrix}0 & 1\end{bmatrix}^T\right) \cdot e^{\lambda t}.$$

If $c_2 = 0$ then $\mu_x(t)/\mu_v(t) = 1/\lambda$ and it conveys that the system dynamics follows a linear path. For $c_2 \neq 0$ the dynamics follows a rectilinear path.

$$\mu_x(t)/\mu_v(t) = \left(c_1 + c_2 \cdot t\right)/\left(\lambda \cdot c_1 + \lambda \cdot c_2 \cdot t + c_2\right).$$

So $\lim_{t\to\infty} \mu_x(t)/\mu_v(t) = 1/\lambda$. If $\lambda < 0$ then the rectilinear path enters the critical point with the slope equal to the inverse of the Eigen value and the system is stable otherwise the system becomes unstable.

a) If $\omega < 1$ then $\lambda < 0$ and the system is asymptotically stable.
b) If $\omega > 1$ then $\lambda > 0$ and the system dynamics is unstable.

Figure 3 shows a typical plot of this case. All the lines enter the critical point with slope equal to the slope of the straight lines. If $\omega > 1$ then the nature of the graph will be same except the direction of the arrows will be reversed.

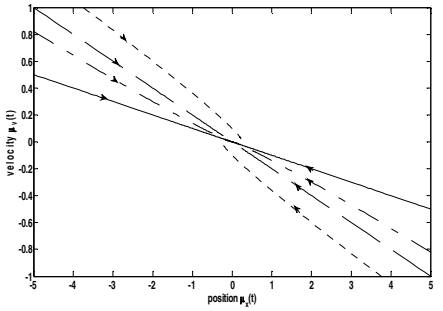


Fig. 1. Phase plot of case 1

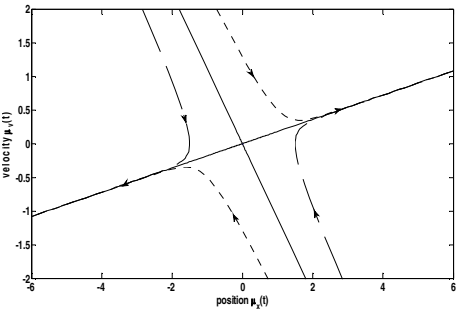


Fig. 2. Phase plot of case 2

2.4 Case 4: λ_1, λ_2 Are Complex Conjugates Having Non-zero Real Part

$(1-\omega)^2/4 < c$ is the required condition and under this condition, and we can write:

$\lambda_1, \lambda_2 = -\left((1-\omega) \pm j\sqrt{4c-(1-\omega)^2}\right)/2$. Considering $(1-\omega)/2 = \sigma$ and $\sqrt{4c-(1-\omega)^2}/2 = \delta$, we can write the solution of (11) as follows:

$\mu_x(t) = e^{-\sigma t} \cdot (c_1 \cdot \cos(\delta t) + c_2 \cdot \sin(\delta t))$ and

$$\mu_v(t) = e^{-\sigma t} \cdot \begin{pmatrix} -c_1 \cdot (\sigma \cdot \cos(\delta t) + \delta \cdot \sin(\delta t)) \\ +c_2 \cdot (\delta \cdot \cos(\delta t) - \sigma \sin(\delta t)) \end{pmatrix}.$$

It is clear from the expressions of $\mu_x(t)$ and $\mu_v(t)$ that they follows a spiral path and approach to the critical point as $t \to \infty$ iff $(1-\omega)/2 = \sigma > 0$ or $\omega < 1$ and the system is stable. If $\sigma < 0$ then clearly the system follows a rectilinear diverging path and the system becomes unstable.

a) When $\omega < 1$ then, the critical point is asymptotically stable.
b) When $\omega > 1$ then, it will act as an unstable system.

2.5 Case 5: λ_1, λ_2 Are Both Pure Imaginary Numbers

When $\omega = 1$ and $c > 0$, the above mentioned condition is satisfied. We can write the system response easily from section 2.4 only by replacing σ with zero. Thus the dynamics of (11) has a solution like:

$$\mu_x(t) = (c_1 \cdot \cos(\delta t) + c_2 \cdot \sin(\delta t)).$$
$$\mu_v(t) = \left(-c_1 \cdot \delta \cdot \sin(\delta t) + c_2 \cdot \delta \cdot \cos(\delta t)\right).$$

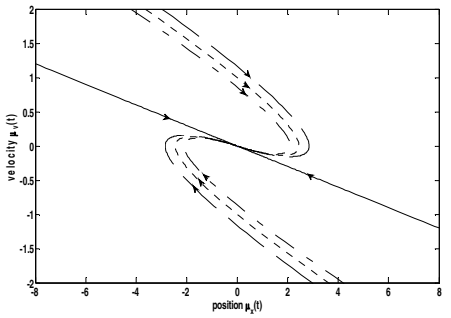


Fig. 3. Phase plot of case 3

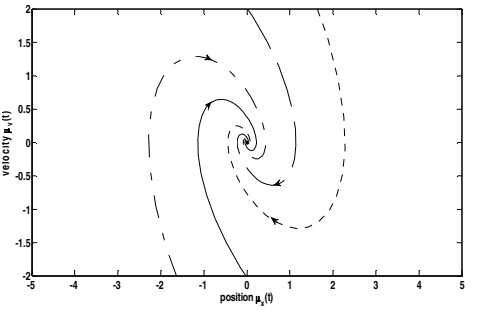


Fig. 4. phase plot of case 4

It is clear from above that the path does not enter the critical point (0,0) as $t \to \infty$ rather it encircles the critical point infinitely many time in an elliptic path arbitrarily near to the point(0,0). In this case the critical point is called as stable center. Thus, if $\omega = 1$ and $c > 0$ then the system is stable but not asymptotically stable. A typical plot is given below in figure 5.

The different conditions for different types of stabilities and instabilities are given in Table 1. We plot the function $(1-\omega)^2 - 4c = 0$ and we show the region of stability, instability and limit cyclic nature. Points on the parabola of figure 6 follows the relation $(1-\omega)^2 = 4c$; thus making the critical point a node and depending on whether $\omega > 1$ or $\omega < 1$ the system is unstable or asymptotically stable. Any point on $\omega = 1$ line with $c > 0$ represents elliptic paths. The region $c < 0$ or $\omega > 1$ is always unstable.

Table 1. Condition for stability

Value of ω	Value of c	Nature of the critical point	Stability of the critical point
$\omega < 1$	$c < 0$	Saddle point	Unstable
$\omega < 1$	$(1-\omega)^2/4 > c > 0$	Node	***Asymptotically stable***
$\omega < 1$	$c = (1-\omega)^2/4$	Node	***Asymptotically stable***
$\omega < 1$	$c > (1-\omega)^2/4$	Spiral point	***Asymptotically stable***
$\omega = 1$	$c < 0$	Saddle point	Unstable
$\omega = 1$	$c > 0$	Center	**Stable**
$\omega > 1$	$c < 0$	Saddle point	Unstable
$\omega > 1$	$(1-\omega)^2/4 > c > 0$	Node	Unstable
$\omega > 1$	$c = (1-\omega)^2/4$	Node	Unstable
$\omega > 1$	$c > (1-\omega)^2/4$	Spiral point	Unstable

In the above section we have shown how the dynamics of global best particle associated with an exponential converging behaviour. System (10) represents the dynamics of a general particle in the swarm. From that we can easily find out that the

critical point of that system is, $\boldsymbol{\mu}_x(t) = \left(a \cdot \boldsymbol{L}(t) + b \cdot \boldsymbol{g}(t)\right) \big/ \left(a+b\right)$ and $\boldsymbol{\mu}_y(t) = 0$. So the critical point is time varying in nature, every time when the particle finds a better position, the critical point changes, analysis of which is beyond scope of this paper.

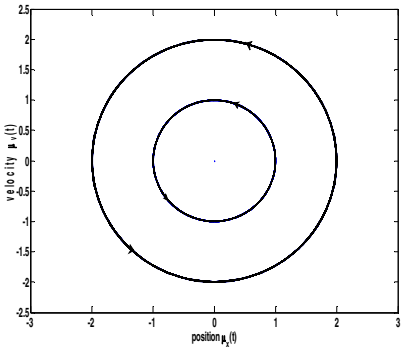


Fig. 5. Phase plot of case 5

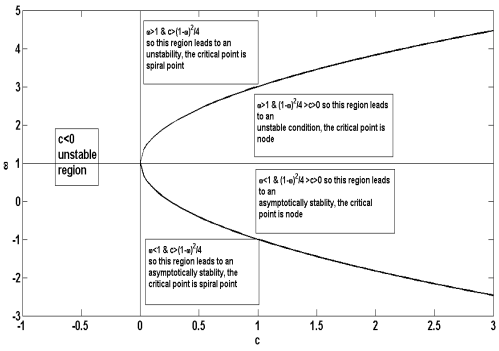


Fig. 6. Region on stability on the $\omega - c$ plane

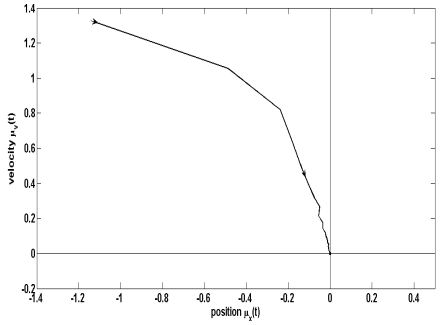


Fig. 7a. Node

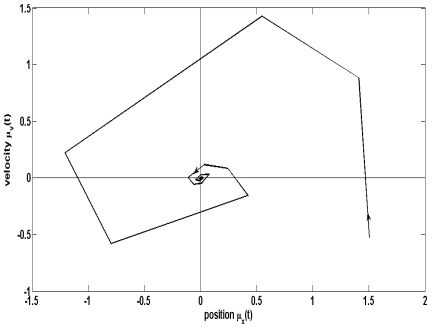


Fig. 7b. Spiral point

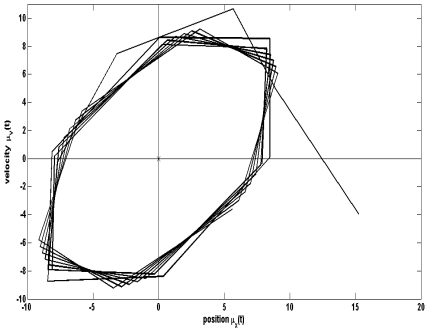


Fig. 7c. Circle

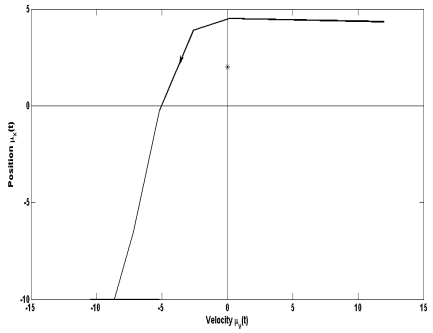


Fig. 7d. Saddle point

3 Simulation Results

Here we present the phase plots of one dimensional component of mean position ($\mu_x(t)$) versus the corresponding dimensional component of mean velocity ($\mu_v(t)$).The simulation results successfully justify the analytical results. Figure 7a is obtained by setting $\omega = 0.4$, $a = 0.07$ and $b = 0.05$ which satisfies the condition for node. Figure 7b is obtained by setting $\omega = 0.7$, $a = 1$ and $b = 0.5$ which satisfies the conditions for a spiral. By setting $\omega = 1$, $a = 0.7$ and $b = 0.2$ we get figure 7c which is a circle. Figure 7d, representing a saddle point, is obtained by choosing $\omega = 0.7$ $a = -2$ and $b = -1.5$.

4 Conclusion

This paper successfully shows the various natures of the dynamics in response to its different parameter settings. Apart from spiral or limit cyclic nature, we introduced the nature of node and saddle point that can also be found in the dynamics. All the natures shown in the simulation results justify the exponential behavior of the dynamics of a global best particle. As a future work, the analysis can be extended to explore the dynamics of a general particle.

References

1. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proc. IEEE Int. Conf. Neural Netw (ICNN), vol. 4, pp. 942–(November 1948)
2. Terelea, I.C.: The particle swarm optimization algorithm: convergence analysis and parameter selection. Information Processing Letters 85, 317–325 (2003)
3. Ozcan, E., Mohan, C.K.: Analysis of a simple particle swarm optimization system. In: Proc. Intell. Eng. Syst. Through Artificial. Neural Netw. (ANNIE 1998), vol. 8, pp. 253–258 (October 1998)
4. Ozcan, E., Mohan, C.: Particle swarm optimization: surfing the waves. In: Proc. 1999 Congr. Evol. Comput. IEEE Service Center, Piscataway (1944)
5. Clerc, M., Kennedy, J.: The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. on Evolutionary Computation 6(1), 58–73 (2002)
6. Brandstatter, B., Baumgartner, U.: Particle swarm optimization-massspring system analogon. IEEE Trans. Magnetics 38(2), 997–1000 (2002)
7. Kennedy, J.: Small Worlds and Mega-Minds: Effects of Neighborhood Topology on Particle Swarm Performance. In: Proceedings of the 1999 Congress of Evolutionary Computation, vol. 3, pp. 1931–1938. IEEE Press (1999)
8. Mendes, R., Kennedy, J., Neves, J.: The fully informed particle swarm: simpler, maybe better. IEEE Transactions on Evolutionary Computation 8(3), 204–210 (2004)
9. Fernández-Martínez, J.L., García-Gonzalo, E.: The generalized PSO: A new door for PSO evolution. J. Artif. Evol. Appl. 2008(861275), 15 (2008)
10. Ross, S.L.: Differential Equation, 3rd edn. ch.13

Adaptive Differential Evolution with Directional Information Based Search Moves

Satyajit Neogi[1], Deblina Das[1], and Swagatam Das[2]

[1] Dept. of Electronics and Telecommunication Engg., Jadavpur University, Kolkata -32, India
[2] Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata –108, India
{satmon228,dasdeblina18}@gmail.com, swagatam.das@isical.ac.in

Abstract. Differential Evolution (DE) is one of the most simple and efficient Evolutionary Algorithms exist till now for global optimization problems. It has reported exceptionally good results when tested over all the benchmark problems and some of the real world problems, although it suffers from the troubles of slow and premature convergence. Generally the performance of DE is sensitive to the choice of mutation and crossover strategies and their associated control parameters. In this paper we propose a DE called Adaptive Differential Evolution with Directional Information based Search Moves (ADE-DISM) in which we basically have improved the mutation and crossover strategies adopted in 'DE/rand/1/bin'. In ADE-DISM we varied the control parameters F and CR in an adaptive manner and have introduced a new parameter w. We have used some directional information based moves over the population and introduced a Mean_Best_Vector for mutation purpose. However, the proposed scheme is shown to be statistically significantly better than or at least comparable to several existing DE variants when tested over the CEC 2005 benchmark problems for 30 and 50 dimensions of the problems.

Keywords: Differential Evolution, Global Optimization, Parameter Adaptation, Mutation and Crossover Strategies.

1 Introduction

DE, proposed by Storn and Price [1] is a fast and simple technique for optimizing real valued functions. It is very popular in wide range in the fields of engineering [2-5]. The performance of DE algorithm is based on mutation and crossover strategies and associated control parameters like *NP*, *CR* and *F*. Like other Evolutionary algorithms, two fundamental processes which help in DE are: (a) Variation process which invents different various regions of search space and (b) Selection process which ensures exploitation of previous knowledge about fitness landscape [6].

Practically DE stops progressing towards global optimum even though the population has not converged to a local optimum. This is called stagnation [6].So there has been works on tuning the parameters of DE e.g. *NP, F* and *CR* to increase the convergence speed and accuracy of DE.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 433–441, 2012.

Thus in the rest of the paper we will present a brief outline of DE algorithms in section 2, some of the improved DE variants in section 3, our proposed ADE-DISM in section 4, comparison of our results with some of the best existing DE variants in section 5 and a brief conclusion in section 6.

2 DE Algorithm

DE starts with a population *NP* *D*-dimensional real valued parameter vectors.

Step 1: Initialization
Initialize the generation number G=0 and a population of *NP* numbers $P_{G=}\{X_{1,G,}X_{2,G,}\ldots\ldots\ldots X_{NP,G}\}$. Since the parameter vectors are likely to be changed over different generations, we may represent the i^{th} vector of the population at the current generation as $X_{i,G} =\{x_{1,i,G,}x_{2,i,G}\ldots\ldots x_{D,i,G}\}$, i=1,…., *NP* uniformly distributed over the range $[X_{min,}X_{max}]$ where $X_{min}=\{x^1_{min,\ldots\ldots,} x^D_{min}\}$ and $X_{max}=\{x^1_{max,\ldots\ldots,} x^D_{max}\}$. We initialize j^{th} component of i^{th} vector as

$$x_{j,i,0} = x_{j,min} + rand_{i,j}[0, 1]. (x_{j,max} - x_{j,min})$$

where $rand_{i,j}[0, 1]$ is a uniformly distributed random number lying between 0 and 1 and is instantiated independently for each component of i^{th} vector .

Step 2: Mutation
After initialization, DE creates a donor vector $V_{i,G}$ corresponding to each population member $X_{i,G}$ in current generation through mutation . Five most frequently used mutation strategies are:

DE/rand/1: $V_{i,G}= X^i_{r1,G} +F .(X^i_{r2,G} - X^i_{r3,G})$
DE/best/1: $V_{i,G}= X_{best,G} +F .(X^i_{r1,G} - X^i_{r2,G})$
DE/current-to-best/1: $V_{i,G}= X_{i,G} +F .(X_{best,G} - X_{i,G}) + F .(X^i_{r1,G} - X^i_{r2,G})$
DE/best/2*:* $V_{i,G}= X_{best,G} +F .(X^i_{r1,G} - X^i_{r2,G})+ F .(X^i_{r3,G} - X^i_{r4,G})$
DE/rand/2: $V_{i,G}= X^i_{r1,G} +F .(X^i_{r2,G} - X^i_{r3,G})+ F .(X^i_{r4,G} - X^i_{r5,G})$

The indices r^i_1, r^i_2, r^i_3, r^i_4, r^i_5 are mutually exclusive integers randomly chosen from [1, *NP*] and are different for index *i.* Here, $X_{best,G}$ is the best individual with best fitness.

Step 3: Crossover Step
Through crossover , the donor vector mixes its components with the target vector $X_{i,G}$ to form the trial vector $U_{i,G}=\{U_{1,i,G,}U_{2,i,G}\ldots\ldots\ldots U_{D,i,G}\}$. Each component of $U_{i,G,}$ is taken from i^{th} mutant vector $V_{i,G}$ with probability *CR* and from the i^{th} individual $X_{i,G}$ with probability (1-*CR*). *CR* is called the crossover rate, predefined by the user. In order to make sure at least one component of $U_{i,G}$ is inherited from $V_{i,G}$ randomly choose one component index between 1 and *D* and the chosen component $U_{i,G}$ will take the same position as $V_{i,G,}$ with probability 1.The crossover step is outlined as:

$u_{j,i,G} = v_{j,i,G}$ if $rand\ (0, 1) \leq CR$ *or* $j==k$
$= x_{j,i,G}$ otherwise

where k ($1 \leq k \leq D$) is a randomly generated integer.

Step 4: **Selection Step**

Selection determines whether the target or the trial vector survives to the next generation i.e. $G=G+1$.The selection operation is described as:

$X_{i,G+1} = U_{i,G}$ if $f(U_{i,G}) \leq f(X_{i,G})$
$= X_{i,G}$ if $f(U_{i,G}) \geqq f(X_{i,G})$

where $f(X)$ is the objective function to be minimized. Therefore if the new trial vector yields an equal or lower value of the objective function, it replaces the corresponding target vector in the next generation; otherwise, the target is retained in the population.

3 Review of Previous Works on DE Algorithm

DE is sensitive to control parameters for complex problems [7]. Inappropriate choice of Scale Factor F and Crossover Rate CR may lead to premature convergence or stagnation [8]. In DE higher the population size NP, higher is the probability of finding the global minimum. But this larger population means slow rate of convergence. Initially 10D was a good choice for DE [4].However 5D to 10D [8], 3D to 8D [7], 2D to 4D [9] were suggested.

Crossover Rate CR is a probability 0<=CR<=1 for mixing trial and target vector. A large CR increases the convergence speed [4, 7, and 8].Initially CR=0.1 is a good choice [8], while CR= 0.9 or 1.0 can increase convergence speed.

Scale Factor F is a parameter to escape local optimum [7, 9]. It is chosen to be kept between (0.5, 1) [4]. F must be strictly larger than zero. A good initial choice for F=0.6 or 0.5 [7, 8] while in [9] F=0.9 is a good value. F is kept generally between 0.4 and 0.95 [9].

Choosing the control parameters and mutation strategies DE performance is improved in various adaptation schemes proposed in [6, 10, 11, 12].

Control Parametric Setup for 2 Classical DE and 4 State-of-Art ADE Variants

(i) DE/current-to-best/1/bin with F=0.8 and CR=0.9.
(ii) DE/rand/1/bin with F=0.8 and CR=0.9.
(iii) JADE with c=0.1, p=0.05 and optional external archive [11].
(iv) jDE with F_l=0.1, F_u=0.9, T_1=T_2=0.1 [10].
(v) SADE with CR =N (0.5, 0.15), $F_i = F_{i1} + N\ (0, 0.5)^* (F_{i2}-F_{i3})$ [12].
(vi) DEGL with α=β=F=0.8, CR=0.9, neighborhood size=0.1*NP [13].

The population size NP for most of the DE variants is kept 100, if not otherwise stated.

4 Adaptive Differential Evolution with Directional Information Based Search Moves (ADE-DISM)

The performance of DE in solving numerical optimization problems is based on the mutation and crossover strategies adopted and its associated parameter values [16].In ADE-DISM we propose different mutation and crossover strategies than usual DE algorithms, e.g. DE/rand/1 or DE/best/1, with different parameter values of F and CR. Also for the crossover we have introduced a new parameter w.

As usual DE variants, in the initialization step, we have initialized the population (NP), the search space, individual dimension D, the maximum number of generations (max_gen), and set the generation counter g=0. The individual with the best fitness among the population is designated as $f_{best.}$

The mutation strategy of ADE-DISM is basically based on some directional information based search moves over the population and the mean best vector C_g. The mutant vector $V_{i,g}$ corresponding to a target vector $X_{i,g}$ from the population is formed on the basis whether the fitness is improved (FI) or not in a generation. Here, we use a difference vector set $X_{diff,g}$ [17], generated in each generation where size(difference vector set) <= NP, for the formation of $V_{i,g}$, if fitness is improved (FI=1) in that generation.

$$V_{i,g} = X_{r1,g} + F \,.\, (X_{diff,r2,g}) \qquad \text{,where } r1 \text{ and } r2 \text{ are randomly chosen among the population.}$$

If fitness is not improved (FI=0), we take the best fit q% of the population in some generation and take their mean to form the mean_best_vector C_g, and use a mutation procedure using C_g.

$$V_{i,g} = X_{r3,g} + F \,.\, (\, C_g - X_{r4,g}) \qquad \text{, where r3 and r4 are randomly chosen among the population.}$$

q is reduced in each generation accordingly.

$$q_{g+1} = q_g - q_g \,.\, (g/max\_gen) \qquad \text{, where } max\_gen\text{=maximum no. of generations.}$$

In the crossover step, we propose a new parameter w, which basically decides the percentage of parent vector (or mutant vector) to be retained in the offspring or trial vector $U_{i,g}$.

$$u_{j\,i,g} = w \,.\, v_{j,i,g} + (1-w) \,.\, x_{j,i,g} \qquad \text{,if } rand_{i,j}\,(0,1) < CR \text{ or } k_i = j$$
$$= x_{j,i,g} \qquad \text{,otherwise}$$

where k_i is a randomly chosen integer between 1, 2…, D and

$$U_{i,g} = \{\, u_{j,i,g} \,\},\ j=1\ (1)\ D.$$

At the end of the crossover in a generation the fitness of each trial vector is calculated.

If the generated trial vector is better in fitness than its corresponding target vector in that generation, it becomes the parent in the next generation. If not, the target vector retains its parental character in the next generation. At the end of each generation the fitness of each individual is calculated and the best one replaces the previous f_{best} if and only if

$$f_{best,g+1} \leq f_{best,g}$$

This paper uses the mutation criterion of [17] as one of the two mutation schemes used. We show here the results of ADE-DISM (mean and standard deviation of the error) for 30*D* and 50*D* of 25 benchmark problems of CEC-2005 [18, 19].We have used NP=100 for 30*D* and 50*D* problems.

$F = F_{min} + (F_{max} - F_{min}) \cdot (g/max\_gen)$, for problems 1, 3, 5, 6-25
$= Gaussian\ (0.5, 0.1)$, for problems 2 and 4.
$CR = Gaussian\ (0.5, 0.1)$, for problems 1, 3, 5, 6-25
$= 0.7$, for problems 2 and 4.
$w = rand_g(0.85, 1)$.
$q\_initial = 15$ and $q\_final = 10$.

5 Results on Numerical Benchmarks of CEC-2005

We present in this section the results of 30*D* in Table.1, 50*D* in Table.2 while in the following two the tables we show the comparisons of our results with those of DE/rand/1/bin, DE/current-to-best/1/bin, JADE, jDE, SADE and DEGL. The maximum numbers of Function Evaluations (FES) used are 3, 00,000 for 30*D* and 5,00,000 for 50*D*. The statistically significant results are marked in boldface.

Table 1. Mean and standard deviation values of the errors for 30*D*.The best entries are marked in boldface.

	Algorithms						
Function	DE/rand/1 Mean(Std)	DE/c-b/1 Mean(Std)	JADE Mean(Std)	jDE Mean(Std)	SADE Mean(Std)	DEGL Mean(Std)	ADEDISM Mean(Std)
f_1	2.453e-05 (3.475e-05)	2.453e-25 (3.4756e-05)	1.3258e-54 (9.2436e-54)	3.8652e-29 (4.5732e-29)	6.7843e-30 (2.3879e-30)	2.3462e-20 (5.6234e-20)	**0.000e+00 (0.000e+00)**
f_2	5.497e-02 (1.275e-02)	6.295e-08 (1.2927e-07)	**2.514e-26 (3.4269e-26)**	7.506e-06 (7.3804e-06)	9.719e-08 (4.8596e-07)	1.1757e-07 (6.5592e-08)	5.381e-01 (5.334e-01)
f_3	2.892e+05 (1.94e+05)	1.8191e+05 (1.039e+05)	**4.7421e+04 (1.621e+04)**	2.2663e+05 (1.608e+04)	5.052e+04 (1.575e+05)	2.3114e+05 (1.032e+05)	3.5421e+05 (1.375e+05)
f_4	5.043e-01 (8.578e-01)	1.0714e-02 (2.7821e-02)	**5.115e-07 (4.0194e-07)**	2.7305e-01 (2.7305e-01)	5.816e-06 (1.4479e-05)	1.5746e+03 (9.501e+00)	1.7033e+02 (2.970e+01)
f_5	1.271e+03 (2.83e+02)	5.7628e+02 (1.247e+02)	**3.279e+02 (1.849e+02)**	1.1108e+03 (3.724e+02)	7.880e+02 (1.244e+03)	5.0692e+02 (5.803e+02)	5.741e+02 (5.789e+01)
f_6	2.776e+01 (1.02e+01)	9.2736e+00 (8.283e+00)	**5.6094e+00 (1.944e+01)**	1.1196e+01 (1.398e+01)	2.124e+01 (1.341e+01)	4.784e+01 (1.322e+00)	2.331e+01 (1.045e+00)
f_7	9.657e-01 (9.14e-02)	8.4253e-01 (9.1442e-02)	**6.959e-03 (4.484e-03)**	9.8597e-03 (1.1445e-02)	8.272e-03 (4.5371e-03)	6.999e-01 (4.5371e-03)	1.356e-02 (7.543e-03)
f_8	2.094e+0 (6.052e-02)	2.0942e+01 (5.0063e-02)	2.0929e+01 (2.6628e-02)	2.0931e+01 (2.5067e-02)	2.014e+01 (5.728e-02)	2.0097e+01 (2.3029e-02)	**2.0090e+01 (4.008e-02)**
f_9	4.374e+01 (9.73e+00)	2.3465e+01 (8.639e+00)	**5.927e-22 (8.9543e-22)**	8.3264e-16 (2.3645e-15)	2.273e-15 (1.1369e-14)	1.7591e+01 (3.022e+00)	7.412e+01 (1.712e-01)

Table 1. (*continued*)

f_{10}	1.564e+02 (4.56e+01)	1.9617e+02 (2.168e+01)	**3.031e+01 (8.355e+00)**	5.2547e+01 (4.466e+00)	3.575e+01 (6.081e+00)	3.741e+01 (5.288e+00)	1.831e+02 (4.919e-01)
f_{11}	3.264e+01 (1.09e+01)	3.1683e+01 (7.598e+00)	**2.6456e+01 (1.916e+01)**	3.1370e+01 (2.395e+00)	2.6562e+01 (1.127e+00)	2.7278e+01 (1.573e+00)	3.939e+01 (4.483e-01)
f_{12}	8.433e+04 (6.25e+04)	9.8362e+04 (6.226e+03)	2.697e+04 (6.800e+03)	3.8376e+04 (6.537e+03)	**8.734e+02 (9.338e+02)**	2.5359e+04 (2.888e+03)	8.5545e+05 (8.673e+03)
f_{13}	4.513e+00 (2.26e+00)	3.5182e+00 (2.269e+00)	1.628e+00 (4.873e-02)	1.8568e+00 (1.031e-01)	**1.207e+00 (1.342e-01)**	2.3595e+00 (5.282e-01)	1.2517e+01 (5.02e-03)
f_{14}	1.334e+01 (3.476e-01)	1.3453e+01 (3.479e-01)	1.277e+01 (2.205e-01)	1.3545e+01 (9.940e-02)	**1.276e+01 (2.593e-01)**	1.2961e+01 (4.114e-01)	1.336e+01 (3.433e-02)
f_{15}	4.843e+02 (2.14e+01)	3.8424e+02 (5.143e+01)	2.888e+02 (9.050e+01)	3.5642e+02 (1.871e+01)	3.277e+02 (9.645e+01)	3.4400e+02 (5.066e+01)	**2.054e+02 (4.067e+00)**
f_{16}	2.822e+02 (1.13e+01)	2.2282e+02 (1.639e+02)	**7.438e+01 (3.934e+01)**	1.2854e+02 (4.073e+01)	1.379e+02 (1.702e+01)	2.0350e+02 (1.206e+02)	1.981e+02 (1.002e+00)
f_{17}	3.094e+02 (1.56e+01)	2.3465e+02 (1.583e+02)	**8.4619e+01 (3.576e+01)**	1.6189e+02 (4.725e+01)	1.509e+03 (9.363e+02)	1.4519e+02 (7.324e+01)	2.3407e+02 (1.01e+01)
f_{18}	9.134e+02 (8.43e-01)	9.5042e+02 (2.103e+01)	8.1679e+02 (1.6523e-01)	8.6111e+02 (1.870e+00)	9.544e+02 (3.438e+01)	9.2253e+02 (1.576e+00)	**8.024e+02 (2.009e+00)**
f_{19}	9.918e+02 (1.21e+00)	9.518e+02 (2.114e+01)	8.1644e+02 (1.541e-01)	8.4801e+02 (3.179e+00)	8.458e+02 (6.215e+01)	9.1960e+02 (1.338e+00)	**8.151e+02 (9.546e-01)**
f_{20}	9.131e+02 (1.16e+00)	9.4110e+02 (2.931e+01)	8.1697e+02 (1.723e-01)	8.5466e+02 (9.54e-01)	2.040e+03 (8.768e+02)	9.1989e+02 (3.071e+01)	**8.1146e+02 (3.671e-01)**
f_{21}	5.813e+02 (2.62e+02)	8.3154e+02 (2.624e+02)	8.5838e+02 (1.101e+00)	8.6002e+02 (1.136e+00)	1.730e+03 (5.118e+02)	7.5734e+02 (1.013e+01)	**4.9998e+02 (2.02e-02)**
f_{22}	9.642e+02 (1.14e+01)	9.3438e+02 (4.146e+01)	5.0762e+02 (1.655e+00)	5.0340e+02 (2.915e+00)	1.582e+03 (4.252e+02)	9.2154e+02 (9.841e+01)	**4.9901e+02 (7.081e-02)**
f_{23}	6.213e+02 (3.06e+01)	8.5963e+02 (2.878e+02)	8.6558e+02 (6.610e-01)	6.1835e+02 (4.548e+00)	5.506e+02 (2.489e+01)	7.524e+02 (4.001e+01)	**5.2589e+02 (1.002e+00)**
f_{24}	3.143e+02 (3.22e+01)	3.0493e+02 (3.556e+01)	2.1141e+02 (1.566e+01)	2.1081e+02 (2.883e+00)	2.0985e+02 (1.245e-04)	6.561e+02 (5.012e+01)	**1.989e+02 (1.342e+00)**
f_{25}	9.864e+02 (2.17e+01)	9.6754e+02 (1.306e+01)	2.1052e+02 (2.558e+00)	9.761e+02 (2.409e+01)	5.0012e+02 (5.683e-02)	9.889e+02 (3.015e+01)	**1.9807e+02 (2.102e-01)**

Table 2. Mean and standard deviation values of the errors for 50*D*.The best entries are marked in boldface.

	Algorithms						
Function	DE/rand/1 Mean(std)	DE/c-b/1 Mean(std)	JADE Mean(std)	jDE Mean(std)	SADE Mean(std)	DEGL Mean(std)	ADE-DISM Mean(std)
f_1	9.6015e-06 (1.283e-05)	2.1443e-06 (6.125e-06)	7.4651e-04 (2.419e-14)	3.1544e-09 (4.994e-09)	1.4872e-11 (2.8335e-11)	6.4679e-10 (9.864e-11)	**1.152e-13 (1.247e-15)**
f_2	3.96e+03 (9.307e+03)	2.136e+03 (1.128e+03)	5.6310e-04 (7.823e-04)	5.202e+03 (1.486e+03)	2.28e-03 (8.545e-03)	**1.2960e-05 (9.561e-06)**	3.604e+03 (8.564e+02)
f_3	5.469e+07 (1.328e+07)	1.03e+07 (6.375e+06)	**8.7156e+04 (3.6847e+04)**	2.977e+07 (5.744e+06)	7.179e+05 (1.007e+06)	2.311e+05 (1.032e+05)	6.964e+06 (2.541e+06)
f_4	1.180e+04 (3.332e+03)	8.5675e+03 (2.864e+03)	**3.16e+03 (4.134e+03)**	1.0194e+04 (2.182e-01)	9.778e+04 (9.865e+01)	2.8851e+04 (1.893e+01)	9.245e+03 (2.187e+02)

Table 2. (*continued*)

f_5	8.709e+03 (6.938e+02)	7.462e+03 (1.32e+03)	3.055e+03 (5.485e+02)	4.206e+03 (5.088e+02)	5.992e+03 (4.664e+02)	6.093e+03 (6.840e+02)	**3.632e+03 (4.012e+02)**
f_6	4.9162e+01 (1.182e+01)	1.0872e+07 (1.237e+07)	1.5413e+01 (1.0642e+01)	4.1758e+01 (8.910e+00)	**1.1337e+01 (1.044e+01)**	6.29e+01 (1.108e+01)	1.151e+02 (1.142e+01)
f_7	6.1953e+03 (4.594e-12)	6.6691e+03 (1.795e+02)	6.1932e+03 (1.84e+00)	6.3114e+03 (1.596e+01)	6.1951e+03 (4.594e-12)	6.1953e+03 (4.594e-12)	**3.0893e+01 (4.425e-01)**
f_8	2.1142e+01 (3.33e-02)	2.1133e+01 (4.841e-02)	2.1136e+01 (3.251e-02)	2.1132e+01 (3.807e-02)	2.1132e+01 (3.458e-02)	2.1131e+01 (3.917e-02)	**2.112e+01 (3.145e-02)**
f_9	3.468e+02 (1.199e+01)	2.406e+02 (2.939e+01)	1.352e+02 (2.591e+00)	1.716e+02 (1.409e+01)	**1.148e+02 (1.266e+01)**	1.620e+02 (1.743e+01)	2.892e+02 (3.568e+01)
f_{10}	3.763e+02 (1.578e+01)	2.5467e+02 (7.663+01)	1.935e+02 (2.06e+01)	1.9597e+02 (5.623e+01)	**6.342e+01 (1.287e+01)**	1.0217e+02 (3.559e+01)	2.891e+02 (7.612e+01)
f_{11}	7.264e+01 (1.212e+00)	**4.950e+01 (4.451e+00)**	6.208e+01 (1.744e+00)	7.330e+01 (1.008e+00)	6.634e+01 (1.485e+00)	6.29e+01 (1.36e+01)	7.212e+01 (2.841e+00)
f_{12}	2.049e+06 (5.925e+05)	2.505e+05 (1.137e+05)	1.768e+05 (7.105e+04)	1.473e+05 (1.928e+05)	**8.781e+03 (7.092e+03)**	5.781e+04 (4.566e+04)	4.3446e+06 (4.789e+05)
f_{13}	3.596e+01 (1.446e+00)	3.412e+01 (8.904e+00)	2.3112e+01 (4.784e-01)	2.5603e+01 (1.322e+00)	2.771e+01 (4.112e+00)	3.063e+01 (4.361e+00)	**1.9635e+01 (2.65e+00)**
f_{14}	2.339e+01 (1.486e-01)	2.286e+01 (4.091e-01)	2.284e+01 (2.5486e-01)	2.309e+01 (2.843e-01)	2.284e+01 (2.0634e+01)	**2.262e+01 (3.375e-01)**	2.2989e+01 (5.011e-01)
f_{15}	5.09e+02 (7.981e+01)	4.989e+02 (5.001e+01)	3.769e+02 (8.764e+01)	4.000e+02 (000e+00)	3.8827e+02 (1.0775e+02)	3.8982e+02 (4.9284e+01)	**2.011e+02 (1.065e+00)**
f_{16}	2.7343e+02 (1.0498e+01)	2.5387e+02 (1.4757e+01)	1.437e+02 (5.2267e+01)	2.716e+02 (4.719e00)	1.542e+02 (6.168e+01)	**1.3153e+02 (1.998e+01)**	2.7045e+02 (1.547e+01)
f_{17}	3.7286e+02 (3.1287e+01)	2.5234e+02 (5.7296e+01)	**1.896e+02 (3.8745e+01)**	3.059e+02 (1.163e+01)	1.934e+02 (2.9679e+00)	1.7659e+02 (2.3653e+01)	3.0254e+02 (5.674e+00)
f_{18}	9.9043e+02 (4.8709e+01)	9.2089e+02 (5.663e+01)	9.206e+02 (1.893e+00)	9.145e+02 (3.163e+01)	9.041e+02 (5.208e+01)	9.6067e+02 (2.8458e+01)	**9.025e+02 (5.014e+00)**
f_{19}	9.4100e+02 (3.8003e+01)	9.2667e+02 (5.9865e+01)	9.6031e+02 (2.523e+01)	9.209e+02 (1.0406e+01)	9.3493e+02 (1.9639e+01)	9.143e+02 (2.0105e+01)	**8.6012e+02 (2.038e+01)**
f_{20}	9.8536e+02 (4.4965e+01)	9.3586e+02 (5.3925e+01)	9.8672e+02 (1.8675e+02)	9.9121e+02 (1.5365e+01)	9.3167e+02 (2.0137e+01)	9.2196e+02 (4.5874e+01)	**8.989e+02 (2.018e+00)**
f_{21}	9.1108e+02 (5.7474e+02)	8.9465e+02 (1.7465e+02)	8.523e+02 (3.5175e+02)	8.0619e+02 (1.0869e+02)	8.6400e+02 (1.5779e+02)	8.3600e+02 (2.1772e+02)	**7.321e+02 (8.125e+00)**
f_{22}	9.9463e+02 (1.3465e+01)	9.3443e+02 (1.012e+01)	9.137e+02 (2.435e+01)	9.796e+02 (1.4851e+01)	9.7245e+02 (3.3383e+01)	9.4242e+02 (3.5647e+01)	**5.000e+02 (2.395e-01)**
f_{23}	9.1185e+02 (2.5986e+01)	9.2645e+02 (2.5986e+01)	8.103e+02 (2.4572e+02)	8.3044e+02 (1.0787e+02)	8.6405e+02 (1.5266e+02)	8.3934e+02 (1.662e+02)	**7.352e+02 (3.457e+00)**
f_{24}	7.9849e+02 (2.4586e+01)	7.9645e+02 (1.3642e+01)	**2.000e+02 (0.000e+00)**	**2.000e+02 (0.000e+00)**	**2.000e+02 (0.000e+00)**	7.2465e+02 (8.2066e+01)	**2.000e+02 (0.000e+00)**
f_{25}	1.7586e+03 (4.5365e+00)	1.7846e+03 (6.7654e+00)	1.6632e+03 (5.5842e+00)	1.728e+03 (6.2562e+00)	1.7586e+e+03 (3.1453e+00)	1.671e+03 (6.5096e+00)	**9.600e+02 (4.241e+00)**

From above results, we can see that ADE-DISM is specifically better for the hybrid composition functions. And also, as dimensionality increases the efficiency of ADE-DISM increases for optimizing all the benchmark problems of CEC-2005.

6 Conclusion

The performance of DE is enhanced with proper mutation and crossover strategies and the proper tuning of the associated parameter values. In our ADE-DISM we use a mutant vector using a different mutation criterion using the Mean-best-vector and the difference vector set. When run over the problems of CEC-2005 [14, 15], it is statistically comparable to or better than some of the other DE variants as we have presented here. In future, we will apply our DE scheme to various existing DE algorithms and also to some of the real world optimization problems.

References

[1] Storn, R., Price, K.: Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces, Technical Report TR-95-012, ICSI (1995), http://http.icsi.berkeley.edu/~storn/litera.html

[2] Joshi, R., Sanderson, A.C.: Minimal representation multisensor fusion using differential evolution. IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans 29(1), 63–76 (1999)

[3] Rogalsky, R., Derksen, R.W., Kocabiyik, S.: Differential evolution in aerodynamic optimization. In: Proc. of 46th Annual Conference of Canadian Aeronautics and Space Institute, pp. 29–36 (1999)

[4] Storn, R.: On the usage of differential evolution for function optimization. In: Biennial Conference of the North American Fuzzy Information Processing Society (NAFIPS), pp. 519–523. IEEE, Berkeley (1996)

[5] Venu, M.K., Mallipeddi, R., Suganthan, P.N.: Fiber bragg grating sensor array interrogation using differential evolution. Optoelectronics and Advanced Materials - Rapid Communications 2(11), 682–685 (2008)

[6] Chakraborthy, U.K., Das, S., Konar, A.: Differential evolution with local neighborhood. In: Proceedings of Congress on Evolutionary Computation, pp. 2042–2049. IEEE Press (2006)

[7] Gämperle, R., Müller, S.D., Koumoutsakos, P.: A parameter study for differential evolution. In: Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation, pp. 293–298. WSEAS Press, Interlaken (2002)

[8] Storn, R., Price, K.: Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11(4), 341–359 (1997)

[9] Rönkkönen, J., Kukkonen, S., Price, K.: Real-parameter optimization with differential evolution. In: IEEE Congress on Evolutionary Computation, pp. 506–513 (2005)

[10] Brest, J., Greiner, S., Boscovic, B., Mernik, S., Zumer, V.: Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. IEEE Transactions on Evolutionary Computation 10(8), 646–657 (2006)

[11] Zhang, J.: Jade: Adaptive differential evolution with optional external archive. IEEE Transactions on Evolutionary Computation 13(5), 945–958 (2009)

[12] Qin, A.K., Huang, V.L., Suganthan, P.N.: Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Transactions on Evolutionary Computation 13(2), 398–417 (2009)

[13] Das, S., Konar, A., Chakraborty, U.K.: Two improved differential evolution schemes for faster global search. In: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, pp. 991–998 (2005)
[14] Mallipeddi, R., Suganthan, P.N.: Differential Evolution Algorithm with Ensemble of Parameters and Mutation and Crossover Strategies. Nanyang Technological University, Singapore
[15] Zhang, X., Yuen, Yin, S.: A Directional Mutation Operator for Differential Evolution Algorithms, City University of Hong Kong, Electronic Engineering.
[16] Suganthan, P. N, Hansen, N, Liang, J. J, Deb, K, Chen, Y. P., Auger, A., Tiwari, S.: Problem definitions and evaluation criteria for the CEC2005 special session on real-parameter optimization, Nanyang Technol. Univ., Singapore (May 2005)
[17] Suganthan, P.N., Hansen, N., Liang, J.J., Deb, K., Chen, Y.-P., Auger, A., Tiwari, S.: Problem definitions and evaluation criteria for the CEC2005 special session on real-parameter optimization. IIT, Kanpur, India, KanGAL Rep. 2005005 (May 2005)

A Fuzzy Programming Method for Solving Multiobjective Chance Constrained Programming Problems Involving Log-Normally Distributed Fuzzy Random Variables

Animesh Biswas[1,*] and Arnab Kumar De[2]

[1] Department of Mathematics, University of Kalyani
Kalyani – 741235, India
abiswaskln@rediffmail.com
[2] Department of Mathematics, Academy of Technology
G.T. Road, Adisaptagram, P.O.-Aedconagar
Dist – Hooghly, 712121, India
arnab7339@yahoo.co.in

Abstract. In this paper a fuzzy programming technique is presented to solve multiobjective chance constrained programming problem having the right sided parameters associated with system constrains follow log-normal distribution. In model formulation process the imprecise probabilistic problem is converted into an equivalent fuzzy programming model by applying chance constrained programming methodology. Then by considering fuzzy nature of parameters involved with the system constraints, the problem is decomposed on the basis of tolerance values of the parameters. The individual optimal value of each objective is found to construct the membership goals of the objectives. A priority based fuzzy goal programming approach is used for achievement of the highest membership degree to the extent possible under different priority structures to achieve the ideal point dependent solution in the decision making context. To expound the potentiality of the proposed approach, an illustrative example is solved and the solution is compared with other existing technique.

Keywords: Chance Constrained Programming, Log Normal Distribution, Fuzzy Numbers, Fuzzy Goal Programming, Fuzzy Random Variable.

1 Introduction

As a special field of mathematical programming, chance constrained programming deals with the situation where some or all parameters appear in the system constraints are probabilistic in nature and follow some probability distribution. Among the various types of probability distributions, log-normal distribution plays an important role for its wide applications in the fields engineering, earth sciences, medicines,

* Corresponding author.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 442–450, 2012.

radioactivity and other decision making arena. The genesis of log normal distribution is found during nineteenth century in the work of Weber [1], Galton [2], Fechner [3], and others.

In 1959, Charnes and Cooper [4] first introduced chance constrained programming (CCP) technique for solving stochastic programming problem with random parameters associated with the problems. Different aspects of CCP technique was further investigated by Kataoka [5], Prekopa [6] and other researcher in the past.

In formulating CCP model, the decision makers (DMs) are very often confused with the vague nature of parameters. To deal with such kind of vagueness involved with a multiobjective linear programming model, Zimmermann [7] introduced a fuzzy programming (FP) methodology based on fuzzy set theory [8]. The methodological development of that approach was further studied by Leberling [9], Sakawa [10] and others.

In the process of deriving models of CCP it is important to consider that the possible values of the random parameters under the occurrence of events as fuzzy numbers. Realizing the fact the concept of fuzzy random variables (FRVs) was first introduced by Kwakernaak [11]. FRVs [12] are considered as random variables whose values are represented by fuzzy numbers. The optimization model involving FRVs was first developed by Luhandjula [13]. Based on Lai and Hwang's [14] max-min approach, Sinha et al. [15] developed an FP approach for solving multiobjective probabilistic linear programming problems where only right side parameters of system constraints are random variables. These models were further studied by Liu [16], Rommelfanger [17], and others. A brief survey on the developments of stochastic programming models including FRVs is found in the work of Luhandjula [18].

Goal programming (GP), introduced by Charnes and Cooper [19], plays an important role in the field of multiobjective decision making (MODM) problems having conflicting objectives. The main drawback of classical GP is that the aspiration levels of the goals need to be specified precisely in making a decision. Also the classical GP technique cannot capture directly the uncertainty arises due to the presence of chance constraints associated with the problems. Under this context fuzzy goal programming (FGP) [20, 21] is used as an efficient tool for making decision in an imprecisely defined probabilistic MODM arena. FGP technique for solving CCP problems involving FRVs have been recently presented by Biswas and Modak [22, 23]. A genetic algorithm based approach in the framework of FGP for solving CCP has been investigated by Jana and Sharma [24].In this paper only discrete FRVs are considered. To the best of authors' knowledge an efficient solution technique for solving multiobjective CCP (MOCCP) following log normal distribution from the view point of its potential use in different planning problems involving fuzzy parameters is yet to appear in the literature.

This paper develops a methodology to solve fuzzy MOCCP (FMOCCP) problem consisting of log-normally distributed FRVs associated with right side values of the system constraints. Also some parameters of the system constraints are considered as fuzzy numbers. At first the problem is converted into an equivalent FP problem considering log-normal distribution of the parameters. Then considering the fuzzy aspects of the model, the problem is decomposed on the basis of their tolerance ranges of fuzzy parameters. The individual optimal value of each objective is derived to construct the membership function for measuring the degree of satisfaction of each of the

objective in the decision making situation. A priority based FGP model is used for achievement of the highest membership degree (unity) to the extent possible of each fuzzy goal defined for the membership functions under different priority structures. Finally developed model is solved to achieve the most satisfactory solution in the decision making context.

2 Prerequisites

Log normal distributions [1-3] are usually characterized in terms of log transformed variables whose parameters are represented by mean and standard deviation of its probability distribution. The log normal variates are those variates whose logarithm form follows normal distribution. By definition log-normal distributions are symmetrical at the log level whereas it is skewed in the ordinary level. A random variable satisfying log-normal distribution always considers only positive values and the distribution is skewed to the left. A density function for log normal distribution is presented in the following figure.

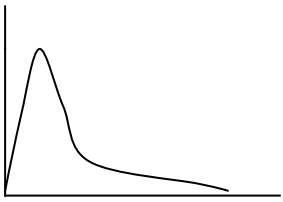


Fig. 1. Density curve of log-normal distribution

Let X be a log-normal variate. Then $ln(X)$ follows normal distribution. Considering two parameters, mean, μ, and standard deviation, σ, of $ln(X)$, the density function of log-normal distribution can be presented as $f(x) = \frac{1}{x\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2}(\log(x) - \mu)^2\right)$

The mean and variance of the distribution is given by $\exp\left(\mu + \frac{\sigma^2}{2}\right)$ and $\exp(2\mu + \sigma^2)\,(\exp(\sigma^2) - 1)$, respectively.

3 Formulation of FMOCCP Model

The generic form of a fuzzy CCP model having K number of objectives with a target to maximize the objectives under m number of system constraints is presented as

$$\begin{aligned} &\text{Find} \quad X(x_1, x_2, \ldots . x_n) \\ &\text{so as to} \quad \text{Maximize} \;\; Z_k = \textstyle\sum_{j=1}^{n} c_{kj}\, x_j \;;\; k = 1,2, \ldots \ldots, K \\ &\text{subject to} \;\; \Pr\left(\textstyle\sum_{j=1}^{n} \tilde{a}_{ij} x_j \le \tilde{b}_i\right) \ge 1 - \alpha_i \;;\;\; i = 1,2, \ldots \ldots, m \\ &\qquad\qquad x_j \ge 0 \;; j = 1,2, \ldots \ldots, n \end{aligned} \tag{1}$$

Where $\tilde{b}_i$ represent log-normally distributed FRVs and $\tilde{a}_{ij}$ are triangular fuzzy numbers, α_i denote any real number lying in the interval $[0, 1]$.

As $\tilde{b}_i$ is log-normally distributed FRV, the mean and variance of the fuzzy normal variate $\ln(\tilde{b}_i) = \tilde{d}_i$, are denoted by $E(\tilde{d}_i) = m_{\tilde{d}_i}$ and $Var(\tilde{d}_i) = \sigma^2_{\tilde{d}_i}$. Hence the mean and variance of the log-normal variates $\tilde{b}_i$ can be obtained as $E(\tilde{b}_i) = \exp\left(m_{\tilde{d}_i} + \frac{\sigma^2_{\tilde{d}_i}}{2}\right)$ and $Var(\tilde{b}_i) = \exp\left(2m_{\tilde{d}_i} + \sigma^2_{\tilde{d}_i}\right)(\exp\left(\sigma^2_{\tilde{d}_i}\right) - 1)$.

On the basis of the above considerations, the FP model is derived in the following subsection.

3.1 FP Model Construction

Now considering only chance constraints defined in Model (1), the following derivations are made. The chance constraint is represented here as

$$\Pr\left(\sum_{j=1}^{n} \tilde{a}_{ij}x_j \le \tilde{b}_i\right) \ge 1 - \alpha_i; \quad \text{i.e.,} \quad \Pr\left(\tilde{b}_i \le \sum_{j=1}^{n} \tilde{a}_{ij}x_j\right) \le \alpha_i;$$

$$\text{i.e., } \Pr\left(\frac{\tilde{d}_i - m_{\tilde{d}_i}}{\sigma_{\tilde{d}_i}} \le \frac{\ln\left(\sum_{j=1}^{n} \tilde{a}_{ij}x_j\right) - m_{\tilde{d}_i}}{\sigma_{\tilde{d}_i}}\right) \le \alpha_i \text{ i.e., } \Phi\left(\frac{\ln\left(\sum_{j=1}^{n} \tilde{a}_{ij}x_j\right) - m_{\tilde{d}_i}}{\sigma_{\tilde{d}_i}}\right) \le \alpha_i;$$

$$\text{i.e., } \ln\left(\sum_{j=1}^{n} \tilde{a}_{ij}x_j\right) \le m_{\tilde{d}_i} + \Phi^{-1}(\alpha_i)\sigma_{\tilde{d}_i}.$$

Hence the Model (1) is converted into the following form as

$$\begin{aligned} &\text{Find} \quad X(x_1, x_2, \ldots . x_n) \\ &\text{so as to} \quad \text{Maximize } Z_k = \sum_{j=1}^{n} c_{kj}\, x_j \;;\; k = 1,2, \ldots \ldots, K \\ &\text{subject to} \quad \sum_{j=1}^{n} \tilde{a}_{ij}x_j \le \exp\left(m_{\tilde{d}_i} + \Phi^{-1}(\alpha_i)\sigma_{\tilde{d}_i}\right) ; i = 1,2, \ldots \ldots, m \\ &\qquad\quad x_j \ge 0 \;;\; j = 1,2, \ldots \ldots, n \end{aligned} \tag{2}$$

Here the function $\Phi(\cdot)$ represents the cumulative distribution function of the standard normal fuzzy random variate.

Characterization of Membership Functions. In fuzzy decision making situation, it is assumed that the mean and standard deviation associated with the FRVs $\tilde{d}_i$ and the fuzzy parameters $\tilde{a}_{ij}$ are triangular fuzzy numbers. A triangular fuzzy number can be represented by a triple of three real numbers as $\tilde{a} = (a^L, a, a^R)$, the membership function of whose takes the form

$$\mu_{\tilde{a}}(x) = \begin{cases} 0 & if\ x < a^L\ or\ x > a^R \\ \frac{x - a^L}{a - a^L} & if\ a^L \le x \le a \\ \frac{a^R - x}{a^R - a} & if\ a \le x \le a^R \end{cases},$$

where a^L and a^R denote, respectively, the left and right tolerance values of the fuzzy number $\tilde{a}$.

Now, considering mean, $m_{\tilde{d}_i}$, and standard deviation, $\sigma_{\tilde{d}_i}$, of the FRV, $\tilde{d}_i$, and parameters, $\tilde{a}_{ij}$, associated with the system constraints in (2) as triangular fuzzy numbers of the form

$m_{\tilde{a}_i} = (m_{d_i}^L, m_{d_i}, m_{d_i}^R), \sigma_{\tilde{a}_i} = (\sigma_{d_i}^L, \sigma_{d_i}, \sigma_{d_i}^R), \tilde{a}_{ij} = (a_{ij}^L, a_{ij}, a_{ij}^R)$; for $i = 1,2,\ldots,m; j = 1,2,\ldots,n,$ the Model (2) can be described as

$$\begin{aligned} &\text{Find} \quad X(x_1, x_2, \ldots . x_n) \\ &\text{so as to} \quad \text{Maximize } Z_k = \textstyle\sum_{j=1}^{n} c_{kj} x_j; \ k = 1,2,\ldots K \\ &\text{subject to} \end{aligned}$$

$$\begin{aligned} &\textstyle\sum_{j=1}^{n} \{a_{ij}^L + (a_{ij} - a_{ij}^L)\beta\} x_j \le \exp\left(m_{d_i}^L + \left(m_{d_i} - m_{d_i}^L\right)\beta + \Phi^{-1}(\alpha_i)(\sigma_{d_i}^L + (\sigma_{d_i} - \sigma_{d_i}^L)\beta)\right) \\ &\textstyle\sum_{j=1}^{n} \{a_{ij}^R - (a_{ij}^R - a_{ij})\beta\} x_j \le \exp\left(m_{d_i}^R - \left(m_{d_i}^R - m_{d_i}\right)\beta + \Phi^{-1}(\alpha_i)(\sigma_{d_i}^R - (\sigma_{d_i}^R - \sigma_{d_i})\beta)\right) \\ &0 \le \beta \le 1 \ ; i = 1,2,\ldots,m \ ; j = 1,2,\ldots,n \ . \end{aligned} \tag{3}$$

Here β represents the degree of fuzziness involved with parameters of the system constraints. Now, considering the ambiguous nature of human judgments, it is quite natural to assume that the DM may have a fuzzy goal for each of the objective functions. In a maximization type problem, the DM specifies the fuzzy goal in such a manner that the objective function value should be substantially greater than or equal to some assigned value. To obtain the aspiration level to the fuzzy goals, each objective is solved individually under the system constraints defined in Model (3). Let Z_k^b and $Z_k^w; k = 1,2,\ldots,K$ be the best and worst values obtained by solving each objective independently. Hence the fuzzy objective goal for each of the objectives can be expressed as:

$$Z_k \gtrsim Z_k^b \quad \text{for } k = 1,2,\ldots,K$$

Such a fuzzy goal can be quantified by eliciting the corresponding membership functions on the basis of the achieved values. Thus the membership function for each of the objectives can be written as :

$$\mu_{Z_k}(x) = \begin{cases} 0 & if \quad Z_k \le Z_k^w \\ \frac{Z_k - Z_k^w}{Z_k^b - Z_k^w} & if \quad Z_k^w \le Z_k \le Z_k^b \\ 1 & if \quad Z_k \ge Z_k^b \end{cases}; k = 1,2,..,K \tag{4}$$

3.2 Priority Based FGP Model Formulation

In a priority based FGP model, the elicited membership functions are considered as flexible goals by introducing under- and over- deviational variables to each of them and assigning unity as the aspiration level. The full achievement of all the membership goals is not possible in an MODM context. Also the objectives of the model are not equally important to achieve the ideal point dependent solution in the decision making arena. So the under deviational variables are minimized on the basis of priority of importance of achieving the goal values of objectives in the decision making environment. The priority based FGP model can be formulated as

$$\begin{aligned} &\text{Find} \quad X(x_1, x_2, \ldots . x_n) \\ &\text{so as to} \quad \text{Min D} = \lfloor P_1(d^-), P_2(d^-), \ldots P_I(d^-) \rfloor \\ &\text{and satisfy} \quad \frac{Z_k - Z_k^w}{Z_k^b - Z_k^w} + d_k^- - d_k^+ = 1 \text{ for } k = 1,2,\ldots,K \\ &\textstyle\sum_{j=1}^{n} \{a_{ij}^L + (a_{ij} - a_{ij}^L)\beta\} x_j \le \exp\left(m_{d_i}^L + \left(m_{d_i} - m_{d_i}^L\right)\beta + \Phi^{-1}(\alpha_i)(\sigma_{d_i}^L + (\sigma_{d_i} - \sigma_{d_i}^L)\beta)\right) \end{aligned}$$

$$\sum_{j=1}^{n}\{a_{ij}^{R}-(a_{ij}^{R}-a_{ij})\beta\}x_j \le \exp\left(m_{d_i}^{R}-\left(m_{d_i}^{R}-m_{d_i}\right)\beta+\Phi^{-1}(\alpha_i)(\sigma_{d_i}^{R}-(\sigma_{d_i}^{R}-\sigma_{d_i})\beta)\right)$$

$$0 \le \beta \le 1;\ i = 1,2,\dots,m\ ;\ j = 1,2,\dots,n$$

$$d_k^{-}\ , d_k^{+} \ge 0\ \ ;d_k^{-}.d_k^{+} = 0 \tag{5}$$

where D represent I priority achievement function consisting of the weighted under deviational variables, d_k^-. The priority factors have the relationship

$$P_1 >>> P_2 >>> \cdots P_i >>> \cdots >>> P_I$$

which implies that the goals at the highest priority level will achieve the goal values to the extent possible before considering the achievement of the goals at the lower priority levels. The priority function $P_i(d^-)$ has the form:

$$P_i(d^-) = \sum_{k=1}^{K} w_{ik}^{-}\, d_k^{-},$$

where $w_{ik}^{-} \ge 0$ represents the numerical weights of importance of the goals at the i-th priority level P_i, and which are determined as:

$$w_{ik}^{-} = \frac{1}{(z_k^b - z_k^w)}.$$

The developed model (5) is solved to find the most satisfactory solution in the decision making environment.

4 An Illustrative Example

To demonstrate the feasibility and efficiency of the developed approach, the following FMOCCP with log-normally distributed FRVs is considered as

$$\begin{aligned}
&\text{Find } X(x_1, x_2)\\
\text{so as to}\quad &\text{Maximize } Z_1 = 10x_1 + 5x_2\\
&\text{Maximize } Z_2 = 3x_1 + 7x_2\\
\text{subject to}\quad &Pr\left[\tilde{1}x_1 + \tilde{1}x_2 \le b_1\right] \ge .94\ , Pr\left[\tilde{4}x_1 + \tilde{3}x_2 \le b_2\right] \ge .93,\\
&Pr\left[\tilde{2}x_1 + \tilde{5}x_2 \le b_3\right] \ge .91 \quad x_1\ ,x_2\ \ge 0
\end{aligned} \tag{6}$$

Here $\tilde{b}_1, \tilde{b}_2, \tilde{b}_3$ are independent FRVs following log-normal distribution and other parameters are fuzzy numbers. Let $\tilde{d}_i = ln\left(\tilde{b}_i\right), i = 1,2,3$ are independent FRVs which follow normal distribution.

Considering mean and standard deviations of $\tilde{d}_i$; and applying general CCP technique the model (6) is converted into the following form as

$$\begin{aligned}
&\text{Find } X(x_1, x_2)\\
\text{so as to}\quad &\text{Maximize } Z_1 = 10x_1 + 5x_2\\
&\text{Maximize } Z_2 = 3x_1 + 7x_2\\
\text{subject to}\quad &\tilde{1}x_1 + \tilde{1}x_2 \le \exp(m_{\tilde{d}_1} + 0.15\sigma_{\tilde{d}_1}),\\
&\tilde{4}x_1 + \tilde{3}x_2 \le \exp(m_{\tilde{d}_2} + 0.18\sigma_{\tilde{d}_2}),\\
&\tilde{2}x_1 + \tilde{5}x_2 \le \exp\left(m_{\tilde{d}_3} + 0.23\sigma_{\tilde{d}_3}\right),\ x_1, x_2 \ge 0
\end{aligned} \tag{7}$$

Here, the mean and standard deviation of $\tilde{d}_i$, $i = 1,2,3$ are assumed as triangular fuzzy numbers with the following forms

$m_{\tilde{d}_1} = (-.05,0,.05), \sigma_{\tilde{d}_1} = (1.95,2,2.05)$; $m_{\tilde{d}_2} = (1.65,1.7,1.75)$, $\sigma_{\tilde{d}_2} = (.95,1,1.05)$ and $m_{\tilde{d}_3} = (1.55,1.6,1.65)$, $\sigma_{\tilde{d}_3} = (.95,1,1.05)$.

Also the coefficients $\tilde{1}, \tilde{4}, \tilde{3}, \tilde{2}, \tilde{5}$ are taken as triangular fuzzy numbers with the form

$\tilde{1} = (.98,1,1.08), \tilde{1} = (.95,1,1.05), \tilde{4} = (3.95,4,4.05), \tilde{3} = (2.98,3,3.02)$, $\tilde{2} = (1.95,2,2.05)$ and $\tilde{5} = (4.95,5.5.05)$

On the basis of fuzzily defined mean and variance of the FRVs and triangular fuzzy number the problem is decomposed as

$$
\begin{aligned}
&\text{Find } X(x_1, x_2)\\
&\text{so as to} \quad \text{Maximize } Z_1 = 10x_1 + 5x_2\\
&\qquad\quad \text{Maximize } Z_2 = 3x_1 + 7x_2\\
&\text{subject to}\\
&(0.98 + 0.02\beta)x_1 + (0.95 + 0.05\beta)x_2 \le \exp\big((-0.05 + 0.05\beta) + 0.15(1.95 + 0.05\beta)\big)\\
&(1.08 - 0.08\beta)x_1 + (1.05 - 0.05\beta)x_2 \le \exp\left((0.05 - 0.05\beta) + 0.15(2.05 - 0.05\beta)\right)\\
&(3.95 + 0.05\beta)x_1 + (2.98 + 0.02\beta)x_2 \le \exp\left((1.65 + 0.05\beta) + 0.18(0.95 + 0.05\beta)\right)\\
&(4.05 - 0.05\beta)x_1 + (3.02 - 0.02\beta)x_2 \le \exp\left((1.75 - 0.05\beta) + 0.18(1.05 - 0.05\beta)\right)\\
&(1.95 + 0.05\beta)x_1 + (4.95 + 0.05\beta)x_2 \le \exp\left((1.55 + 0.05\beta) + 0.23(0.95 + 0.05\beta)\right)\\
&(2.05 - 0.05\beta)x_1 + (5.05 - 0.05\beta)x_2 \le \exp\left((1.65 - 0.05\beta) + 0.23(1.05 - 0.05\beta)\right)\\
&\qquad x_1, x_2 \ge 0\ ,\ 0 \le \beta \le 1\ ; \qquad (8)
\end{aligned}
$$

Now each objective is considered independently and is solved with respect to the system of constraints in (8) to find the individual optimal values of the objectives.

The results are obtained as $Z_1^b = 14.60390$, $Z_1^w = 0, Z_2^b = 8.761802, Z_2^w = 0$.

Then the fuzzy goals of the objectives are found as: $Z_1^b \gtrsim 14.60390, Z_2^b \gtrsim 8.761802$.

Hence the FGP model under preemptive priority structure can be presented by elicited the membership goals as

$$
\begin{aligned}
&\text{Find } X(x_1, x_2)\\
&\text{so as to} \quad \text{Min D} = \lfloor P_1(0.0685d_1^-), P_2(0.1141d_2^-)\rfloor\\
&\text{and satisfy} \quad \mu_{f_1}: \frac{10x_1 + 5x_2}{14.60390} + d_1^- - d_1^+ = 1\\
&\qquad\qquad\quad \mu_{f_2}: \frac{3x_1 + 7x_2}{8.761802} + d_2^- - d_2^+ = 1\\
&\text{subject to the system constraints defined in (8)} \qquad (9)\\
&\text{with } d_1^-, d_1^+, d_2^-, d_2^+ \ge 0 \text{ with } d_1^-.d_1^+, d_2^-.d_2^+ = 0
\end{aligned}
$$

Now the Model (9) is executed to find the most satisfactory solution in the decision making context. The software LINGO (ver 13.0) is used to solve the problem.

The resultant decision is $x_1 = 0.870, x_2 = 0.616$ with the objective values $Z_1 = 11.778, Z_2 = 6.919$.

The achieved objective values obtained by using the technique developed by Sinha et al. [15] is $Z_1 = 10.553$, $Z_2 = 6.406$.

The result shows the superiority of the developed methodology in terms of achieving the objective values over the other technique.

5 Conclusions

This article introduces a new methodology for solving FMOCCP involved with log-normally distributed FRVs and fuzzy numbers by using priority based FGP technique. The method captures both types of uncertainties like fuzziness and randomness simultaneously. The presented model is flexible enough to introduce the tolerance limits of the fuzzy goals initially to arriving at the compromise decisions on the basis of priorities of importance of the objectives. The method can also be adapted to hierarchical decision making environment with multiplicity of objectives. However it is hoped that the proposed methodology may open up new vistas into the way of making decision in a fuzzily defined probabilistic decision making arena.

Acknowledgements. The authors are thankful to the anonymous reviewers for their valuable comments and suggestions to improve the clarity and quality of the paper.

References

1. Weber, H.: De Pulsa Resorptione Auditu et Tactu. Annotationes Anatomicae et Physiologicae. Koehler, Leipzig (1834)
2. Galton, F.: The geometric mean in vital and social statistics. Proceedings of the Royal Society 29, 365–367 (1879)
3. Fechner, G.T.: Kollektivmasslehre. Engelmann, Leipzig (1897)
4. Charnes, A., Cooper, W.W.: Chance Constrained Programming. Mgmt. Sci. 6, 73–79 (1959)
5. Kataoka, S.: A stochastic programming model. Econometrica 31, 181–196 (1963)
6. Prekopa, A.: Contribution to the Theory of Stochastic Programming. Mgmt. Sci. 4, 202–221 (1973)
7. Zimmermann, H.J.: Fuzzy programming and linear programming with several objective functions. Fuzzy Sets and Systs. 1, 45–55 (1978)
8. Zadeh, L.A.: Fuzzy sets. Inform. Control. 8, 338–353 (1965)
9. Leberling, H.: On finding compromise solution in multicriteria problems using the fuzzy min-operator. Fuzzy Sets and Systs. 6, 105–118 (1980)
10. Sakawa, M.: Fuzzy Sets and Interactive Multiobjective Optimization. Plenum Press, New York (1993)
11. Kwakernaak, H.: Fuzzy random variable–I. Definitions and theorems. Inform. Sci. 15, 1–29 (1978)
12. Shapiro, A.E.: Fuzzy random variables. Insurance: Math. and Econ. 44, 307–314 (2009)
13. Luhandjula, M.K.: Fuzziness and randomness in an optimization framework. Fuzzy Sets and Systs. 77, 291–297 (1996)
14. Lai, Y.J., Hwang, C.L.: Fuzzy Multiple Objective Decision Making: Methods and Applications. Springer, Berlin (1994)

15. Sinha, S.B., Biswal, M.P., Hulsurkar, S.: Fuzzy programming approach to multiobjective probabilistic linear programming problems where only b_i's are probabilistic. J. Fuzzy Math. 5, 63–73 (1998)
16. Liu, B.: Fuzzy random chance-constrained programming. IEEE Trans. Fuzzy Syst. 9, 713–720 (2001)
17. Rommelfanger, H.: A general concept for solving linear multicriteria programming problems with crisp, fuzzy, stochastic variables. Fuzzy Sets and Systs. 158, 1892–1904 (2007)
18. Luhandjula, M.K.: Fuzzy Stochastic linear programming: survey and future research directions. Euro. J. Opl. Res. 174, 1353–1367 (2006)
19. Charnes, A., Cooper, W.W.: Goal programming and multiple objective optimizations. Euro. J. Opl. Res. 1, 39–54 (1977)
20. Hannan, E.L.: Linear programming with multiple fuzzy goals. Fuzzy Sets and Systs. 6, 235–248 (1980)
21. Narasimhan, R.: On fuzzy goal programming – Some comments. Dec. Sci. 11, 532–538 (1980)
22. Biswas, A., Modak, N.: Using fuzzy goal programming technique to solve multiobjective chance constrained programming problems in a fuzzy environment. Int. J. Fuzzy Syst. Appls. 2, 71–80 (2012)
23. Biswas, A., Modak, N.: A fuzzy goal programming method for solving chance constrained programming with fuzzy parameters. Comm. Comp. Inform. Sci. 140, 187–196 (2011)
24. Jana, R.K., Sharma, D.K.: Genetic algorithm-based fuzzy goal programming for class of chance-constrained programming problems. Int. J. Comp. Math. 87, 733–742 (2010)

Minimization of Side Lobe of Optimized Uniformly Spaced and Non-uniform Excited Time Modulated Linear Antenna Arrays Using Genetic Algorithm

Gopi Ram Hardel[1], Durbadal Mandal[1], Sakti Prasad Ghoshal[2], and Rajib Kar[1]

[1] Department of Electronics and Communication Engineering, National Institute of Technology, Durgapur, West Bengal, India
[2] Department of Electrical Engineering, National Institute of Technology, Durgapur, West Bengal, India
{gopi203hardel,durbadal.bittu, spghoshalnitdgp,rajibkarece}@gmail.com

Abstract. In this paper optimal side lobe reduction is achieved for time modulated linear antenna arrays with optimized uniform inter-element spacing and non-uniform excitations. An evolutionary approach, RGA, is proposed for the optimization. The side lobe level of time modulated linear array can be reduced significantly by optimal rearrangement of static excitation amplitudes and inter-element spacing of each element by RGA. The approach is illustrated through 12-, 16-, 20- element time modulated linear antenna arrays. Various results are presented to show the advantage of this approach considering maximum side lobe reduction.

1 Introduction

Time modulated linear antenna arrays have attracted the antenna designers for their advantages over conventional array antennas for a few years. Previous research works have shown time modulated linear antenna arrays are attractive for the synthesis of low/ultralow side lobes [3-8]. As compared to a conventional antenna array, the time modulated antenna array introduces a fourth dimension—time—into the design. Consequently, it has more flexibility [11-19]. In this paper, real coded genetic algorithm (RGA) is adopted to realize optimized non-uniform current excitation weights and uniform inter-element spacing for time modulated linear antenna arrays to achieve maximum side lobe reduction. The time modulation period is divided into numerous minimal time steps with the same length, where the ON-OFF status for each time step is followed by a novel scheme designed by the authors given in the paper.

In this work, broadside time modulated linear antenna arrays with non-uniform amplitude distribution and uniform spacing are considered. The phase difference between any two elements is kept zero. The excitation and inter-element spacing of each element are optimized using RGA. A cost function is defined, which keeps the SLL at low levels. Section 2 deals with the theoretical background of time modulation

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 451–458, 2012.

technique. This is followed by description of RGA in section 3; Numerical Results and convergence curves in section 4; and conclusion in section 5.

2 Design Equation

Consider a broadside linear array of 2M equally spaced isotropic elements as shown in Fig. 1, in which each element is controlled by a high speed RF switch and excited with complex amplitude. The array is symmetric in both geometry and excitation with respect to the array center.

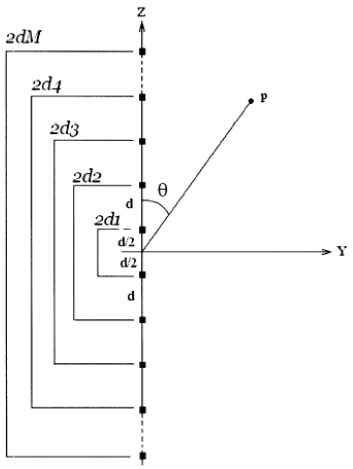


Fig. 1. Geometry of a 2M-element linear array along the z-axis

The array is used to transmit a rectangular pulse of width T , with a pulse repetition frequency, $prf = 1/T_p$, and T_p is the pulse repetition period. For broadside beams, the array factor is given by

$$AF(\theta,t) = 2\sum_{n=1}^{M} U_n(t) I_n \cos[(\frac{2n-1}{2})kd\cos(\theta)] \tag{1}$$

where

$\theta =$ Angle of radiation of electromagnetic plane wave;
$d =$ Spacing between elements;
$k =$ Propagation constant;
$2M =$ Total number of elements in the array;
$I_n =$ Excitation amplitude of n th element;
$U_n =$ Corresponding periodic switch on time.

Switching function is given by (2) and shown in Fig. 2; each is switched 'ON' for $\tau_n (0 \le \tau_n \le T)$.

$$U_n(\tau_n) = 1 \qquad \text{if } \tau_n \le n * \frac{T}{M}$$
$$= 0 \qquad \text{Else} \tag{2}$$

where τ_n is the time for which n^{th} element will be 'ON' and T is the pulse repetition time. All the antenna elements are assumed isotropic. Only amplitude excitations of each element and inter-element spacing are used to change the antenna radiation pattern. Fig. 2 show the 'ON'-'OFF' time sequence for time modulated linear antenna array of M=10 elements.

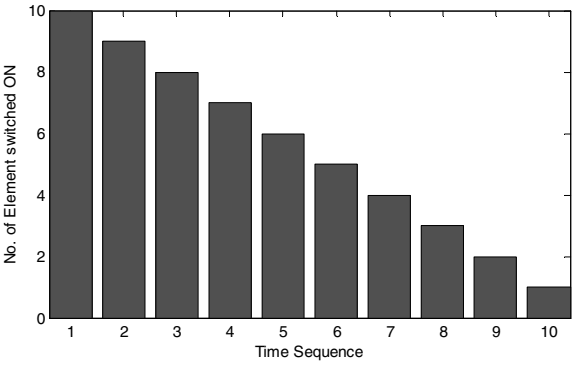


Fig. 2. Time Sequence of M=10 time modulated linear antenna array

The cost function (CF) for reducing the side lobe level is given below

$$CF = \left|AF(\theta_{msl}, I_n)\right| / \left|AF(\theta_0, I_n)\right| \tag{3}$$

where θ_0 is the angle where the highest maximum of central angle is attained in $\theta \in [0, \pi]$. θ_{msl} is the angle where maximum side lobe $AF(\theta_{msl}, I_n)$ is attained on either side of main beam. Minimization of CF means maximum reduction of SLL. RGA technique is employed for optimizing non-uniform current excitation weights and inter-element spacing. Results of the minimization of CF and SLL are described in section 4.

3 Evolutionary Techniques Employed

GA is mainly a probabilistic search technique, based on the principles of natural selection and evolution. At each generation it maintains a population of individuals where each individual is a coded form of a possible solution of the problem at hand and called chromosome. Chromosomes are constructed over some particular alphabet,

e.g., the binary alphabet {0, 1}, so that chromosomes' values are uniquely mapped onto the decision variable domain. Each chromosome is evaluated by a function known as fitness function, which is usually the objective function of the corresponding optimization problem.

Steps of RGA as implemented for optimization of non-uniform current excitation weights and inter-element spacing are [2, 3]:

- Initialization of real chromosome strings of n_p population, each consisting of a set of current excitation weights (M) and uniform inter-element (01). Size of the set is M+1, in a particular M-element array design.
- Decoding of strings and evaluation of CF of each string.
- Selection of elite strings in order of increasing CF values from the minimum value.
- Copying of the elite strings over the non-selected strings.
- Crossover and mutation to generate off-springs.
- Genetic cycle updating.
- The iteration stops when the maximum number of cycles is reached. The grand minimum CF and its corresponding chromosome string or the desired optimal solution of M number of current excitation weights and one number of uniform inter-element spacing are finally obtained.

4 Experimental Results

This section gives the experimental results for three number of time modulated linear antenna array designs obtained by RGA technique. For each time modulated linear antenna array, uniform inter-element spacing is maintained. The best parameters for the RGA are set after many trial runs. It is found that the best results are obtained for the initial population size (n_p) of 120 chromosomes; and maximum number of generations, Nm as 400. For selection operation, the method of natural selection is chosen with selection probability of 0.3. Crossover is randomly selected as dual point. Crossover ratio is 0.8. Mutation probability is 0.15. RGA generates a set of optimal normalized non-uniform current excitation weights and optimal uniform inter-element spacing $\left(d \in [\lambda/2, \lambda]\right)$ for each set of time modulated linear antenna arrays. Sets of time modulated linear antenna arrays considered are of 12-, 16- and 20-elements. Tables 2 show the optimal results. Tables 1 depict SLL values and BWFN values for all corresponding uniformly excited time modulated linear antenna arrays.

4.1 Analysis of Radiation Patterns

Figs. 3-5 depict the optimal radiation patterns of 12-, 16- and 20-element time modulated linear antenna array sets, respectively, with optimum non-uniform excitation weights and optimum uniform inter-element spacing using RGA. From each figure, it is clearly visible that beside noticeable reduction of SLL, BWFN is also well restricted upon optimizing. As seen from the Tables 2, for optimal non-uniformly

excited and optimal uniformly spaced symmetric time modulated 12-element, 16-element and 20-element, linear antenna arrays, SLL reduces to -23.66 dB, -20.72 dB and -18.26 dB, respectively, against -5.424 dB, -5.261 dB and -5.148 dB, respectively, for corresponding uniform time modulated linear arrays. For the 12-element case, Nulls at first and third peak are achieved. Fig. 4 depicts wide Nulls for the case of 16- element. And finally, for the case of 20- element, Nulls at fourth side lobe peak are achieved.

4.2 Convergence Profiles of RGA

The minimum *CF* values against number of iteration cycles are recorded to get the convergence profile for each set. Fig. 6 portrays the convergence profiles of minimum CF of time modulated 12- and 16-element linear antenna array sets with improved SLL, respectively. Simulation programming was done in MATLAB language using MATLAB 7.5 on dual core(TM) processor, 2.88 GHz with 2 GB RAM.

Table 1. Initial values of SLL and BWFN for uniformly excited arrays having (I_n =1) and $\lambda/2$ inter-element spacing

Set No.	No. of Element	SLL(dB)	BWFN (deg.)
I	12	-5.424	14.76
II	16	-5.261	10.80
III	20	-5.148	8.64

Table 2. Optimal current excitation coefficients, optimal inter-element spacing, SLL and BWFN for three time modulated linear antenna array sets

Set No.	$I_1, I_2 \ldots\ldots I_M$		Optimized uniform inter-element spacing (λ)	SLL (dB)	BWFN(deg)
I	0.9505	0.4601	0.8289	-23.66	16.56
	0.2861	0.1297			
	0.0904	0.0459			
II	0.9343	0.7132	0.9049	-20.72	10.80
	0.3945	0.2007			
	0.1758	0.1144			
	0.0602	0.0306			
III	0.9894	0.9636	0.7951	-18.26	8.64
	0.5022	0.4160			
	0.2608	0.2120			
	0.1446	0.0923			
	0.0667	0.0612			

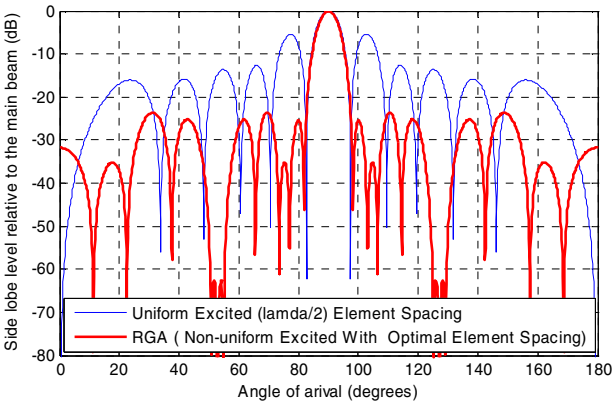


Fig. 3. Best array pattern found by RGA for the 12-element array with improved SLL

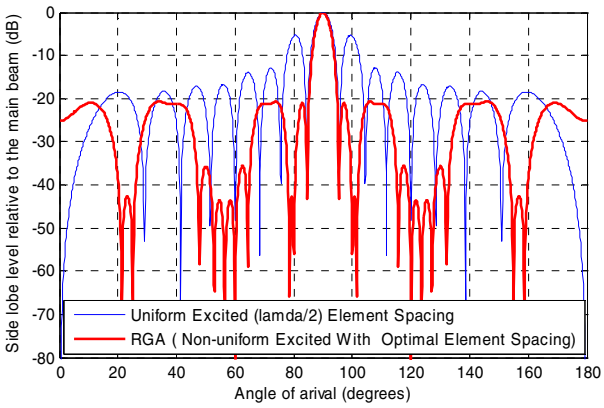


Fig. 4. Best array pattern found by RGA for the 16-element array with improved SLL

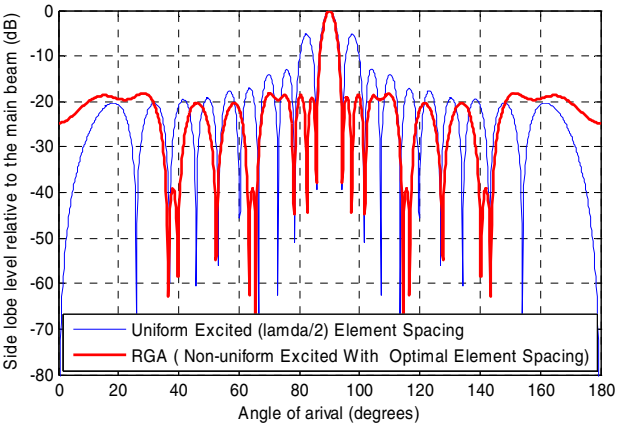


Fig. 5. Best array pattern found by RGA for the 20-element array with improved SLL

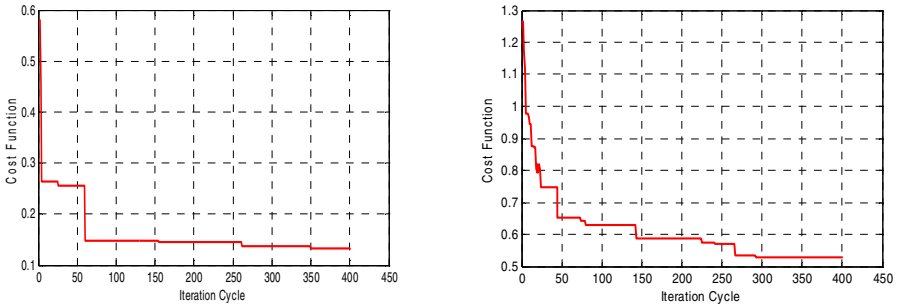


Fig. 6. Convergence profile of RGA for the (a) 12-element and (b) 16-element time modulated linear antenna array with improved SLL

5 Conclusions

In this paper the optimal design of non-uniformly excited time modulated linear antenna arrays with uniform inter-element spacing has been described using the technique of real coded genetic algorithm. Experimental results reveal that the optimal design of non-uniformly excited time modulated linear antenna arrays with optimal inter-element spacing offers a considerable SLL reduction with respect to corresponding time modulated uniform linear arrays with uniform inter-element spacing of $\lambda/2$. For the time modulated linear antenna array sets having 12, 16 and 20 elements, SLLs have reduced to -23.66 dB, -20.72 dB and, -18.26 dB, respectively, against -5.424 dB, -5.261 dB and -5.148 dB, respectively, of corresponding uniform time modulated linear arrays with a very little change in BWFN. Figs. 3-5 depict additional achievement of deeper nulls and wide nulls. The authors' contributions in this paper are: (i) the application of evolutionary algorithm in the optimal design and (ii) the novel design of time switching function. Other time switching functions and other antenna geometries and constraints in many different areas of antenna design will be considered in future research for tuning antenna characteristics and parameters. Genetic algorithm seems a good candidate to deal with the problems.

References

1. Balanis, C.A.: Antenna Theory Analysis and Design. John Wiley & Sons, New York (1997)
2. Haupt, R.L., Werner, D.H.: Genetic Algorithms in Electromagnetics. IEEE Press, Wiley-Interscience (2007)
3. Haupt, R.L.: Phase-only adaptive nulling with a genetic algorithm. IEEE Trans. Antennas Propagat. 45(6), 1009–1015 (1997)
4. Yang, S., Gan, Y.B., Qing, A.: Sideband suppression in time-modulated linear arrays by the differential evolution algorithm. IEEE Antennas Wireless Propag. Lett. 1, 173–175 (2002)

5. Yang, S., Gan, Y.B., Tan, P.K.: Linear antenna arrays with bidirectional phase center motion. IEEE Trans. Antennas Propagat. 53(5), 1829–1835
6. Yang, S., Gan, Y.B., Qing, A., Tan, P.K.: Design of a uniform amplitude time modulated linear array with optimized time sequences. IEEE Trans. Antennas Propag. 53(7), 2337–2339 (2005)
7. Yang, S., Gan, Y.B., Tan, P.K.: A new technique for power-pattern synthesis in time-modulated linear arrays. IEEE Antennas Wireless Propag. Lett. 2, 285–287 (2003)
8. Yang, S., Gan, Y.B., Tan, P.K.: Comparative study of low sidelobe time modulated linear arrays with different time schemes. J. Electromagn. Waves Appl. 18(11), 1443–1458 (2004)
9. Yang, S., Gan, Y.B., Qing, A.: Moving phase center antenna arrays with optimized static excitations. Microw. Opt. Tech. Lett. 38, 83–85
10. Schrank, H.E.: Low sidelobe phased array antennas. IEEE Antennas Propagat. Soc. Newslett. 25(2), 4–9 (1983)
11. Fondevila, J., Brégains, J.C., Ares, F., Moreno, E.: Optimizing uniformly excited linear arrays through time modulation. IEEE Antennas Wireless Propag. Lett. 3, 298–301 (2004)
12. Yang, S., Gan, Y.B., Tan, P.K.: A new technique for power-pattern synthesis in time modulated linear arrays. IEEE Antennas Wireless Propag. Lett. 2, 285–287 (2003)
13. Fondevila, J., Brégains, J.C., Ares, F., Moreno, E.: Application of time modulation in the synthesis of sum and difference patterns by using linear arrays. Microw. Opt. Technol. Lett. 48(5), 829–832 (2006)
14. Li, G., Yang, S., Nie, Z.: A study on the application of time modulated antenna arrays to airborne pulsed radar. IEEE Trans. Antenna Propag. 57(5), 1578–1582 (2009)
15. Tannat, A., Chambers, B.: A two element time modulated arrays with direction finding properties. IEEE Antennas Wireless Propag. Lett. 6, 64–65 (2007)
16. Bregains, J.C., Fondevila-Gomez, J., Franceschetti, G., Ares, F.: Signal radiation and power losses of time modulated arrays. IEEE Trans. Antennas Propag. 56(6), 1799–1804 (2008)
17. Li, G., Yang, S., Huang, M., Nie, Z.: Sidelobe suppression in time modulated linear arrays with unequal element spacing. J. Electromagn. Wave Appl. 24(5/6), 775–783 (2010)
18. Li, G., Yang, S., Chen, Y., Nie, Z.: Anovel electronic beam steering technique in time modulated antenna arrays. Prog. Electromagn. Res., PIER 97, 391–405 (2009)
19. Manica, L., Rocca, P., Poli, L., Massa, A.: Almost time-independent performance in time-modulated linear arrays. IEEE Antenna Wireless Propag. Lett. 8, 843–846 (2009)
20. Hardel, G.R., Yalapragada, N.T., Mandal, D., Bhattacharjee, A.K.: Introducing Dipper Nulls in Time modulated Linear symmetric Antenna Array Using Real Coded Genetic Alorithm. In: IEEE Symposiumon Computers and Informatics (ISCI 2011), pp. 249–254 (March 2011)
21. Mandal, D., Ghoshal, S.P., Bhattacharjee, A.K.: Design of Concentric Circular Antenna Array With Central Element Feeding Using Particle Swarm Optimization With Constriction Factor and Inertia Weight Approach and Evolutionary Programing Technique. Journal of Infrared Milli Terahz Waves 31(6), 667–680 (2010)
22. Mandal, D., Ghoshal, S.P., Bhattacharjee, A.K.: A Novel Particle Swarm Optimization Based Optimal Design of Three-Ring Concentric Circular Antenna Array. In: IEEE International Conference on Advances in Computing, Control, and Telecommunication Technologies, ACT 2009, pp. 385–389 (2009)
23. Haupt, R.L.: Adaptive nulling in monopulse antennas. IEEE Trans. Antennas Propagat. 36(2), 202–208 (1988)

Circular Antenna Array Design Using Novel Perturbation Based Artificial Bee Colony Algorithm

Digbalay Bose[1], Souvik Kundu[1], Subhodip Biswas[1], and Swagatam Das[2]

[1] Faculty of Engineering & Technology, Jadavpur University, Kolkata 700 032, India
[2] Indian Statistical Institute, Kolkata – 700 108, India
{digbose92,sk210892,subho.opto.placis91}@gmail.com,
swagatam.das@isical.ac.in

Abstract. The field of optimization is abundant with algorithms, which are inspired from nature based phenomena. The increasing popularity of such algorithms stems from their applications in real life situations. Here in this article a real life problem in the form of the design of circular antenna array has been discussed. The design of the antenna array is based on the application of a novel variant of Artificial Bee Colony Algorithm using selective neighborhood called sNABC**.** We use a neighborhood based perturbation on the basis of Euclidean distance and fitness of individuals are used for obtaining minimum side lobe levels, maximum directivity and appropriate null control. To illustrate the effectiveness of our design procedure, the results have been compared with several existing algorithms like DE, ABC and PSO**.**

1 Introduction

In this modern era where long distance communication is highly prevalent, antennas possessing greater directive radiation nature are desirable. For achieving such a directive radiation pattern antenna arrays [1-3] are used in place of a single antenna, whose elements of the are combined in specific electrical and geometrical combinations. The main utility of the antenna array is to obtain a set of positions for the antenna elements such its radiation pattern matches the desired one. The consideration of the design of antenna array as a specific problem has attracted the interests of the researchers particularly in the field of electromagnetic optimization. Earlier the approach was primarily based on applying numerical techniques but their inadequacies have forced the researchers to implement nature inspired metaheuristics [4]. Notable works have been reported in this regard, most importantly that of Panduro *et.al.* [5], [9-11] in which Differential Evolution [6], Genetic Algorithm [7] and Particle Swarm Optimization [8], have been applied to obtain the optimal design of scanned antenna arrays. Gurel and Ergul [10] applied GA for designing circular array antenna where every element is a log periodic antenna.

This paper has been organized into 6 sections. Section 2 describes the design problem in detail .Section 3 gives a detailed analysis of the Artificial Bee Colony algorithm. In Section 4 our proposed sNABC algorithm have been illustrated in detail. Section 5 specifies the experimental design section of the three possible cases and

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 459–466, 2012.

discusses the results obtained. Lastly the paper is concluded in Section 6 with future research scope being specified.

2 Design Problem

Circular antenna array problems consist of N antenna elements distributed on a circle of radius **r.** Arrangement of the circular elements is given below

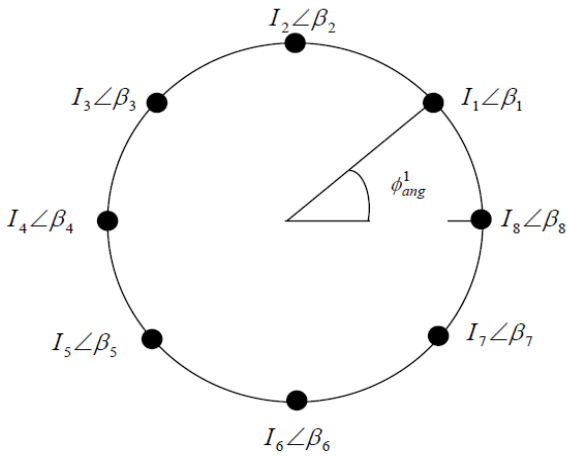


Fig. 1. Geometrical arrangement of circular antenna array

The Array Factor for the circular antenna array is calculated as :-

$$AF(\phi)=\sum_{n=1}^{N} I_n \exp\left[jkr\left(\cos\left(\phi-\phi^n_{ang}\right)-\cos\left(\phi_0-\phi^n_{ang}\right)\right)+\beta_n\right] \quad (1)$$

The array factor includes the following terms:-

(i) $\boldsymbol{I_n}$ is the current excitation associated with the nth element.
(ii) β_n is the phase excitation associated with the nth element.
(iii) $\emptyset$ is the angle of the incident plane wave.

In the array factor $\emptyset^n_{ang}$ is the angular position of the n[th] element of the antenna array in x-y plane. It is calculated as:-

$$\emptyset^n_{avg} = 2\pi\left(\frac{n-1}{N}\right) \quad (2)$$

Here in case of circular antenna array, if *k* is the wavenumber and *r* is the radius of the circle defined by circular array then the relation ***kr=Nd*** holds. $\emptyset_0$ is the direction of maximum radiationThe current and the phase excitations of the antenna elements are varied so as to suppress the sidelobes,minimize the beamwidths and obtain null control in the directions as desired. Symmetrical excitation of the circular antenna array is considered in which the following relations hold

$$
\begin{aligned}
I_{n/2+1}\angle\beta_{n/2+1} &= conj\left(I_1\angle\beta_1\right),\\
I_{n/2+2}\angle\beta_{n/2+2} &= conj\left(I_1\angle\beta_1\right),.....\\
I_n\angle\beta_n &= conj\left(I_{n/2}\angle\beta_{n/2}\right)
\end{aligned}
\tag{3}
$$

The objective function considered for the design problem is given as

$$
OF = \left|AR\left(\varphi_{sll},\vec{I},\vec{\beta},\varphi_0\right)\right| \Big/ \left|AR\left(\varphi_{\max},\vec{I},\vec{\beta},\varphi_0\right)\right| + 1\Big/DIR\left(\varphi_0,\vec{I},\vec{\beta}\right) + \left|\varphi_0-\varphi_{des}\right| + \sum_{k=1}^{mm}\left|AR\left(\varphi_k,\vec{I},\vec{\beta},\varphi_0\right)\right| \tag{4}
$$

The objective function can be minimised by separately considering the four components of the objective function.The first component attempts to minimize the level of the sidelobes, where $\emptyset_{sll}$ is the angle at which the maximum level of the side lobe is attained. The second component's task is to maximize the directivity of the array configuration. Directivity, which measures the directive gain of the antenna configuration has emerged as a key figure of merit for comparing various antenna patterns. The third component of the objective function attempts to drive the array configuration towards the desired maxima i.e. $\emptyset_{des}$. The fourth component deals with the nulls and penalizes the objective function if the null control of desired proportion is not achieved. ***nl*** is the number of null control directions and $\emptyset_k$ is the the kth null control direction.

3 Artificial Bee Colony Algorithm

The Artificial Bee Colony Algorithm intoduced by Karaboga [9], simulates the foraging behavior of the bees in solving intricate computational problems. The ABC algorithm allows the division of labour scheme, in which the entire bee colony is subdivided into three groups:-***employed bees,onlooker bees, scout bees.*** The basic steps of the ABC algorithm is elucidated below as:-

1 Foragers which are unemployed search the entire search space for potential food. Once food source is found it is promoted to an Employer Bee and carries local exploitation of the source, analyzing food sites based on their nectar content.
2 The employed bee chooses a fit source to load nectar, memorizes its position and returns to hive where, the employer unloads their nectar and in the process communicate the presence of a fit source to waiting onlookers via waggle dance.
3 Onlooker probabilistically selects one of the sources advertized to it,which is exploited by onlooker and again the steps performed by Employer bees are repeated.A food source is abandoned by a forager when it has exhausted its nectar content and it becomes a scout bee performing random walks in search space.

4 The *sNABC* Algorithm

The *sNABC* algorithm is a novel variant of the original Artificial Bee Colony Algorithm in which a novel mutation based on selective neighborhood has been introduced, which makes it superior in comparison with the original ABC algorithm. The main steps of the algorithm are detailed as below:-

(i) Initialization of Food Sources: The algorithm starts by initializing the food sources randomly in the search space .The food sources are initialized in the following manner:-

$$Food^{j}{}_{i} = Food^{j}{}_{min} + r.\,(Food^{j}{}_{max} - Food^{j}{}_{min}) \tag{5}$$

Food $^{j}{}_{max}$ and Food $^{j}{}_{min}$ are the upper and the lower limits of the j^{th} dimension of the i^{th} Food source, respectively. Here, **j Є [1, D]** for a D-dimensional problem, i belongs to **[1, FN]** for FN number of Food Sources and ***r*** is a random number uniformly generated in the range [0, 1] and later designated as ***rand***.

(ii) Employed Phase: Foragers are mainly responsible for carrying out the exploitation of the food sources. They are responsible for constant searching of the food sources. In order that the search process of the employed phase is balanced between exploration and exploitation. A perturbation scheme is applied taking into account randomly selected *k* parameters and perturbing them with a food source selected from a list of neighbouring members sorted in order of increasing Euclidean distance. The value of *k* is randomly varied but truncated to the range [1, *FN*/2].A random number is generated between 0 and 1 and if it is less than $\boldsymbol{Prb}_{\text{nbd}}$ (neighborhood selection probability) which is set to 0.5, then the perturbation is done with respect to k nearest neighbors of the current individual .Otherwise the farthest k number of neighbors of the current individuals are selected.

$$\vec{V}_i = \begin{cases} \overrightarrow{Food_i} + \emptyset_i \times (\overrightarrow{Food_i} - \overrightarrow{Food_k}^{N} & if\ rand \le Prb_{nbd} \\ \overrightarrow{Food_i} + \emptyset_i \times (\overrightarrow{Food_i} - \overrightarrow{Food_k}^{F} & Otherwise \end{cases} \tag{6}$$

The selection factor S determines the frequency of perturbation and ideally ranges from 0.3 to 0.6 for competitive results. We have chosen S to be 0.6 since it provides a good tradeoff between exploration and exploitation. Positional perturbation of the forager's position is enacted if a randomly generated number *rand*, in the range (0, 1), is less than the selection factor. The positional modification is as follows:-

$$\vec{U}_i{}^{j} = \begin{cases} \vec{V}_i{}^{j} & if\ rand \le S \\ \overrightarrow{Food_i}{}^{j} & Otherwise \end{cases} \tag{7}$$

This is followed by a greedy (fitness-based) selection by which the forager replaces it's currently employed Food Source by a fitter one recently discovered.The fitness is calculated using the following formula :-

$$fit(\overrightarrow{Food}) = \begin{cases} \dfrac{1}{1 + f(\overrightarrow{Food})} & f(\overrightarrow{Food}) \le 0 \\ 1 + \left| f(\overrightarrow{Food}) \right| & f(\overrightarrow{Food}) > 0 \end{cases} \tag{8}$$

In the above mentioned formula, *fit(Food)* is the fitness value and *f(Food)* is the objective functional value of the food source respectively.

(iii) Onlooker Phase: In the Onlooker Phase, an waiting onlooker bee selects a food source by means of probabilistic selection method and determines an adjacent food source for exploitation to get a better solution. The onlooker bees select food sources if the randomly generated number in the range [0, 1] is less than $\boldsymbol{prob}_i$.

$$prob_i = 0.9 \times \frac{fit_i}{\max(fit_i)} + 0.1 \quad (9)$$

(iv) Scout Phase: After rejecting a food source, the concerned employee bee becomes a scout bee for searching a better food source in the the unexplored region. A trial counter is set which counts the number of cycles for which the food source does not improve upon its previous fitness value. As the trial counter reaches a predefined limit this food source is reinitialized.

5 Simulation Results

Three instances of the problem are considered here. The cases involve design of optimal array pattern possessing no null control, having null at 50^o and nulls at both 50^0 and 120^0. The results of our algorithm *sNABC* has been compared with the following algorithms using the parametric settings as inferred from literature.

(i) Differential evolution(DE). [6]
(ii) Particle swarm optimization(PSO). [8]
(iii) Artificial bee colony algorithm (ABC). [9-10]

The simulation of the benchmark suite have been performed on an Intel dual-core machine with 2 GB RAM 2.36 GHz speed using MATLAB 7.5. The results for each algorithm have been tabulated based on the data obtained from 25 trial runs. The mean and standard deviation for the objective function have been reported.

5.1 Parametric Settings

For the algorithm ***sNABC*** the parameters used are identical to that of classical ABC:

(i) Colony Size (CS)=100
(ii) limit=100
(iii) K (number of nearest individuals)=10

5.2 Results

Case I: 12 element array having no null control

Table 1. Results for the median of 25 trials(*Case I*)

Algorithm	SLL(dB)	Directivity(dB)
sNABC	**-20.16**	**11.53**
ABC	-18.47	11.27
PSO	-18.64	11.31
DE	-18.61	11.29

Table 2. Mean function values and standard deviations over 25 test runs (*Case I*)

Algorithm	Mean obj. func value	Standard deviation
sNABC	**0.1878**	**0.0032**
ABC	0.2154	0.0048
PSO	0.2241	0.0432
DE	0.2176	0.0311

The optimal array pattern thus obtained is shown in figure 2. The results clearly indicate that *sNABC* finds the best possible array pattern with minimized side lobes and greater directivity.

Case II: 12 element array having null at 50^0

Table 3. Results for the median of 25 trials (Case II)

Algorithm	SLL(dB)	Directivity(dB)	Array factor (dB) at 50^0
sNABC	**-18.87**	**11.34**	**-70.44**
ABC	-14.57	10.65	-68.45
PSO	-17.83	10.87	-69.43
DE	-18.13	11.08	-47.44

Table 4. Mean function values and standard deviations over 25 test runs (*Case II*)

Algorithm	Mean obj.func value	Standard deviation
sNABC	**0.2194**	**0.0093**
ABC	0.2791	0.0194
PSO	0.2243	0.0487
DE	0.2178	0.0318

From the obtained results it is visible that sNABC clearly outperforms other competing algorithms by suppressing the array pattern to -70.44 dB, at the required null direction of 50^0.The array pattern thus obtained is shown in figure 3.

Case III*:* 12 element array having null at 50^0 and 120^0

Table 5. Results for the median of 25 test runs (Case III)

Algorithm	SLL(dB)	Directivity(dB)	Array factor (dB) at 50^0	Array factor (dB) at 120^o
sNABC	**-18.76**	**11.21**	**-68.13**	**-70.35**
ABC	-13.73	10.68	-41.42	-52.17
PSO	-11.42	10.34	-42.92	-54.15
DE	-18.05	10.96	-52.53	-48.47

Table 6. Mean function values and standard deviations over 25 test runs (Case III)

Algorithm	Mean	Standard deviation
sNABC	**0.2017**	**0.0089**
ABC	0.2558	0.0115
PSO	0.3865	0.0287
DE	0.2211	0.0345

The results given above clearly indicate that our algorithm has provided the best possible minimized sidelobe levels in both 50^o and 120^0 directions. The corresponding array pattern is thus shown in figure 4.

5.3 Computational Effort and Complexity

To do away with the platform hardware specifications, we state the computational time as the ration of the time taken by sNABC to that of basic ABC to optimize the

problem. The average value was found out to be 1.021 which is suitable for real world applications. Similar ratios of 1.011 and 1.042 were obtained with respect to PSO and DE. Due to computation of the distance matrix the worst case complexity is $O(CS^2)$. But due to nature of the benchmark problem the difference in time is not pronounced. Usually the average case complexity is $O(CS\ log\ CS)$ owing to symmetric nature of distance matrix which saves repeatative calculation.

6 Conclusion

In the current scenario, the design of such an optimal antenna array pattern, where side lobes have to be minimized apart from maximizing the directivity is quite challenging task. In this paper we have focused on the application of a novel variant of the Artificial Bee Colony Algorithm, where the employed phase is modified by incorporating a trade-off scheme between exploitation and exploration, which has proved to be superior in comparison with the other contender algorithms. The optimization task considered here is in the form of a cost function that considers the average levels of the side lobes, null control, and circumference of the array. On all the three cases considered above, our algorithm has significantly outperformed the competing state of art algorithms regarding all the possible criteria considered (SLL, directivity etc).Future research works will stress on exploring the possibility of application of our algorithm in case of other array geometries. Further research work is possible by considering the components of the cost function as a multi objective problem but suitable problem specific knowledge is required in order to point out the best solution from the Pareto-optimal set.

References

[1] Godara, L.C.: Handbook of Antennas in Wireless Communications, Boca Raton, FL (2002)
[2] Balanis, C.A.: Antenna Theory-Analysis and Design. Wiley, India (2005)
[3] Chandran, S.: Adaptive Antenna Arrays:Trends and Applications. Springer (2004)
[4] Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, New York (1999)
[5] Panduro, M.A., Brizuela, C.A., Balderas, L.I., Acosta, D.A.: A comparison of genetic algorithms, particle swarm optimization and the differential evolution method for the design of scannable circular antenna arrays. Progress In Electromagnetics Research B 1, 171–186 (2009)
[6] Storn, R., Price, K.: Differential Evolution- A Simple Efficient heuristic Strategy for Global Optimization over Continuous Spaces. Journal of Global Optimization 11, 341–359 (1997)
[7] Goldberg, D.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading (1989)
[8] Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proc. 6th Int. Symp. Micromach. Human Sci., pp. 39–43 (1995)
[9] Karaboga, D., Basturk, B.: A powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm. Journal of Global Optimization 39(3) (2007)
[10] Karaboga, D., Basturk, B.: On the performance of artificial bee colony (ABC) algorithm. Applied Soft Computing 8, 687–697 (2008)

[11] Mandal, A., Zafar, H., Das, S., Vasilakos, A.V.: Efficient Circular Array Synthesis with a mementic Differential Evolution Algorithm. Progress in Electromagnetics Research, 367–385 (2012)
[12] Gurel, L., Ergul, O.: Design and simulation of circular arrays of trapezoidal-tooth log-periodic antennas via genetic optimization. Progress In Electromagnetics Research (PIER) 85, 243–260 (2008)

Supplementary Attachment: Optimal Array Pattern for the cases discussed above

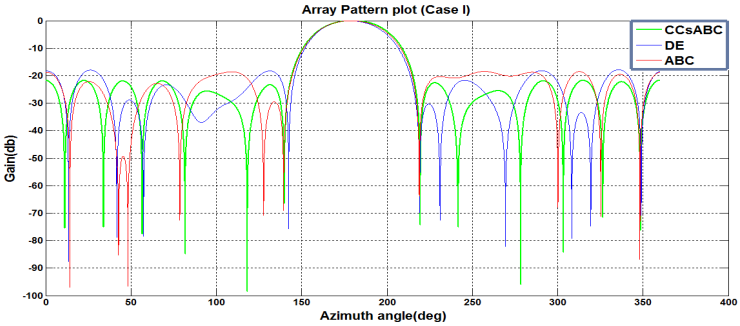


Fig. 2. Antenna array pattern for 12 element array having no null control

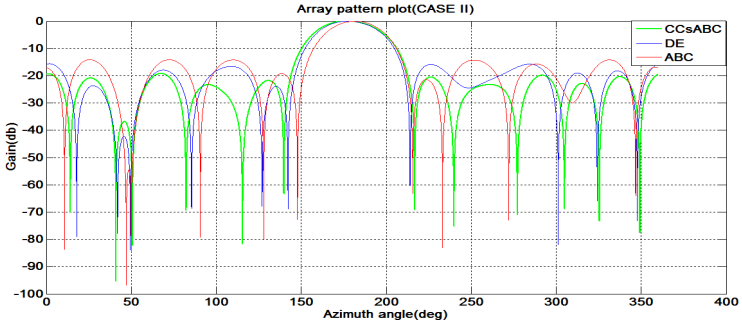


Fig. 3. Antenna array pattern for 12 element array having null at 50^0

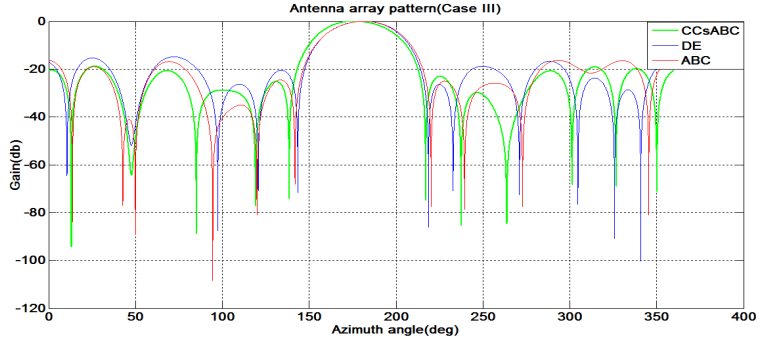


Fig. 4. Antenna array pattern for 12 element array having null at 50^0 and 120^0

Cooperative Co-evolutionary Teaching-Learning Based Algorithm with a Modified Exploration Strategy for Large Scale Global Optimization

Subhodip Biswas, Souvik Kundu, Digbalay Bose, and Swagatam Das

Dept. of Electronics and Communication Engineering,
Jadavpur University, Kolkata 700 032, India
{subho.opto.placis91,sk210892,digbose92}@gmail.com

Abstract. Evolutionary Algorithms, inspired from the Darwinian theory on evolution of species, are heuristic method for solving difficult unimodal and multimodal functions. But the ultimate disadvantage of those Evolutionary Algorithms is premature convergence, i.e. trapping in a local optimum due to poor exploration strategy. In case of High Dimensional problems, there are huge chances of convergence prematurely due to the large search space, which grows exponentially with the increase of dimension of the problem. In this paper a modified Teaching-Learning-Based technique is used to investigate the effectiveness of different cooperative co-evolutionary framework for solving high dimensional problems.

1 Introduction

The curse of dimensionality in Evolutionary Algorithms (EAs) [1-3] is a recurrent phenomenon and to find a way to do away with this major setback, Potter and De Jong [4] proposed Cooperative Co-evolution (CC) for tackling Large Scale Optimization problems based on the idea of partitioning the parameters upon which the problem depends into smaller subcomponents, of lesser number of parameters. It is treated as a sub-problem of lower dimensionality and thereby will not affect the performance of the optimizer. CC seems to be a promising approach for large scale optimization but its performance degrades in case of non-separable problems due to parameter linkage. Generally all interacting variables should be grouped into number of subcomponents such that they capture inter-dependencies among them with a fair degree of success. But due to lack of prior information about the interaction among the variables, we have no clue about the proper grouping. This introduces the need of sophisticated approach based on trial-and-error to latch the linkage and minimize its effect on the optimizer.

The rest of the paper is structured as follows: Section 2 elucidates the CC framework and the optimizers with mathematical justifications of including the proposed techniques as improvement in the CC frameworks. Section 3 contains the comparative study and experimental analysis. The concluding remarks are given in Section 4 with a brief scope for future works that may be undertaken.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 467–475, 2012.

2 Preliminaries

This section concludes brief review of cooperative co-evolution, CC framework, and the modified teaching learning based optimization algorithm on the basis of CC framework.

2.1 Cooperative Co-evolution: Motivation and Earlier Work

Potter [5] introduced the CC concept in Genetic Algorithm to model CCGA, which became the reference work for similar strategy adaptation in Evolutionary Strategy (CCES) [6], Particle Swarm Optimization (CPSO) [7] and Differential Evolution (CCDE) [8]. Initially Potter described the CC framework by decomposing the D dimensional vector into D subcomponents and optimizes the whole vector by improving the single dimension at a time. During the development work in CPSO, it was introduced that the D dimensional problem vector in decomposed into m s-dimensional vectors. This idea improved the result than the earlier CC concept but could not give satisfied outcomes in case of High dimensional problems since it failed to include the interaction between the variables in case of non separable problems. CCDE also suffered from the same thing, i.e. curse of dimensionality.

Yang *et al.* [9-10] introduced a systematic approach of finding the interaction through random grouping and co-adaptation. Then MLCC [11] was introduced and this framework improved the performance of the Differential Evolution very much. It ensured promising results also in thousand dimensional problems too. Suganthan *et al.* [12] used a multi-trajectory search technique in Self-Adaptive DE to optimize the high dimension problems with promising results.

2.2 Random Grouping and Adaptive Weighting

One of the major problems to deal with the high dimensional problem is to establish the connection between the interacting variables for non-separable problems. Yang introduced the concept of random grouping to overcome this drawback. He proved that by random grouping the decision variables at the beginning of each cycle, enhances the probability of grouping two interacting variables in a same subcomponent.

The probability of grouping two interacting particles for at least k cycles can be mathematically represented as

$$p_k = \sum_{r=k}^{N} \binom{N}{r} \left(\frac{1}{m}\right)^r \left(1 - \frac{1}{m}\right)^{N-r} \tag{1}$$

Where N is the total number of cycles and m is the number of subcomponents.

Yang experimented that the probability of two interacting particles to remain in same subcomponent for at least two cycles is 96.6% for a 1000D problem decomposed into 10 subcomponents.

Along with the random grouping an *Adaptive Weighting* scheme is introduced to enhance the co-adaptation of the subcomponents. In adaptive weighting a numeric

weight value is assigned to each of the subcomponent, which forms a *weight vector*. It can be easily stated that optimize the weight vector is far easy than optimize the given problem vector because of the dimensional difference. Weight vector is m-dimensional, which is same as the number of subcomponents. But the problem vector is m*s dimensional which is quite large. Yang outlined the weight vector co-adaptation [10] by the following way –

- Start a new cycle
- Applying the random grouping strategy the D dimensional problem vector is decomposed into m subcomponents each consists of s dimensions. It provides the variable equal chance to be assigned in any of the subcomponent.
- Optimize each individual subcomponent by the corresponding optimizer using a predefined number of function evaluations.
- Construct a weight vector and evolve it by an EA for the best, worst and a random member of the specified population.
- Go to step 1 for beginning of a new cycle or stop if the maximum no of function evaluation has been achieved.

The previously mentioned modified TLBO algorithm is the base algorithm here.

2.3 Multilevel Cooperative Co-evolution

The major disadvantages of the Random Grouping and Adaptive weighting is that there was no provision of dynamically changing the subcomponent size reviewing the past performance in a specified population size. There were two major disadvantages as pointed out by Yang. Firstly the assumption that weighting vector enhances the performance of the algorithm; it is very computationally expensive since 2/3 of the maximum fitness evaluation is expended. Secondly the two particle interaction concept is not a generalized version because it does not conclude the interaction of *l* independent particle interaction for at least *k* cycles in a particular subcomponent.

Thereafter Yang *et al.* [11] introduced a newer and modified version of the random grouping excluding the adaptive weighting scheme. It is generalized, i.e. he introduced a new formula which calculates the probability of residing *l* independent particles in a particular subcomponent for at least *k* cycles. It can be represented as –

$$p(X \geq k) = \sum_{r=k}^{N} \binom{N}{r} \left(\frac{1}{m^{v-1}}\right)^{r} \left(1 - \frac{1}{m^{v-1}}\right)^{N-r} \quad (2)$$

N is the total number of cycles. X denotes the number of times *l* individual variables successfully grouped in a subcomponent for at least *k* cycles. If a total 50 cycles are considered, the probability of 10 interacting variables grouped in a particular subcomponent for at least one cycle is 4.88%. From the probability plot it can be concluded that the above mentioned probability is increased by 60% if 5 independent variables are considered for a maximum number of 1e+4 cycles. So it is feasible to use the more frequent random grouping rather than using the general random grouping and adapting weighting scheme, which need a lot of FEs. In cost of that much FEs we can extrapolate the maximum number of cycles to optimize the subcomponents better.

A decomposer pool of varying group sizes is used in Multi-level framework and represented as S= {s_1, s_2, …, s_{t-1}, s_t} and the performance obtained of the optimizer is stored in a record list, i.e. R={r_1, r_2,…, r_{t-1}, r_t}. A selection probability set is constructed based on the relative rank of its past performance, is represented as **p** = {p_1, p_2, …, p_{t-1}, p_t}. The members of the **p** set is generated by the formula as–

$$p_i = \frac{e^{m \times r_i}}{\sum_{j=1}^{t} e^{m \times r_j}} \quad i = \{1,2,\ldots,t\} \tag{3}$$

Where r_i Є R and is updated according to the formulae –

$$r_i = \frac{(f^* - f)}{|f^*|} \tag{4}$$

Where f^* and f are the best fitness of the previous cycle and present cycle respectively.

The steps involved in the MLCC framework can be summarized as the following –

- Determine the decomposer pool **S** which decides the number of parameters for each variable in a subcomponent.
- Construct the record list **R** by setting each individual r_i equal to 1, which varies cyclically.
- Generate the initial population consisting of trial vectors.
- Memorize the best fit value *f* of current population.
- Check: if $r<\varepsilon$ then go to step 2.
- Calculate the selection probability set **p.**
- A particular member s_i of the selection set **S** is selected based on the value of $\mathbf{p_i}$.
- Now m subcomponents are created using random grouping technique such that s_i*m=D, where D is the dimension of the problem.
- Optimize the subcomponent using the specified optimizer.
- Update r_i .
- Check: If termination criteria are achieved, else go to step 6.

In this paper, a modified TLBO (Teaching-Learning-Based Optimization) approach is used to optimize the subcomponents, which is discussed in the section 2.5.

2.4 TLBO Algorithm

TLBO, recently proposed by Rao *et al.* [14], algorithm is structured on the teaching learning procedure in an institution or class. Students, i.e. the learners enhance their knowledge by the influence of both the teacher and the other students in the class. This optimization technique is based on these two facts that a learner learns from the teacher, i.e. the teacher phase and from the interaction among the other learners, i.e. the learning phase. TLBO is a population-based approach where the group of learners is treated as the population, number of subjects as parameters and their results are treated as the fitness function. The individual corresponding to the best fitness value is assigned as the Teacher. The algorithm can be divided into two parts Teacher Phase and Learner Phase. In Teacher phase, the teacher actuates to increase the mean of the

learners to his or her level. But it is quite impossible for any learner in a class to learn and reach the level of teacher. So this teaching phase just improve the existing mean of the learners to a better value. This is done via a difference mean method given as –

$$Mean^{DIF}{}_i = \text{rnd}_i \,.\, (M_T - \mathbf{T_F} \,.\, M_L) \tag{5}$$

M_T is the mean of the Teacher and M_L is the mean of the learner. T_F is the teaching factor which takes randomly 1 or 2 with equal probability. At the next generation the learner position can be updated as shown –

$$X_i^{C+1} = X_i^{C} + Mean^{DIF}{}_i \tag{6}$$

In the next phase, a learner improves his/her performance by interacting among each other. This can be mathematically represented as –

$$X_i^{C+1} = X_i^{C} + \text{rnd}_i \,.\, (X_i^{C} - X_j^{C}) \quad \text{if } f(X_i) < f(X_j) \tag{7}$$

$$X_i^{C+1} = X_i^{C} + \text{rnd}_i \,.\, (X_j^{C} - X_i^{C}) \quad \text{if } f(X_j) < f(X_i) \tag{8}$$

where X_i and X_j are two different learners at C^{th} generation. X_i^{C+1} is accepted for next iteration if it provides better functional value.

2.5 Modified TLBO Algorithm (*m*-TLBO)

Premature convergence is a major problem for many optimizers. To preclude this premature convergence we have introduced some explorative strategy in the modified version of the TLBO. In the Learner phase of the original TLBO algorithm explorative strategies have been introduced.

I. ***Teacher Phase:*** In the Teacher phase the mean difference is calculated as mentioned above and thereafter the learner position is updated accordingly as represented by the equation **(5)** and **(6)**.

II. ***Learner Phase:*** In the learner phase one learner meliorates his/her position via interaction. The learner with better fitness attracts other learner by means of a difference vector as shown in equation **(7)** and **(8)**. To maintain the diversity in the population two new modifications have been introduced. One is the application of Cauchy random number and the other is the introduction of difference vector which differentiates between the best and worst learner. Cauchy distribution sustains the diversity due to its long tail. Two parameters help to obtain the Cauchy number, mean and sigma (standard deviation) is set to 0.5 each. Now the interaction between the learners can be represented mathematically as –

$$X_i^{C+1} = X_i^{C} + k_{1i} \,.\, (X_i^{C} - X_j^{C}) + k_{2i} \,.\, (X_{best}^{C} - X_{worst}^{C}) \quad \text{if } f(X_i) < f(X_j) \tag{9}$$

$$X_i^{C+1} = X_i^{C} + k_{1i} \,.\, (X_j^{C} - X_i^{C}) + k_{2i} \,.\, (X_{best}^{C} - X_{worst}^{C}) \quad \text{if } f(X_j) < f(X_i) \tag{10}$$

k_1 and k_2 are two random number from Cauchy distribution. X_{best} is the fittest learner while X_{worst} is the worst fit learner. The difference vector is introduced to enhance diversification. The pseudo code of the proposed algorithm is elucidated below–

- Begin a new cycle.
- Randomly initialize the population (learners).

REPEAT

- Determine the mean of the Teacher and the learner and calculate the Mean Difference according to the equation **(5)**.
- Update the position of a learner according to the equation **(6)**.
- Generate the Cauchy random numbers. Then again update the position of a learner by using equation **(9)** or **(10)**. If the finally updated learner provides better fitness value it will be updated as the population individual.

STOP IF THE TERMINATION CRITERION HAS BEEN OBTAINED

3 Comparative Study of CC Framework in m-TLBO Algorithm

3.1 Benchmark Used and Parametric Set Up

The above mentioned algorithm is tested on benchmark functions from CEC 2008 Special Session and Competition on Large Scale Global Optimization. There is 2 unimodal and 5 multimodal functions. Interested readers are asked to consult the Technical Report [13] to get a better understanding of the test problems.

Table 1. CEC 2008 Benchmark Set

Function	Function Name	Nature	Linkage
F_1	Shifted Sphere Function	Unimodal	Separable
F_2	Shifted Schwefel's Problem 2.21	Unimodal	Non-separable
F_3	Shifted Rosenbrock's Function	Multimodal	Non-separable
F_4	Shifted Rastrigin's Function	Multimodal	Separable
F_5	Shifted Griewank's Function	Multimodal	Non-separable
F_6	Shifted Ackley's Function	Multimodal	Separable
F_7	Fast Fractal "DoubleDip" Function	Multimodal	Non-separable

Mean and Standard deviation of error values based on 25 test runs have been tabulated. All the 7 functions mentioned are minimization problem, so the best performance given by the TLBO variants is judged from the one with the least mean. Test runs for D =100, 500 and 1000. The maximum number of Fes allotted is D*5000. The population of the class is kept constant at 50.

3.2 The Different CC Frameworks Applied

The different CC frameworks applied to **m-TLBO** algorithm are given as follows

Table 2. Brief overview of different CC frameworks investigated

Algorithm	Nature of Cooperative Co-evolution
*n*CC-*m*-TLBO	Splitting *n*-dimensional problem by original CC proposed by Potter
*n/2*CC-m-TLBO	Splitting *n*-dimensional vector into 2 *n/2*-dimensional vectors
CC-m-TLBO-G	CC framework with random grouping and adaptive weighting proposed by Yang
CC-m-TLBO	Similar to CC-m-TLBO-G but with adaptive weighting disabled and the sub-component optimizer made to run once
MLCC-m-TLBO	Similar to CC-m-TLBO but selects decomposers from a pool of various sizes based on historical records of the decomposer size
CC-m-TLBO-ML	CC-m-TLBO with provision for selecting subcomponent size based on probabilistic selection.

3.3 Experimental Results and Analysis

Table 3. CEC 2008 Benchmark Set Results for 100 Dimensions

Function / Algo.	F_1	F_2	F_3	F_4	F_5	F_6	F_7
*n*CC-*m*-TLBO	3.13E-17 (5.76E-12)	1.23E+02 (3.16E+01)	2.87E+04 (7.88E+03)	1.97E+02 (1.28E+02)	4.45E-02 (1.07E-04)	1.37E+01 (1.35E+01)	-4.13E+02 (9.96E+01)
*n/2*CC-m-TLBO	5.15E -18 (5.28E-18)	2. 94E+01 (1.66E+01)	1.54E+03 (4.63E+02)	9.17E+01 (2.55E+00)	1.59E-02 (3.37E-05)	1.01E+01 (2.35E-01)	-5.62E+02 (2.43E+02)
CC-m-TLBO-G	4.25E-20 (7.66E-21)	3.46E+00 (5.87E-01)	5.12E+02 (7.81E+01)	4.19E+01 (2.77E+00)	2.61E-02 (1.19E-02)	7.86E-02 (4.53E-04)	-9.86E+02 (2.45E+02)
CC-m-TLBO	**3.68E-24 (6.63E-30)**	2.58E+00 (7.23E-01)	2.11E+02 (1.12E+01)	6.81E+00 (2.14E+00)	6.97E-01 (4.23E-03)	1.17E+00 (6.81E-01)	-1.35E+03 (8.14E+01)
MLCC-m-TLBO	7.14E-19 (9.82E-25)	9.28E-01 (6.58E-02)	2.09E+02 (9.12E+00)	3.45E-09 (5.78E-15)	**4.15E-14 (7.49E-14)**	**4.86E-07 (7.31E-09)**	-1.47E+03 (3.21E+02)
CC-m-TLBO-ML	4.91E-22 (2.11E-27)	**1.28E-05 (2.53E-06)**	**1.51E+02 (3.72E+01)**	**7.53E-39 (2.61E-46)**	5.71E-04 (8.26E-06)	9.65E-07 (8.40E-05)	**-1.62E+03 (7.85E+01)**

Table 4. CEC 2008 Benchmark Set Results for 500 Dimensions

Function / Algo.	F_1	F_2	F_3	F_4	F_5	F_6	F_7
*n*CC-*m*-TLBO	7.09E-15 (9.84E-04)	1.86E+02 (4.95E+01)	5.92E+04 (1.23E+04)	3.14E+02 (1.34E+02)	1.41E-01 (2.16E-03)	7.78E+01 (4.85E+01)	-7.23E+02 (1.57E+02)
*n/2*CC-m-TLBO	3.24E -17 (5.28E-08)	2. 94E+01 (1.66E+01)	7.54E+03 (2.76E+03)	1.07E+02 (1.56E+01)	4.32E-02 (7.04E-03)	2.52E+00 (1.16E+00)	-9.34E+02 (3.57E+02)
CC-m-TLBO-G	8.17E-22 (9.48E-29)	7.75E+00 (4.61E-01)	8.92E+02 (1.11E+02)	2.27E+00 (3.67E-01)	8.21E-04 (5.61E-05)	2.21E-01 (1.30E-02)	-2.12E+03 (3.47E+02)
CC-m-TLBO	7.43E-26 (1.69E-29)	3.04E+00 (4.71E-01)	8.54E+02 (5.12E+01)	2.06E+00 (4.46E-02)	5.74E-03 (2.46E-04)	**1.23E-06 (1.41E-05)**	**-4.13E+03 (2.84E+02)**
MLCC-m-TLBO	4.27E-20 (6.35E-24)	2.67E+00 (8.15E-01)	7.79E+02 (8.65E+01)	4.16E-07 (2.41E-09)	**5.24E-11 (1.63E-12)**	2.82E-03 (4.21E-04)	-2.47E+03 (3.21E+02)
CC-m-TLBO-ML	**4.91E-26 (3.21E-28)**	**3.46E-02 (5.38E-04)**	**4.21E+02 (2.31E+01)**	**9.11E-32 (7.71E-38)**	5.62E-11 (2.57E-10)	2.17E-05 (4.35E-06)	-3.88E+03 (9.10E+01)

Table 5. CEC 2008 Benchmark Set Results for 1000 Dimensions

Function / Algo.	F_1	F_2	F_3	F_4	F_5	F_6	F_7
*n*CC-*m*-TLBO	1.44E-04 (6.84E-03)	8.07E+02 (6.05E+01)	2.55E+05 (5.14E+04)	6.76E+02 (2.25E+02)	8.62E-01 (5.26E-02)	1.01E+02 (7.24E+01)	2.65E+03 (1.26E+02)
n/2CC-m-TLBO	4.68E -05 (3.16E-08)	5. 46E+01 (3.74E+00)	8.54E+03 (2.48E+03)	3.54E+02 (9.13E+01)	7.37E-02 (6.28E-03)	8.61E+00 (2.73E+00)	-5.78E+03 (3.57E+02)
CC-m-TLBO-G	8.70E-21 (3.07E-23)	4.17E+00 (2.55E-01)	1.87E+03 (4.53E+02)	8.72E+00 (1.04E+00)	1.83E-03 (7.68E-05)	7.23E-01 (2.23E-02)	-9.60E+03 (2.57E+02)
CC-m-TLBO	**7.43E-24 (1.69E-27)**	6.57E+00 (3.21E-01)	9.64E+02 (6.18E+01)	3.48E+01 (5.73E-01)	2.72E-03 (1.87E-04)	**2.52E-04 (3.55E-05)**	**-1.63E+04 (3.22E+03)**
MLCC-m-TLBO	5.63E-18 (2.27E-23)	3.74E+01 (1.21E+00)	8.58E+02 (4.26E+01)	6.12E-05 (2.13E-07)	**2.49E -08 (5.26E-09)**	5.57E-03 (5.26E-04)	-1.11E+04 (4.04E+03)
CC-m-TLBO-ML	2.31E-21 (1.37E-24)	**2.64E-01 (2.37E-02)**	**6.75E+02 (1.97E+01)**	**5.59E-16 (6.27E-18)**	3.07E-06 (5.20E-08)	2.50E-04 (4.46E-06)	-1.66E+04 (7.16E+02)

Discussion. From the results tabulated in Table 3, 4 and 5 we get an idea of the effectiveness of different CC framework based on the proposed optimizer **m-TLBO**. The use of Cauchy mutation has its own advantage since it prevents over-exploitation by the teacher whose purpose is to attract the class mean towards the best achieved value. Using Cauchy-sampled random numbers and usage of difference vector induces diversity among the learners. It not prevents them from converge to the best learner but also promotes a thorough exploration of parameter space. It is for these combined reasons that the outcome on benchmark functions, both separable and non-separable, has been promising.

4 Conclusion

In this paper, the authors have extended the application of Cooperative Co-evolution framework to a recently proposed optimizer called Teaching and Learning-based algorithm and discussed the relative performance of the used CC variants. From analytical point of view CC-m-TLBO-ML was found out to be the best hybridized TLBO variant that can be used for future works in the field of large scale optimization. Future research may be undertaken to improve the CC variants or development of an adaptive pool that incorporates the various CC variants, used in this paper, based on relative performance.

References

[1] Back, T.: Evolutionary Algorithms in Theory and Practice: Evolutionary Strategies, Evolutionary Programming, Genetic Algorithms. Dover Books on Mathematics. Oxford University Press (1996)

[2] Eberhart, R.C., Kennedy, J.: Particle Swarm Optimization. In: Proceedings of IEEE Int. Conference on Neural Network, pp. 1942–1948 (November-December 1995)

[3] Storn, R., Price, K.: Differential Evolution-A Simple Efficient heuristic Strategy for Global Optimization over Continuous Spaces. Journal of Global Optimization 11, 341–359 (1997)

[4] Potter, M.: The Design and Analysis of a Computational Model of Cooperative Co-evolution Ph. D Thesis, George Mason University (1997)
[5] Potter, M.A., De Jong, K.A.: A Cooperative Coevolutionary Approach to Function Optimization. In: Davidor, Y., Männer, R., Schwefel, H.-P. (eds.) PPSN 1994. LNCS, vol. 866, pp. 249–257. Springer, Heidelberg (1994)
[6] Sofge, D., De Jong, K.A., Schultz, A.: A blended population approach to cooperative co-evolution for decomposition of complex problems. In: Proceedings on Congress on Evolutionary Computation, pp. 413–418 (2004)
[7] Bergh, F., Engelbrecht, A.P.: A Co-operative Approach to Particle Swarm Optimization. IEEE Trans. on Evo. Comp. 3, 225–239 (2004)
[8] Shi, Y., Teng, H., Li, Z.: Cooperative Co-evolutionary Differential Evolution for Function Optimization. In: Proceedings of the First International Conference on Natural Computation, pp. 1080–1088 (2005)
[9] Yang, Z., Tang, K., Yao, X.: Large scale evolutionary optimization using cooperative co-evolution. Information Sciences 178, 2985–2999 (2008)
[10] Yang, Z., Tang, K., Yao, X.: Differential Evolution for High-Dimensional Function Optimization. In: IEEE Congress on Evolutionary Computation, pp. 3523–3530 (2007)
[11] Yang, Z., Tang, K., Yao, X.: Multilevel Cooperative Co-evolution for Large Scale Optimization. In: IEEE Congress on Evolutionary Computation, pp. 1663–1670 (2008)
[12] Zhao, S.Z., Suganthan, P.N., Das, S.: Self-adaptive Differential Evolution with Multi-trajectory Search for Large Scale Optimization. Soft Computing (November 2011), doi:10.1007/s00500-010-0645-4
[13] Tang, K., Yao, X., Suganthan, P.N., MacNish, C., Chen, Y.P., Chen, C.M., Yang, Z.: Benchmark Functions for the CEC '2008 Special Session and Competition on Large Scale Global Optimization. Technical Report, Nature Inspired Computation and Applications Laboatory, USTC, China (2007), http://nical.ustc.edu.cn/cec08ss.php
[14] Rao, R.V., Savsani, V.J., Vakharia, D.P.: Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. Computer-Aided Design 43, 303–315 (2011)

Brain Storming Incorporated Teaching–Learning–Based Algorithm with Application to Electric Power Dispatch

K.R. Ramanand [1], K.R. Krishnanand [2], Bijaya Ketan Panigrahi[3], and Manas Kumar Mallick[4]

[1] Electrical and Electronics Engineering Dept.,
MEA Engineering College, Malappuram, India
ramanandkr@gmail.com
[2] Multi–Disciplinary Research Cell, Siksha 'O' Anusandhan University,
Bhubaneswar, India
krishkr09@gmail.com
[3] Department of Electrical Engineering,
Indian Institute of Technology, Delhi, India
bkpanigrahi@ee.iitd.ac.in
[4] Department of Computer Science and Engineering,
ITER, Siksha 'O' Anusandhan University, Bhubaneswar, India
mk_mallick2004@yahoo.com

Abstract. This paper intends to incorporate a brain storming mechanism into the existing Teaching–Learning–Based Optimization (TLBO) algorithm. The potential solutions of TLBO evolve using the primitive steps that are maintained between the acts of teaching and learning. Another novel algorithm, Brain Storm Optimization (BSO) sticks to the philosophy of interchange of ideas by a team to develop as a whole. The brain storming methods from BSO are introduced into the working of TLBO and applied to a well–studied electric power dispatch problem of high intricacy. The results are compared to best of the existing solutions to demonstrate the efficacy of the proposed hybrid algorithm.

Keywords: Brain storm optimization, Dynamic economic dispatch, Osborn's Rules for Idea Generation, Teaching–learning–based optimization.

1 Introduction

Computations and optimization algorithms derived from commonly occurring mechanisms and naturally found phenomena are not uncommon interests in engineering research [1]. There are many examples of algorithms such as Genetic Algorithm [2], Particle Swarm Optimization [3], Differential Evolution [4], Artificial Bee Colony [5], Seeker Optimization algorithm [6] etc. being used for solving power engineering problems. The economically optimal and technically viable dispatch of electric power to the load points is one of such issues in electrical engineering. The objective in such problems is to find an optimal schedule of thermal generation, or factors that affect

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 476–483, 2012.

generation, such that the fuel cost spent for its operation is minimal [2]. There could be multiple schedules that are viable for a facility, but always subjected to many constraints on each of the generators.

The high nonlinearity of the problem and the uneven distribution of the search space lead conventional gradient based techniques to end up with suboptimal solutions [1]. The intricacy and dimensionality of the problem increases when the scheduling is to be done for many consecutive intervals. Most optimal solutions so far have been found using evolutionary computational techniques [7–14].

TLBO is one of the latest techniques presented in research literature towards the purpose of optimizing nonlinear functions [15–16]. BSO is also a recent technique meant for the same purpose [17]. The learners in TLBO transform to teachers through a process of teacher–learner interaction. BSO is purely based on idea generation and their interchange; controlled by a few guidelines by Osborn [18]. The original TLBO algorithm inherently lacks any idea exchange between learners since they evolve only through interactions with the teacher and through self–improvement (represented by addition of random magnitude of a differential vector). From the philosophical frame of TLBO, it seems only natural to integrate it with the concept of brain storming presented in the BSO algorithm – introducing social behavior in learners through sharing of ideas. This provides a collective growth to the learners without diluting the intrinsic qualities of interactions in TLBO. This paper presents such a new teaching–learning algorithm with brain storming for optimization (TLBSO). The proposed TLBSO algorithm is applied on a 10 unit thermal generation test system which requires 24 hours of scheduling based on forecasted load demands.

The following part of paper is organized such – section 2 details the dynamic economic dispatch problem. It includes the mathematical model of the problem along with the practical constraints taken, found from prior research literature. Section 3 explains the procedure followed by the TLBSO algorithm in executing the three algorithmic phases in it. Section 4 gives the application of TLBSO to the problem formulated and the results thus obtained from experimentation. The conclusion is presented in section 5.

2 Objective of Dynamic Power Dispatch

The economic dispatch of electric power to the load points through collective use of different thermal generators is a tough scheduling problem from the aspect of nonlinearity and dimensionality. Considering an array of U generators for a time interval t, the power productions can be represented as an array $P(t)$.

$$P_{(t)} = \lfloor p_{(t,1)} \quad p_{(t,2)} \quad \cdots \quad p_{(t,U-1)} \quad p_{(t,U)} \rfloor \tag{1}$$

The demand at that time can be given as $d(t)$. The power generation at t should follow the demand constraint given below.

$$\sum P_{(t)} = \sum_{u=1}^{U} p_{(t,u)} = d_{(t)} \tag{2}$$

Also, each generator has a maximum and minimum quantity of power that has to be generated. At any interval t, for the generator u,

$$p_{(u)}^{\min} \le p_{(t,u)} \le p_{(u)}^{\max} \tag{3}$$

The cost of generation, considering valve–point loading, for an interval t is given as

$$Cost_{(t)} = \sum_{u=1}^{U} \left(A_{(u)} p_{(t,u)}^2 + B_{(u)} p_{(t,u)} + C_{(u)} + \left| E_{(u)} \sin\left(F_{(u)} \times \left(p_{(t,u)}^{\min} - p_{(t,u)} \right) \right) \right| \right) \tag{4}$$

where A, B, C, D, E and F are cost coefficients which are unique for a particular thermal unit. The total cost is just the sum of all costs over the number of time intervals T and it is the objective to be minimized.

$$Total\ Cost = \sum_{t=1}^{T} Cost_{(t)} \tag{5}$$

Considering the fact that the generation of a unit for the next interval can only increase or decrease within certain limits from its previous generation, power ramp constraints are imposed.

$$p_{(u)}^{Up\ Ramp} \ge \left(p_{(t+1,u)} - p_{(t,u)} \right) \qquad if \left(p_{(t+1,u)} > p_{(t,u)} \right) \tag{6}$$

$$p_{(u)}^{Down\ Ramp} \ge \left(p_{(t,u)} - p_{(t+1,u)} \right) \qquad if \left(p_{(t+1,u)} < p_{(t,u)} \right) \tag{7}$$

The ramp limits are unique to the generator considered and they cause coupling of consecutive generation schedules. So, it is not practical here to solve the power generation problem for each time interval separately. Hence, the dimension of the problem to be optimized is $T \times U$.

3 Teaching–Learning with Brain–Storming for Optimization (TLBSO) Algorithm

The Teaching–Learning–Based Optimization algorithm and Brain Storm Optimization are very recent algorithms introduced in [15] and [17] respectively. In TLBO, the teachers aim at raising the mean performance of the learners and share their individual knowledge with the learners. The individual learners also try to personally excel through self–effort. Meanwhile, BSO technique relies on the process of human idea generation. The following guidelines are used keeping in view Osborn's Original Rules for Idea Generation in a Brain storming Process [18].

1. Suspend Judgment
2. Anything Goes
3. Cross–fertilize (Piggyback)
4. Go for Quantity

The rule 3 suggests that lot of ideas can and should be based on ideas already generated. Any generated idea can and should serve as a clue to render more ideas. It implies that cross–fertilization is the core action in the process of brain storming. The key idea of brain storming in a teacher–learner–based algorithm is to cross–fertilize the ideas from teacher phase and learner phase to obtain a new set of ideas, which if proved superior, can replace the inferior ideas. This simple amalgamation of mutually accommodating philosophies gives rise to the TLBSO algorithm detailed below.

3.1 Initialization

Initially, a matrix X of R rows and C columns is initialized using randomly generated values within the upper and lower bounds of the variables. R represents population size and C represents the dimensionality of the considered problem. The procedure is configured to run for a total of G iterations.

$$x_{(r,c)}^{(1)} = x_c^{lower} + rand_{(r,c)} \times \left(x_c^{upper} - x_c^{lower} \right) \tag{8}$$

where r and c are row index and column index respectively. *rand(r,c)* represents a random variable uniformly distributed within the range (0,1). Since ideas with prior known validity can speed up the process of generating new valid ideas, the matrix X can be seeded with prior known valid solutions, if any. The r^{th} potential solution vector (or the learner) for the g^{th} generation is given by

$$X_{(r)}^{(g)} = \left[x_{(r,1)}^{(g)}, x_{(r,2)}^{(g)}, \ldots, x_{(r,c)}^{(g)}, \ldots, x_{(r,C)}^{(g)} \right] \tag{9}$$

The evaluation of an individual in the X matrix is represented as

$$Y_{(r)}^{(g)} = f\left(X_{(r)}^{(g)} \right) \tag{10}$$

The Y values collectively form a column vector of objective values at any given generation. For all the mathematical equations used in the algorithm, $r=1,2,\ldots,R$, $c=1,2,\ldots,C$ and $g=1,2,\ldots,G$. All the rand values mentioned in the algorithm are set to follow uniform distribution.

3.2 Teacher Phase

An averaging operation is performed to find the mean values of each dimension separately. The average vector V is computed as

$$V_{average}^{(g)} = \left[mean\begin{pmatrix} x_{(1,1)}^{(g)} \\ \vdots \\ x_{(r,1)}^{(g)} \\ \vdots \\ x_{(R,1)}^{(g)} \end{pmatrix} \quad \ldots \quad mean\begin{pmatrix} x_{(1,c)}^{(g)} \\ \vdots \\ x_{(r,c)}^{(g)} \\ \vdots \\ x_{(R,c)}^{(g)} \end{pmatrix} \quad \ldots \quad mean\begin{pmatrix} x_{(1,C)}^{(g)} \\ \vdots \\ x_{(r,C)}^{(g)} \\ \vdots \\ x_{(R,C)}^{(g)} \end{pmatrix} \right] \tag{11}$$

The best row vector which gives the least value of objective function is regarded as the teacher ($X_{Teacher}^{(g)}$) for the current iteration. The mutation in the r^{th} vector influenced by the teacher vector is given below.

$$Xnew_{(r)}^{(g)} = X_{(r)}^{(g)} + rand^{(g)} \times \left(X_{Teacher}^{(g)} - T_F V_{average}^{(g)} \right) \tag{12}$$

where the teaching factor T_F is taken randomly for each iteration as either 1 or 2.

3.3 Learner Phase

The learner phase consists of self–improvement of the learners through differential mutation. This differential process is based on the ephemeral gradients that are present between the learners. For a particular learner $X_{(r)}^{(g)}$, another learner $X_{(\delta)}^{(g)}$ is randomly selected ($r \neq \delta$). The c^{th} dimension of the r^{th} individual of the matrix *Xnew* of this stage is given as

$$Xnew_{(r,c)}^{(g)} = \begin{cases} X_{(r,c)}^{(g)} + rand_{(r,c)}^{(g)} \times \left(X_{(r,c)}^{(g)} - X_{(\delta,c)}^{(g)} \right) & if\left(Y_r^{(g)} < Y_\delta^{(g)} \right) \\ X_{(r,c)}^{(g)} + rand_{(r,c)}^{(g)} \times \left(X_{(\delta,c)}^{(g)} - X_{(r,c)}^{(g)} \right) & otherwise \end{cases} \tag{13}$$

3.4 Brain–Storming Phase

This phase implements the cross–fertilization of ideas obtained from the phases prior to it. A temporary population is created from a weighted sum of *Xnew* from the teacher phase and the learner phase.

$$\overbrace{Xnew_{(r,c)}^{(g)}}^{Temporary} = \alpha_{(c)} \times \overbrace{Xnew_{(r,c)}^{(g)}}^{Teacher\ Phase} + \beta_{(c)} \times \overbrace{Xnew_{(r,c)}^{(g)}}^{Learner\ Phase} \tag{14}$$

where α and β are the weights used to control the contribution of the population from teacher phase and learner phase to the brain storming process. The weights are subject to the constraints below to ensure sufficient participation of learners in brain storming from both the phases.

$$\left(\frac{X_{(c)}^{lower}}{2X_{(c)}^{upper}} \right) \leq \alpha_{(c)} \leq \left(\frac{X_{(c)}^{upper}}{2X_{(c)}^{lower}} \right) \text{ and } \left(\frac{X_{(c)}^{lower}}{2X_{(c)}^{upper}} \right) \leq \beta_{(c)} \leq \left(\frac{X_{(c)}^{upper}}{2X_{(c)}^{lower}} \right) \tag{15}$$

Further, the temporary population is mutated using a smooth nonlinear function multiplied by a stochastic variable.

$$\overbrace{Xnew_{(r,c)}^{(g)}}^{Cross-Fertilized} = \overbrace{Xnew_{(r,c)}^{(g)}}^{Temporary} + \log sig\left(\frac{G-2g}{2K} \right) \times rand_{(r,c)}^{(g)} \tag{16}$$

where *K* is the slope determining factor of the *logsig* function used. The *X* matrix and the final *Xnew* matrices in each phase are now sorted together on the basis of fitness and the next iteration permits only the *R* best learners which are selected.

4 Simulation Results

The proposed TLBSO algorithm is tested on a 10 unit test system, scheduling them for 24 hours. The dimension of the problem is hence 240, which is quite large. This effectively becomes 216 since the last generator in this test system must operate at a constant power. The system data is taken from [4]. The algorithm is run many times and the best schedule is selected as optimal. The algorithm is often seeded with near to optimal values known beforehand. This fastens the convergence process. The best generation schedule obtained is given in Table 1.

Table 1. Generation schedule of each unit for each hour (in MW)

Hour	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}
1	226.608	135.000	167.354	60.000	73.000	122.450	129.588	47.000	20.000	55.000
2	226.625	137.854	188.332	60.088	122.996	122.514	129.590	47.000	20.001	55.000
3	226.625	217.854	206.500	110.087	122.878	122.452	129.590	47.000	20.014	55.000
4	303.249	222.267	286.059	60.087	122.827	159.922	129.589	47.000	20.000	55.000
5	303.249	222.267	297.568	110.002	172.744	122.580	129.590	47.000	20.000	55.000
6	379.873	225.239	318.112	120.453	172.734	159.999	129.590	47.000	20.000	55.000
7	379.873	305.239	299.402	120.411	222.591	122.836	129.648	47.000	20.000	55.000
8	456.497	309.536	293.258	120.243	222.429	122.450	129.587	47.000	20.000	55.000
9	456.497	389.536	302.820	170.243	222.600	122.537	129.589	55.178	20.000	55.000
10	456.498	396.800	335.316	180.971	222.685	159.991	129.590	85.178	49.971	55.000
11	456.497	396.799	340.000	230.970	241.967	160.000	129.590	115.177	20.000	55.000
12	456.497	460.000	340.000	241.519	237.394	160.000	129.591	119.999	20.000	55.000
13	456.498	396.800	300.210	241.219	222.750	159.934	129.590	89.999	20.000	55.000
14	456.497	396.797	294.365	191.219	172.773	122.437	129.607	85.303	20.002	55.000
15	379.873	396.801	313.263	180.818	122.894	122.458	129.590	55.303	20.000	55.000
16	303.248	316.814	305.172	130.820	122.870	123.477	129.599	47.000	20.000	55.000
17	226.624	396.793	289.648	119.908	73.000	122.438	129.589	47.000	20.000	55.000
18	303.249	396.800	310.415	120.519	122.939	122.479	129.591	47.008	20.000	55.000
19	379.873	396.800	297.460	155.005	172.754	122.510	129.590	47.000	20.008	55.000
20	456.497	459.996	339.985	181.228	222.592	159.964	129.674	47.063	20.001	55.000
21	456.497	396.794	297.227	176.857	222.583	122.450	129.591	47.001	20.000	55.000
22	379.873	316.794	257.442	127.083	172.767	122.452	129.589	47.000	20.000	55.000
23	303.249	309.543	185.240	77.254	122.930	82.193	129.591	47.000	20.000	55.000
24	226.625	309.533	185.327	60.014	73.039	77.870	129.592	47.000	20.000	55.000
Total cost = 1,019,073.50 $/24 hours										

Table 2. Comparison of total costs obtained using different techniques

Method	Total cost (in $/24 hrs)	Method	Total cost (in $/24 hrs)
EP-SQP [7]	1,031,746	IPSO [12]	1,023,807
MDE [8]	1,031,612	AIS [13]	10,21,980
MHEP-SQP [9]	1,028,924	DE [4]	1,019,786
DGPSO [10]	1,028,835	HHS [14]	1,019,091
PSO-SQP [11]	1,027,334	**TLBSO (proposed)**	**1,019,074**

The comparison on the obtained total cost is done in Table 2 with some of the best reported in recent research literature. The result obtained from the proposed TLBSO algorithm is very much comparable to others.

5 Conclusion

A unification of teaching-learning process and idea cross-fertilization through brain storming is performed in the paper. The concept of evolving individual through a teacher and through brain storming suits each other. The algorithm thus developed performs optimization through three phases of teaching, learning and brain storming. The proposed algorithm is tested on a complex dynamic economic dispatch problem. The results obtained are comparable to the other reported results and hence the algorithm shows potential for being used as an optimization tool in various areas.

References

1. Back, T., Fogel, D., Michalewicz, Z.: Handbook of evolutionary computation. Oxford University Press, New York (1997)
2. Bakirtzis, A., Petridis, V., Kazarlis, S.: Genetic algorithm solution to the economic dispatch problem. IEE Gen. Trans. Dist. 141(4), 377–382 (1994)
3. Gaing, Z.L.: Particle swarm optimization to solving the economic dispatch considering the generator constraints. IEEE Trans. on Power Syst. 18(3), 1187–1195 (2003)
4. Balamurugan, R., Subramanian, S.: Differential Evolution-based Dynamic Economic Dispatch of Generating Units with Valve-point Effects. Electric Power Components and Systems 36, 828–843 (2008)
5. Nayak, S.K., Krishnanand, K.R., Panigrahi, B.K., Rout, P.K.: Application of Artificial Bee Colony to Economic Load Dispatch Problem with Ramp Rate Limits and Prohibited Operating Zones. In: IEEE Proc. on Nature and Biologically Inspired Computing, pp. 1237–1242 (2009)
6. Krishnanand, K.R., Rout, P.K., Panigrahi, B.K., Mohapatra, A.: Solution to Non-convex Electric Power Dispatch Problem Using Seeker Optimization Algorithm. In: Panigrahi, B.K., Das, S., Suganthan, P.N., Dash, S.S. (eds.) SEMCCO 2010. LNCS, vol. 6466, pp. 537–544. Springer, Heidelberg (2010)
7. Attaviriyanupap, P., Kita, H., Tanaka, E., Hasegawa, J.: A hybrid EP and SQP for dynamic economic dispatch with nonsmooth fuel cost function. IEEE Trans. Power Syst. 17(2), 411–416 (2002)
8. Yuan, X., Wang, L., Yuan, Y., Zhang, Y., Cao, B., Yang, B.: A modified differential evolution approach for dynamic economic dispatch with valve-point effects. Energy Conversion and Management 49(12), 3447–3453 (2008)
9. Victoire, T.A.A., Jeyakumar, A.E.: A modified hybrid EP–SQP approach for dynamic dispatch with valve-point effect. Elect. Power and Energy Syst. 27(8), 594–601 (2005)
10. Victoire, T.A.A., Jeyakumar, A.E.: Deterministically guided PSO for dynamic dispatch considering valve-point effect. Elect. Power Syst. Res. 73(3), 313–322 (2005)
11. Victoire, T.A.A., Jeyakumar, A.E.: Reserve constrained dynamic dispatch of units with valve-point effects. IEEE Trans. on Power Syst. 20(3), 1273–1282 (2005)

12. Yuan, X., Su, A., Yuan, Y., Nie, H., Wang, L.: An improved PSO for dynamic load dispatch of generators with valve-point effects. Energy 34(1), 67–74 (2009)
13. Hemamalini, S., Simon, S.P.: Dynamic economic dispatch using artificial immune system for units with valve-point effect. Elect. Power and Energy Syst. 33(4), 868–874 (2011)
14. Pandi, V.R., Panigrahi, B.K.: Dynamic economic load dispatch using hybrid swarm intelligence based harmony search algorithm. Expert Syst. with Appl. 38(7), 8509–8514 (2011)
15. Rao, R.V., Savsani, V.J., Vakharia, D.P.: Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. Computer-Aided Design 43(3), 303–315 (2011)
16. Krishnanand, K.R., Panigrahi, B.K., Rout, P.K., Mohapatra, A.: Application of Multi-Objective Teaching-Learning-Based Algorithm to an Economic Load Dispatch Problem with Incommensurable Objectives. In: Panigrahi, B.K., Suganthan, P.N., Das, S., Satapathy, S.C. (eds.) SEMCCO 2011, Part I. LNCS, vol. 7076, pp. 697–705. Springer, Heidelberg (2011)
17. Shi, Y.: An Optimization Algorithm Based on Brainstorming Process. Int. J. of Swarm Intel. Res. 2(4), 35–62 (2011)
18. Osborn, A.F.: Applied imagination: Principles and procedures of creative problem solving. Charles Scribner's Son, New York (1963)

Associated and Assorted Recombination in SBX Operator for Problems with Linkages

Arnav Acharyya[1] and Kalyanmoy Deb[2]

[1] Department of Electrical Engineering, Indian Institute of Technology Bhubaneswar, India
arnav@iitbbs.ac.in
[2] Department of Mechanical Engineering, Indian Institute of Technology Kanpur, India
deb@iitk.ac.in

Abstract. This paper explores the efficacy of two different recombination schemes of variable-wise SBX operator. Once two offspring variable values are created by SBX operator on two parent values, their concatenation to form two overall offspring variable vector is an important issue, which is not studied much. Here, we propose two methodologies: associated and assorted recombinations, and demonstrate the working on a number of non-linked, partially-linked and fully-linked problems. Based on the results, we suggest a linkage-based recombination scheme that employs either associated or assorted scheme depending on the level of linkage associated with a variable.

1 Introduction

Variable-wise blending has become a common recombination operator in real-parameter genetic algorithms (RGAs). In this paper, we consider the simulated binary crossover (SBX) operator [1], which takes one variable at a time with a probability, considers values from two parent vectors, and recombines them to find two new values. Variables for which SBX is not performed, the parent variable values are retained. Once a pair of values for each variable are created (or retained), their recombination to form a new offspring vector is an important matter, which has not been paid much attention.

Here, we consider mainly two different recombination schemes and investigate their efficacy in solving problems having different linkages. Both elitist and non-elitist RGAs are considered. Based on results on a number of test problems with varying linkages, a number of interesting observations are made. A linkage-based recombination scheme is then devised and is found to perform consistently well on all test problems used in this study.

The study emphasizes the importance of identifying linkage information present in a real-parameter optimization problem (as in other GA studies [2,3]) and concludes that a suitable recombination scheme of variable-wise values can then be developed using the linkage information. Such a recombination scheme is expected to perform better than any other recombination scheme.

2 Simulated Binary Crossover (SBX) Operator

The SBX operator, developed in 1995, has become almost a default recombination operator for real-parameter genetic algorithms (RGAs). This operator was motivated from

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 484–491, 2012.

the *interval* schema processing point of view [1]. For a n-variable vector, SBX is applied variable-wise with a probability of 0.5 of performing a recombination of each variable. Ignoring the details, the procedure of computing two offspring variable values (c_1 and c_2) from two given parent variable values ($x_i^{(1)}$ and $x_i^{(2)}$) is as follows:

$$c_1 = 0.5(1+\bar{\beta})x_i^{(1)} + 0.5(1-\bar{\beta})x_i^{(2)}, \tag{1}$$

$$c_2 = 0.5(1-\bar{\beta})x_i^{(1)} + 0.5(1+\bar{\beta})x_i^{(2)}, \tag{2}$$

where $\bar{\beta}$ is computed for a random number $u_i \in [0,1]$, as follows:

$$\bar{\beta} = \begin{cases} (\alpha u_i)^{1/(\eta_c+1)}, & \text{if } u_i \le \frac{1}{\alpha}; \\ \left(\frac{1}{2(1-\alpha u_i)}\right)^{1/(\eta_c+1)}, & \text{otherwise.} \end{cases} \tag{3}$$

The parameter $\alpha = 2-\beta^{-(\eta_c+1)}$ and $\beta = 1+\frac{2}{|x_i^{(1)}-x_i^{(2)}|}\min\left[\min\left(x_i^{(1)},x_i^{(2)}\right)-x_i^{(L)}, x_i^{(U)}-\max\left(x_i^{(1)},x_i^{(2)}\right)\right]$. The parameter η_c is the distribution index and a value of 2 was reported to perform well for single-objective optimization [1]. For unbounded variables, $x_i^{(L)} = -\infty$ and $x_i^{(U)} = \infty$ can be chosen. If a variable is not recombined, $c_1 = x_i^{(1)}$ and $c_2 = x_i^{(2)}$ are assumed.

After two offspring variable values are computed using Equations 1 and 2 for each of n variables, there are several ways they can be concatenated to form two offspring vectors:

Associated Recombination: The offspring c_1 from all variables are used to form the first offspring variable vector. Similarly, all c_2 values are concatenated to form the second offspring vector. This recombination is expected to maintain linkage between variables, since each resulting offspring is likely to be close to one of the parent solution vectors.

Assorted Recombination: The offspring vectors are chosen from a random assortment of created offspring variable values. This recombination is likely to maintain diversity in the population.

Line Recombination: We also consider a *line* recombination (which was earlier referred as Line-SBX [4]), in which an associated SBX operation is performed on all variables and using an identical random number $u_i = u \in [0,1]$. Solutions along a line joining the two parent vectors can only be created by this operator.

3 Linkage in Problems and Its Identification

In a multi-variable function ($f(\mathbf{x})$), a variable x_i is said to be *non-linked* with another variable x_j, if the function can be rewritten as addition of independent sub-functions ($g_i(\mathbf{x}\backslash x_j) + g_j(\mathbf{x}\backslash x_i) + g_0(\mathbf{x}\backslash(x_i,x_j))$). Conversely, x_i and x_j are said to be *linked*, if the sub-functions containing x_i cannot be additively separated from the sub-functions containing x_j. On the basis of linkage between variables following functions can be defined:

Separable Function: No two variables are linked in such a function. The function may have the following structure: $f(\mathbf{x}) = \sum_{i=1}^{n} g_i(x_i)$.

Fully-Linked Function: All variables are mutually linked with each other.

Partially-Linked Function: Some variables are linked and others are non-linked between each other.

In this paper, we shall investigate the effect of each of the three SBX recombination schemes in solving problems having different linkages. It is expected that for a separable and low-linked function (having no or small linkage), the assorted recombination should perform well as a random assortment of the variables is likely to increase the probability of getting the optimal combination of variables quicker. However, for problems having a high linkage, random assortment of variables will be too noisy and associated or line recombination should perform better. We also use non-elitist and elitist RGAs to investigate the effect of each SBX recombination scheme on elite preservation. For elitist strategy, we compare both parent and offspring solutions and the better two are kept.

4 Results

Associated, assorted and line recombinations of SBX operations (marked as 1, 2, and 3 in figures) are performed with elitist and non-elitist RGAs. A population of size $10n$, crossover probability of 0.9, mutation probability of $1/n$, SBX and polynomial mutation indices of 2 and 100, respectively, are used. 1,000 generations are run for termination. For each function, 20 runs are performed.

4.1 Separable Problems

The following separable function, constructed with tri-modal sub-functions, is chosen:

$$f(x_1, x_2, \ldots, x_{10}) = \sum_{i=1}^{10} \left[\frac{(x_i - 44)^2 (x_i^2 - 44x_i + 700)(x_i^2 - 20x_i + 150)}{(44^2 \times 700 \times 150)} \right]. \qquad (4)$$

The optimal solution is $x_i^* = 44$ with $f^* = 0$. Figure 1 summarizes the obtained results. Results with associated recombination is shown in the left, assorted recombination in the middle, and line recombination in the right. It is clear that although elitist RGAs perform better in terms best-of-the-runs performance, non-elitist RGA is better with median performance. Variation of population-best objective value for elitist and non-elitist RGAs are shown in Figures 2 and 3, respectively. Considering median performance, non-elitist RGAs with assorted recombination performs the best, as expected.

4.2 Fully and Partially-Linked Problems

We choose the same function as in the previous subsection, except that derived variables $y_i = \sum_{j=1}^{10} a_{ij} x_j$ are used to compute the function value. The parameters a_{ij} comes from $\mathbf{A}_{n \times n} = \begin{bmatrix} R & \mathbf{0} \\ \mathbf{0} & I \end{bmatrix}$, where R is a $r \times r$ matrix with r_{ij} is chosen randomly within $[-1, 1]$.

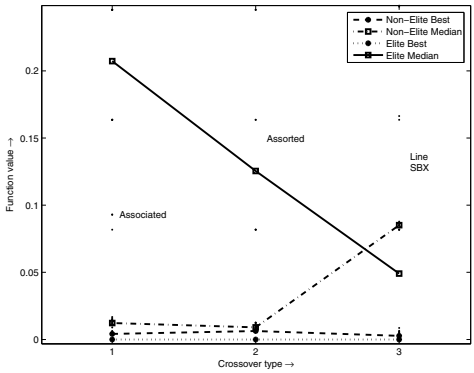


Fig. 1. Best and median performance on separable function

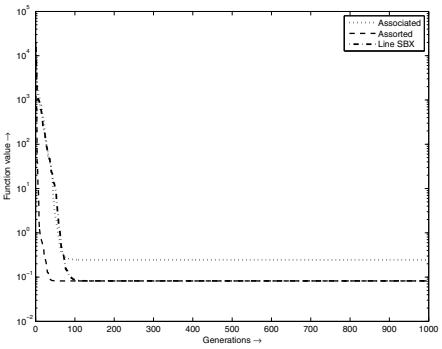


Fig. 2. Plot of convergence in Elitist simulation for non-linked function (eq: 4) minimization

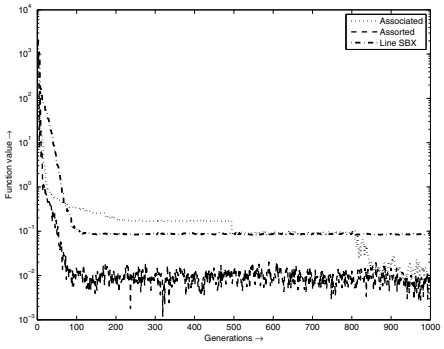


Fig. 3. Plot of convergence in Non-Elitist simulation for non-linked function (eq: 4) minimization

Thus, if $r = 1$, the above function is identical to that in the previous subsection and the generated function is a separable function. But, if $r = n$, all variables are linked and the generated function is fully-linked. A partially-linked function can be formed easily by choosing $1 < r < n$. Note that the optimal objective value is always $f^* = 0$.

The performance on fully-linked function is shown in Figure 4. Here, as well, the elitist RGA performs better. Moreover, associated and line recombinations perform better than the assorted recombination, as expected.

Next, we consider different partially-linked functions by varying r from one to nine. Results with non-elitist RGAs are shown in Figure 5. It is clear that the performance of all RGAs deteriorates as the linkage (r) increases. Figure 6 shows the difference of other recombinations from the associated recombination. Negative values indicate better performance. We observe several aspects from these figure. First, for non-elitist

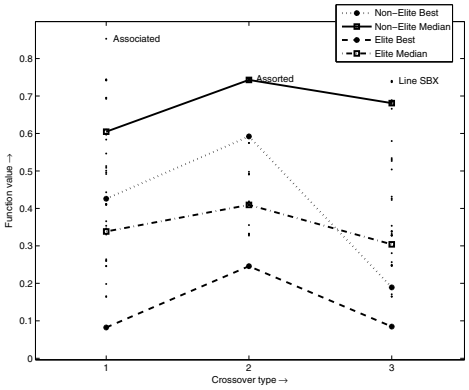


Fig. 4. Best and median performance on fully-linked function

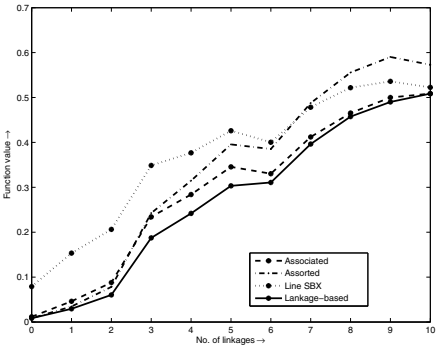


Fig. 5. Median performance on partially-linked functions with non-elitist RGAs

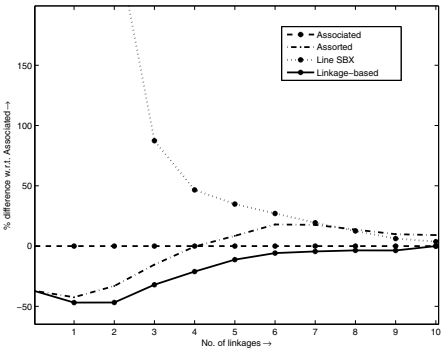


Fig. 6. Percentage difference with respect to associated SBX with non-elitist RGAs

RGAs, the line recombination does not perform well. This is because a line recombination assumes a full linkage in a problem and does not create useful offspring for partially-linked problems. Second, the assorted recombination performs better than the associated recombination up to $r = 4$; thereafter associated recombination performs better. Since an increased linkage requires a better coordination among variables, the associated recombination performs well for high r functions.

Similar simulations are performed with the elitist RGAs and results are presented in Figures 7 and 8. All recombinations (including line recombination) perform better than the associated recombination up to $r = 6$. The added diversity introduced by the assorted SBX operation is balanced better by the increased convergence due to elite preservation, thereby improving the performance of assorted recombination with elitist RGAs. The significant enhancement in performance in line recombination with elite preservation is interesting.

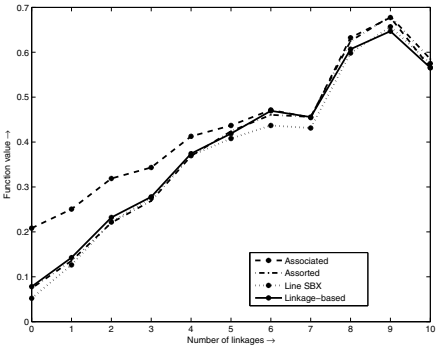


Fig. 7. Median performance on partially-linked functions with elitist RGAs

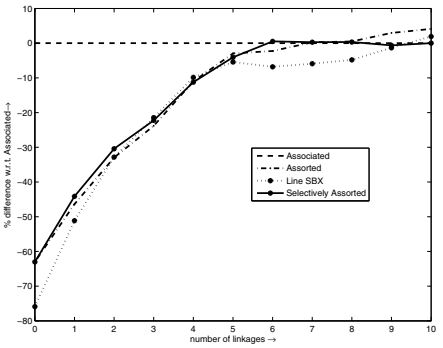


Fig. 8. Percentage difference with respect to associated SBX with elitist RGAs

4.3 Linkage-Based Recombination

From the above extensive results, we observe that assorted SBX recombination works better for separable problems and associated SBX recombination works for fully-linked problems. However, the performance of line recombination is dependent on elite preservation. Therefore, for an arbitrary problem, if the extent of linkage can be identified, assorted or associated recombination can be used accordingly.

Mathematically, linkage can explored by analyzing the partial derivatives of the function with respect to each variable. If the partial derivative of f with respect to a particular variable is independent of another variable, then the two variables can be identified to be non-linked. For a multi-variable function, we identify the linkage between any two variables x_i and x_j using difference in objective values, as follows. For a random solution vector $\mathbf{x}$, we keep other variables fixed and perturb i-th and j-th variables to $\bar{x}_i$ and $\bar{x}_j$. We note the differences $\varepsilon_1 = f(x_i,x_j) - f(\bar{x}_i,x_j)$ and $\varepsilon_2 = f(x_i,\bar{x}_j) - f(\bar{x}_i,\bar{x}_j)$. If $|\varepsilon_1 - \varepsilon_2|/|\varepsilon_1| \leq \delta$ (a small number, 10^{-7} is used here) for 25 different initial random vectors, we consider variables x_i and x_j are non-linked. We perform the above test on all pairs of variables and establish a linkage pattern in a problem.

In the linkage-based SBX recombination, we identify the linkage between variables using the above procedure before applying RGA. The variables that are found to be linked together are combined by using the associated recombination scheme and variables that are not linked are combined using the assorted recombination scheme. Figures 5 to 8 show the performance of this linkage-based scheme on elitist and non-elitist RGAs. In all cases, the linkage-based recombination scheme performs consistently better than other schemes.

4.4 Schwefel and Rosenbrock Functions

Next, we consider two standard test problems which have a full linkage among their variables:

$$f_{\text{sch}}(\mathbf{x}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2, \quad f_{\text{ros}}(\mathbf{x}) = \sum_{i=1}^{n-1} 100 \left(x_{i+1} - x_i^2 \right)^2 + \sum_{i=1}^{n} (1 - x_i)^2 .$$

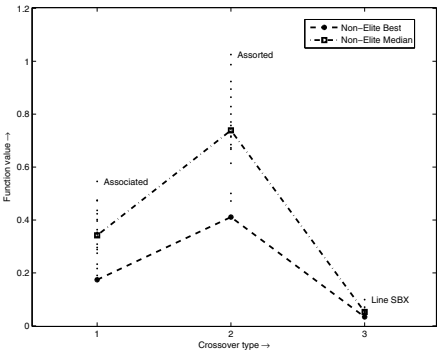


Fig. 9. Best and median performance on Schwefel function with non-elitist RGAs

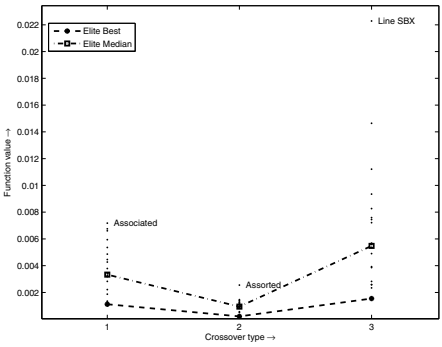


Fig. 10. Best and median performance on Schwefel function with elitist RGAs

The optimal objective values for both problems are zero and solutions are $x_i^* = 0$ and 1, respectively. Since both the above problems are fully-linked, the linkage-based recombination becomes exactly the same as the associated recombination scheme and hence is not applied. Figures 9 and 10 show performance of three recombination schemes with non-elitist and elitist RGAs, respectively. It is interesting to note that elitist RGAs perform much better than non-elitist RGAs. For non-elitist RGAs, assorted recombination does not perform better while for elitist RGAs it performs better. As explained before, diversity maintenance of assorted scheme gets complemented with the selection pressure provided by the elite-preservation, thereby making a balance between exploration and exploitation issue. However, for non-elitist RGAs, the line and associated recombinations are better, as they both help in maintaining the linkage needed to solve the problem in a computationally faster manner.

Figure 11 shows the performance on Rosenbrock function. It is clear that (i) elitist RGAs have performed much better than the non-elitist approach, and (ii) associated and line recombinations perform better. Rosenbrock function is also fully-linked and conclusions similar to that in Schwefel function are also observed here.

Finally, we consider a modified Schwefel function in which a partial linkage is introduced:

$$f_{\text{msch}}(x_1, x_2, \ldots, x_{20}) = \sum_{k=1}^{4} \sum_{i=1}^{5} \left(\sum_{j=1}^{i} x_{5(k-1)+j} \right)^2 \tag{5}$$

The optimal solution is still the same as that of Schwefel function. The first five variables are linked and x_6 to x_{10} are linked to each other, and so on. But there is no linkage between x_1 and x_6, for example. Here, we also apply the proposed linkage-based recombination scheme. Figure 12 shows the performance of all four recombination schemes. It is observed that elitist RGAs perform better with assorted recombination and with the proposed linkage-based schemes. Interestingly, the line recombination performs poorly, due to existence of partial linkage in this problem, as discussed earlier.

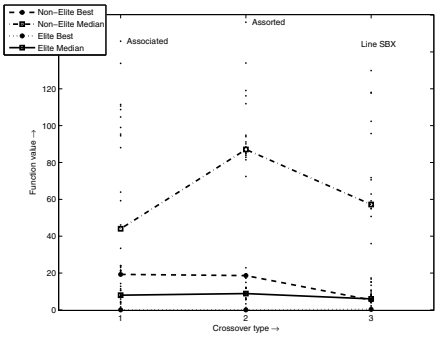


Fig. 11. Best and median performance on Rosenbrock function with elitist and non-elitist RGAs

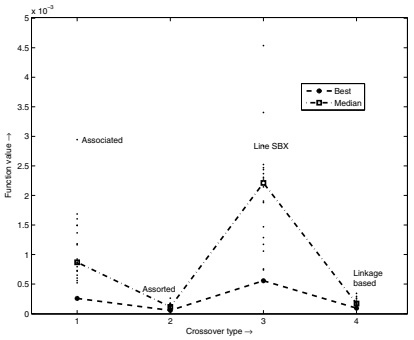


Fig. 12. Best and median performance on modified Schwefel function with elitist RGAs

5 Conclusions

In this paper, we have explored three different recombination schemes for forming an offspring solution vector using created variable values by the well-known variable-wise SBX operator. It has been observed that for low-linkage problems, a random assortment scheme performs better while for high-linkage problems, an associated recombination is a more reliable approach. Elite preservation helps the assorted recombination scheme. Based on these observations, we have devised a linkage-based recombination scheme that performs associated recombination to linked variables and assorted recombination to non-linked variables. In all cases, the linkage-based recombination has performed well. Applications to two well-known test problems agree with our findings. The proposed methods now must be compared with other contemporary real-parameter optimization methods, such as PSO, DE, CMA-ES and others. Further research must also be made in developing procedures for identifying linkage in a problem so that efficient recombination schemes, such as the proposed linkage recombination scheme, can be used to make a better and faster search.

References

1. Deb, K., Agrawal, R.B.: Simulated binary crossover for continuous search space. Complex Systems 9(2), 115–148 (1995)
2. Harik, G., Goldberg, D.E.: Learning linkages. In: Foundations of Genetic Algorithms 4 (FOGA-4), pp. 247–262 (1996)
3. Munetomo, M., Goldberg, D.E.: Linkage identification by non-monotonicity detection for overlapping functions. Evolutionary Computation 7(4), 377–398 (1999)
4. Deb, K., Sinha, A., Kukkonen, S.: Multi-objective test problems, linkages and evolutionary methodologies. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2006), pp. 1141–1148. The Association of Computing Machinery (ACM), New York (2006)

Scalable Fuzzy Genetic Classifier Based on Fitness Approximation

Harihar Kalia[1], Satchidananda Dehuri[2], and Ashish Ghosh[3]

[1] Department of Computer Science and Engineering,
Seemanta Engineering College,
Jharpokharia, Mayurbhanj, 757086, India
smarahari07@gmail.com
[2] Department of Systems Engineering,
Ajou University, San 5, Woncheon-dong,
Yeongtong-gu, Suwon 443-749,
Republic of Korea
satchi.lapa@gmail.com
[3] Center for Soft Computing Research,
Indian Statistical Institute,
203, B.T.Road, Kolkata, 700108, India
ash@isical.ac.in

Abstract. Fuzzy classification rules are widely used for classification, as they are more interpretable as well as efficient in handling the real-world problems, which involves imprecision and vagueness. Genetic algorithms are proven stochastic search techniques employed in automatic generation of fuzzy classification rule. However, genetic algorithms employed for the said task require large number of fitness evaluation or performance evaluations in achieving a reasonable solution requiring a large amount of computational time. Hence, to expedite the execution is a major concern in genetic algorithms. In this paper, we incorporate fitness inheritance mechanism in genetic algorithms to design a scalable genetic fuzzy classifier, which reduce the number of actual fitness function evaluations of subsequent generations and produce rules with acceptable classification accuracy.

Keywords: Fitness inheritance, genetic algorithm, fitness evaluation, fuzzy classification.

1 Introduction

Extraction of useful knowledge from large databases is a long standing problem in the community of data mining researchers. Classification is one of the fundamental tasks in data mining [1]. Also real-life data contains uncertainty, noise, and multiple instances with overlapping attributes, which are difficult to handle by the conventional classification method/algorithm based on crisp set. Fuzzy classification rules

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 492–499, 2012.

efficiently handle these data [12]. Additionally, fuzzy classification rules are more human understandable, owing to the inclusion of linguistic variables.

Many approaches have been proposed for fuzzy classification rule generation such as heuristic procedures [4], neuro-fuzzy techniques [5], and data mining based technique [8], etc. Genetic algorithms are proved heuristic search techniques used by researchers [6, 7] in achieving these goals, while designing fuzzy classifier. But it has been observed that the time complexity of genetic algorithms is high, and the major share of it is from fitness evaluation.

Fitness inheritance [2, 3], one of the fitness approximation methods, uses the fitness of the parents, instead of calculating the fitness of offspring used as a remedy to the above mentioned problem. In this paper, we develop an algorithm based on this concept for extracting fuzzy classification rules with moderate time requirement.

Rest of the paper is organized as follows. Sections 2 and 3 discusses the preliminaries and scalable genetic algorithms respectively. Section 4 contains a case study with corresponding numerical results. Section 5 provides conclusion and future research proposal.

2 Preliminaries

2.1 Fuzzy Rule Based Classifier

2.1.1 Fuzzy Rule Generation

Fuzzy classification rules are represented as *if-then* rules with linguistic values [6]. For *M*-class, *d*-dimensional classification problem, the fuzzy *if-then* rule is represented as follows:

$$If\ x_1\ is\ A_1\ and\ x_2\ is\ A_2\ and\ \ldots\ldots and\ x_d\ is\ A_d\ \ then\ \ class\ \ C_k,\ 1 \le k \le M\ , \qquad (1)$$

where $X =< x_1, x_2, \ldots, x_d >$ is a d-dimensional pattern vector, $A_1, A_2, \ldots, A_k$ are antecedent fuzzy set and C_k is the class label of the pattern. Antecedents of the fuzzy rules are linguistic values like “small”, “medium”, “high” or don’t care.

The consequent class C_k of each fuzzy rule is determined by the following widely used procedure proposed in [6].

- Calculate the compatibility grade of each training pattern $x_q =< x_{q1}, x_{q2}, \ldots, x_{qn} >$ with the fuzzy *if-then* rule *R* as follows:

$$\mu(x_q) = \mu_1(x_{q1}) \times \mu_2(x_{q2}) \times \cdots \times \mu_n(x_{qn}), \qquad (2)$$

  where $\mu_j(x_{qj})$ is membership function of the antecedent fuzzy set *A*.

- Calculate the total grade of compatibility of the training patterns with the rule *R* for each class.

$$\beta_{Class\ \ h}(R) = \sum_{x_q \in Class\ \ h} \mu(x_q), \tag{3}$$

where $\beta_{Class\ \ h}(R)$ is the sum of the compatibility grades of the training pattern in *Class h* with the fuzzy rule *R*.

- Find the class having maximum β value corresponding to the rule *R* and designate that class as the consequent of the rule R. If two or more classes take the same maximum value, then the consequent of the rule *R* cannot be unique and it will be designated as ϕ. If none of the training patterns is compatible with the rule *R*, then $\beta_{Class\ \ h}(R) = 0$. Then the consequent of the rule is specified as ϕ.

2.1.2 Classification of New Patterns

Let Q be the set of fuzzy rules generated by the heuristic procedure as above. Then a new input pattern $x_q =< x_{q1}, x_{q2}, \ldots x_{qn} >$ can be classified using fuzzy reasoning method, called single winner rule in Q, described below.

- Calculate the compatibility grade of the pattern with all the rules in Q.
- Classify the new pattern, same as the consequent of the rule, which has the highest compatibility grade.

2.2 Fitness Approximation in Genetic Algorithm

2.2.1 Genetic Algorithms

Genetic algorithms [11] are probabilistic search method, which generates solutions using nature inspired evolution techniques such as inheritance, selection, crossover, and mutation.

A standard genetic algorithm begins with a randomly generated population, consisting of chromosomes, which composed of genes. The individuals are probabilistically selected based on their fitness value. The individuals become similar when almost 95% of genes share the same value. The population converges when all the individuals becomes similar. The three basic genetic operators selection, crossover, and mutation are used in generating new individuals. Selection is a mechanism which guides GAs for appropriate selection of individuals.

Majority of the genetic algorithms necessitate a bulky number of fitness evaluations, before generating acceptable solutions. So the optimization process involving genetic algorithms become very expensive, which demands some means of relaxation, for computational efficiency. Fitness approximation, in particular fitness inheritance technique is a proposed mean to enhance the efficiency of genetic algorithms.

2.2.2 Fitness Inheritance

Fitness inheritance is an enhancement technique, originally proposed by Smith [2], allows reduces the number of actual fitness evaluations and hence reduces the computational cost. The basic idea is to estimate the objective functional value of an offspring based on the fitness value of its parents. That is in subsequent generations we do not have to evaluate each individual but get its fitness from a surrogate, estimated value of some individuals of its parent.

Incorporating a fitness inheritance mechanisms in genetic algorithms, at first the fitness of all individuals of initial population are evaluated. Then in subsequent populations, fitness values of some proportion of individuals are inherited and fitness of other individuals are calculated as usual. This "proportion" is a parameter called fitness proportion *p* [9, 10]. This parameter is fixed by users, as in crossover and mutation that specifies the probability with which an individual inherits its fitness from its parents and hence determines the savings on the number of evaluations performed. If *p=0*, then none of the individuals inherit fitness value, hence no speed up. On the other hand if *p=1*, then all the individuals get inherited fitness value, which speedily reduces the population diversity and results in premature convergence.

Fitness of the individuals can be calculated by taking the average of the fitness value of the parents. That is, if the initial population has *m* individual, having fitness f_1, f_2,..., f_m then the inheritance fitness value for individuals in offspring is:

$$f_h = \frac{f_1 + f_2 + \cdots + f_m}{m}, \tag{4}$$

where f_h denotes the inherited fitness value.

3 Scalable Genetic Algorithms in Fuzzy Rule Mining

We propose a scalable genetic algorithm, incorporating fitness inheritance technique by using the concept of intra-population and inter-population inheritance for fuzzy classification rule mining. Figure 1 shows the flow diagram of our scalable genetic algorithm.

For inheriting fitness value, we use the method defined in equation (4). That is, the average fitness value of the individuals in parent population is inherited for offspring.

The individuals in offspring population are searched, which are the same as the individuals in the parent population illustrated in Figure 2(A). In the offspring population only three individuals are new and others are the same with initial population. The fitness values of these same individuals are inherited for the offspring population, instead of calculating it again. We called it inter- population inheritance.

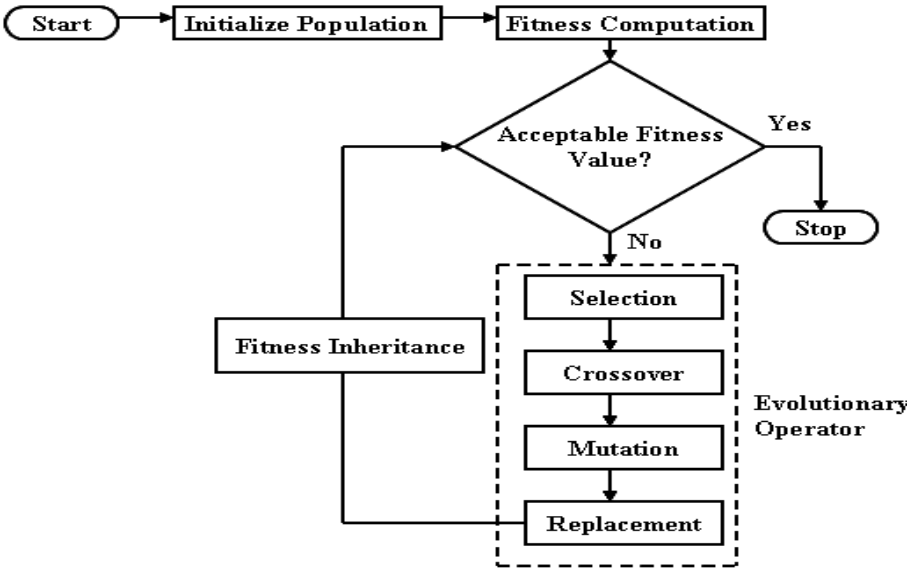


Fig. 1. Flow Diagram of Scalable Genetic Algorithm

The individuals in offspring pool are searched, which are the same as the individuals in the parent population illustrated in Figure 2(B). In the offspring pool only three individuals are new and others are the same with initial population. The fitness values of these same individuals are inherited for the offspring pool, instead of computing it again. We called it inter population inheritance.

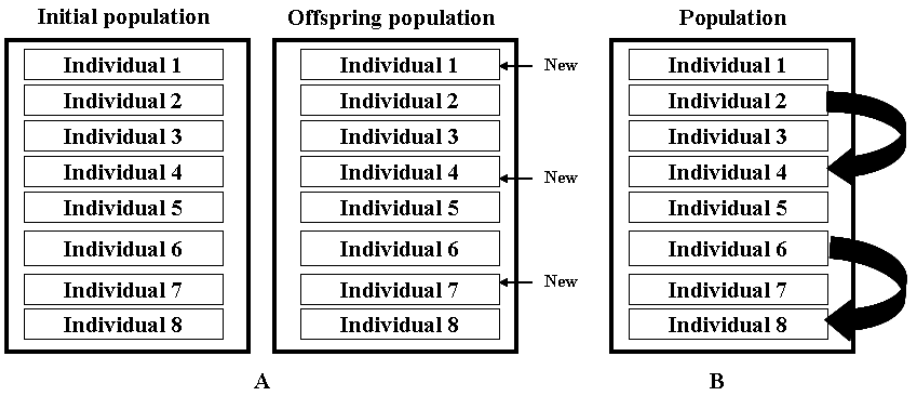


Fig. 2. (A) Inter-Population Similarity, (B) Intra-Population Similarity

Subsequently we find out the repeated individuals in the offspring pool. Hence, instead of computing the fitness of a similar individual, we compute for one and copy it for others. This is termed as intra-population inheritance. In Figure 2(A), we have individuals 2 and 4 and individuals 6 and 8 as the same. So we need to compute the fitness value for two individuals instead of four.

The main advantage of the proposed method is that it reduces the computational complexity of both fuzzy rule generation and optimization.

The complexity of the fitness evaluation for a single generation of the conventional genetic algorithm is n* O (m) = O (m*n), where 'n' is the number of individuals and O (m) is the time of computing the fitness. In fitness evaluation, for each rule we need at least | D| comparisons, where |D| is the size of the database and for 'n' rule (number of individuals) n*|D| comparisons are needed. In the sequel, the fitness evaluation of the pool may be reduced with the help of inter and intra interactions. Hence, the overall complexity of the fitness evaluation for a single generation is << O(m*n).

In consequent class determination of the fuzzy rules, we need not have to compute, at least for 50% of the rule by intra-population inheritance and inter- population similarity concept, which saves computation time in fuzzy rule mining, which is clearly an advantage of our method.

4 Experimental Study and Results

To evaluate the accuracy and efficiency of the proposed algorithm we have used the IRIS dataset available at http:// www.ics.uci.edu/~mlearn. We have implemented the method in MATLAB on a personal computer with Intel Core 2 Duo processor and 2GB RAM. The dataset is normalized using the equation (5).

$$\frac{A - A_{\min}}{A_{\max} - A_{\min}}, \tag{5}$$

where A , $A_{\min}$, $A_{\max}$ respectively represent the original attribute value, minimum and maximum of attribute values.

Triangular membership function and three fuzzy sets namely small, medium, large and don't care are used.

The membership functions of these sets are defined as follows:

$$\mu_{small}(x) = \max\left(\frac{0.5 - x}{0.5}, 0\right), \mu_{medium}(x) = \max\left(\min\left(\frac{x}{0.5}, \frac{1 - x}{0.5}\right), 0\right),$$

$$\mu_{large}(x) = \max\left(\frac{x - 0.5}{0.5}, 0\right), \text{ and } \mu_{don'tcare}(x) = 0.$$

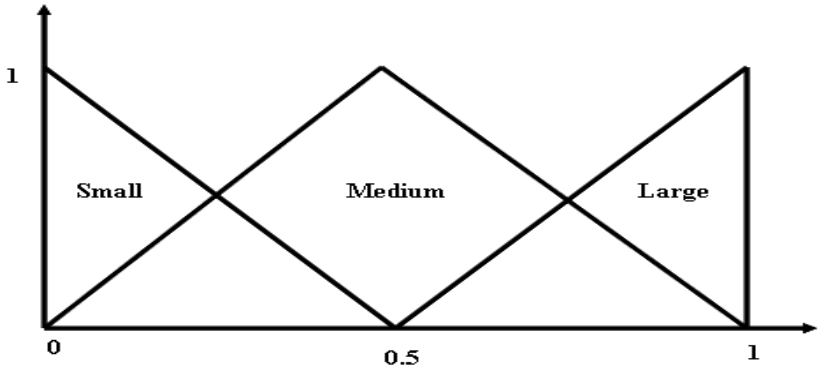


Fig. 3. Fuzzy sets using triangular membership function

Pittsburg approach of rule encoding method is used, where single chromosome represents the whole rule set using a binary encoding scheme. We use 4-rules for each class and 2-bits to represent each fuzzy set. That is, 00 for don't care, 01 for large, 10 for medium and 11 for small.

Following parameters are used in the experiment:

Population Size	100	Number of generation	100
Mutation probability	0.1	Crossover probability	0.7

Tournament selection is adopted. We also adopt elitism, where we keep 20% of good chromosome in each generation.

Without sacrificing our claim (saves computation time), we achieve an average classification accuracy of 96.2%, which is a good achievement over Ishibuchi approach [7] but competitive to Castro approach [8].

Table 1. Classification Accuracy

Classifier	Average Accuracy	Inter/Intra Inheritance
H. Ishibuchi et al.[7]	94.6%	NO
Castro et al. [8]	96.9%	NO
Our approach	96.2%	YES

Overall, the scalability achieved in the proposed approach is 25% compared to non-scalable approach by suppressing other likelihood overheads.

5 Conclusions

We conclude that using our fitness inheritance mechanism we obtain a reduction of tolerance in fitness evaluation and hence time complexity of the proposed algorithm is reduced vis-à-vis to simple genetic algorithm. However, our approach needs extra

memory space because for comparing the individuals with the individuals of the previous population we need all the ancestor populations to be stored.

Our future research includes: i) further validation with more number of datasets; and ii) extend this idea for solving multi-objective problems with more realistic data.

References

1. Fayyad, U., Piatetsky-Shapiri, G., Smyth, P., Uthurasamy, R.: Advances in knowledge discovery and data mining. AAAI Press/MIT Press
2. Smith, R.E., Dike, B.A., Stegmann, S.A.: Fitness inheritance in genetic algorithms. In: Proceedings of the ACM Symposium on Applied Computing, pp. 345–350. ACM Press, New York (1995)
3. Sastry, K., Goldberg, D.E., Pelikan, M.: Don't evaluate, inherit. In: Proceedings of Genetic and Evolutionary Computation Conference, pp. 551–558. Morgan Kaufmann (2001)
4. Ishibuchi, H., Nozaki, K., Tanaka, H.: Distributed representation of fuzzy rules and its application to pattern classification. Fuzzy Sets and Systems 70(1), 21–32 (1992)
5. Nauck, D., Kruse, R.: A neuro-fuzzy method to learn fuzzy classification rules from data. Fuzzy sets and Systems 89(3), 277–288 (1997)
6. Ishibuchi, H., Nozaki, K., Yamamoto, T., Tanaka, H.: Selecting fuzzy if-then rules for classification problems using genetic algorithms. IEEE Transaction on Fuzzy Systems 3(3), 260–270 (1995)
7. Castro, P.A.D., Camargo, H.A.: Improving the genetic optimization of fuzzy rule base by imposing a constraint condition on the number of rules. Proceedings of SBC, 72–981 (2005)
8. Hu, Y.-C., Chen, R.-S., Tzeng, G.-H.: Finding fuzzy classification rule using data mining criteria. Pattern Recognition Letter 24, 509–519 (2005)
9. Reyes-Sierra, M., Coello Coello, C.A.: A study of fitness inheritance and approximation techniques for multi-objective particle swarm optimization. In: Congress on Evolutionary Computation, pp. 65–72 (2005)
10. Reyes-Sierra, M., Coello Coello, C.A.: Dynamic fitness inheritance proportion for multi-objective particle swarm optimization. Technical Report EVOCINV-03-2006, Evolutionary Computation Group at CINVESTAV, México, pp. 1–22 (2006)
11. Goldberg, G.E.: Genetic Algorithms in search optimization and machine learning. Addition Wesley, New York (1998)
12. Kuncheva, L.I.: 'Fuzzy' vs 'Non-fuzzy' in combining classifiers designed by boosting. IEEE Transactions on Fuzzy Systems 11(6), 729–741 (2003)

Multi Objective Integrated Layout Design Problem

I. Jerin Leno[1], S. Saravana Sankar[2], and S.G. Ponnambalam[3]

[1] National college of Engineering, Maruthakulam, Tamil Nadu, India
[2] Kalasalingam University, Krishanankoil, Tamil Nadu, India
[3] Department of Mechatronics, Monash University, Petaling Jaya, Malaysia
lenojerin@yahoo.co.in

Abstract. Traditionally the design of Inter-cell layout and Material Handling System (*MHS*) of the manufacturing system is being carried out in step by step. This leads to sub-optimal solutions for facility layout problems (FLP). In this work an attempt is made to concurrently design Inter-cell layout and the *MHS* using a Genetic Algorithm (GA) based methodology using simulated annealing algorithm (SAA) as local search tool for a Cellular Manufacturing System (*CMS*) environment under open field configuration. The proposed algorithm is employed to simultaneously optimize two contradicting objectives viz. 1. Total material handling cost 2. Distance weighted cost of closeness rating score. The algorithm is tested on two different bench mark layouts and with different initial problem data sets. It is found that the proposed algorithm is able to produce approximate pareto-optimal solutions.

Keywords: Integrated Layout Design, Genetic Algorithm, Mutiobjective optimization.

1 Introduction

Facility layout design is an important issue for any industry, as a poor layout may degrade overall efficiency of the production system. Traditionally, the layout design is being carried out in a sequential manner involving two major steps [1,2], viz (1) Inter-cell Layout design wherein the exact location, orientation, position of input/output stations of each department is determined, (2) Material Handling System (MHS) design wherein the flow path of the materials between input and output stations of different department is determined. Traditionally, these two design phases are performed sequentially and separately in a step by step manner, the design procedure invariably leads to solution that can be far from the total optimum [3, 4]. Owing to computational complexity very little work has been done to solve Integrated *CSL* and *MHS* design problem simultaneously .In the recent years researchers have focused on concurrent design of both inter-cell and *MHS* design by adopting integrated approach [5-8].

In this work an attempt has been made to develop a methodology to solve a Multi objective integrated layout design problem (MOILDP). The proposed algorithm is experimented with two problem instances and found consistent in building satisfactory pareto-optimal solution set.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 500–508, 2012.

2 Problem Description

There are N number of cells which are to be placed in a production floor layout of width W and height H. The cells are considered to be rectangular blocks with known dimension. Given the width w and height h of the individual cell (determined by size and shape of the facilities), the predefined Input and Output stations at the boundary of the cell, quantum and frequency of material flow between the cells, the aim is to find the exact location (x and y coordinates), the orientation of the individual cells, and to decide the aisle distance between the cells (along the department perimeter) with the objective of minimizing the total material handling cost and distance weighted cost of total closeness rating score.

In any facility layout design problem while it is imperative to minimize the total material handling cost which is directly proportional to the distance between the cells, often it also essential to place certain cells as farthest (nearest) as possible depending upon the nature of the production process, the safety issues and the like. Hence these two objectives are considered in this work. The notations used in the model are:

N the total number of cells in the layout

W the width of the floor space

H the height of the floor space

i,j indices to denote cells

f_{ij} the directed flow density from cell i to cell j

w_i width of cell i in the initial orientation

h_i height of cell i in the initial orientation

(I_i^x, I_i^y), (x_i^I, y_i^I) local coordinates, spatial coordinates of the input station of cell i

(O_i^x, O_i^y), (x_i^O, y_i^O) local coordinates, spatial coordinates of the output station of cell i

(x_i, y_i), (x_i', y_i') spatial coordinates of the lower-left corner, the upper- right corner of cell i

l_{ij} equals 1 if cell i is placed to the left of cell j ; (that is $x_i' \le x_j$) and 0 otherwise

b_{ij} equals 1 if cell i is placed below cell j ; (that is $y_i' \le y_j$) and 0 otherwise

d_{ij} shortest contour distance from the output station of cell i to the input station of cell j

$$(u_i, v_i) = \begin{cases} (0,0) \text{ cell } i \text{ in its original orientation} \\ (1,0) \text{ cell } i \text{ is rotated } 90^{\circ} \text{ clockwise from its original orientation} \\ (0,1) \text{ cell } i \text{ is rotated } 180^{\circ} \text{ clockwise from its original orientation} \\ (1,1) \text{ cell } i \text{ is rotated } 270^{\circ} \text{ clockwise from its original orientation} \end{cases}$$

c_{ij} the cost of travel of unit material for unit distance between cell i and j, $c_{ij} = 1 \quad \forall i, j$

r_{ij} closeness rating score between cell i and j

The mathematical model for the Multi objective integrated layout design problem (MOILDP) is formulated based on [5], [9] and shown below:

$$\text{Minimize TMHC} = \sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij} f_{ij} d_{ij} + P \tag{1a}$$

$$\text{Minimize TCRS} = \sum_{i=1}^{N}\sum_{j=1}^{N} r_{ij} d_{ij} + P \tag{1b}$$

Subject to:

$$x_i' = x_i + (1-u_i)w_i + u_i h_i \qquad \forall i \tag{2}$$

$$y_i' = y_i + (1-u_i)h_i + u_i w_i \qquad \forall i \tag{3}$$

$$x_i^{I(O)} = x_i + (1-u_i)(1-v_i)I(O_i^x) + I(O_i^y)u_i(1-v_i) + (w_i - I(O_i^x))(1-u_i)v_i + (h_i - I(O_i^y))u_i v_i \qquad \forall i \tag{4}$$

$$y_i^{I(O)} = y_i + (1-u_i)(1-v_i)I(O_i^y) + (w_i - I(O_i^x))u_i(1-v_i) + (h_i - I(O_i^y))(1-u_i)v_i + I(O_i^x))u_i v_i \qquad \forall i \tag{5}$$

$$l_{ij} + l_{ji} + b_{ij} + b_{ji} \geq 1 \qquad \forall i \tag{6}$$

$$x_i' \leq l_{ij} x_j + W(1-l_{ij}) \qquad \forall i < j \tag{7}$$

$$y_i' \leq b_{ij} y_j + H(1-b_{ij}) \qquad \forall i < j \tag{8}$$

$$x_i', y_i', x_i^{I(O)}, y_i^{I(O)} \geq 0 \qquad \forall i \tag{9}$$

$$u_i, v_i \in \{0,1\} \qquad \forall i \tag{10}$$

$$l_{ij}, b_{ij} \in \{0,1\} \qquad \forall i,j \tag{11}$$

Where, $P = \alpha(P_w + P_h)$ is a penalty term the guarantee that the layout solution satisfies the following floor boundary condition $x_i' \leq w \ \forall i,\ y_i' \leq H \ \forall i$, $P_w = \max\left\{0, \max_i\left\{x_i'\right\} - W\right\}, P_h = \max\left\{0, \max_i\left\{y_i'\right\} - H\right\}$, and α = the weight of penalty and was set to be algebraic sum of flow interaction between each pair of departments.

Constraints (2) and (3) define the x-coordinate of the right boundary and the y-coordinate of the upper boundary of each cell. Constraints (4) and (5) are used to specify the x and y coordinates of I/O stations for each cell. These coordinates are expressed in generalized terms with respect to the lower-left corner point of the cell under the horizontal configuration that is before considering rotation.

Constraints (6) (7) and (8) are to ensure that there is no overlap between any pair of cells by letting each pair of cells be separated in the x or y direction. Constraints (9) (10) and (11) specify the bounds for each variable. The assigned closeness rating score is as follows: A=5; E=4;I=3;O=2;U=0 and X=-5 (undesirable).

3 Proposed Methodology

In this work, a GA-SAA based heuristic is implemented for the optimization purpose to obtain the pareto-optimal solution set which minimizes 1. The total material handling cost (TMHC) and 2. Distance weighted cost of total closeness rating score (TCRS).

3.1 Solution Representation

In a GA approach feasible solutions to the problem are encoded into a string of decision choices that resemble chromosomes. The chromosome that represents a feasible solution is shown in Fig 1.

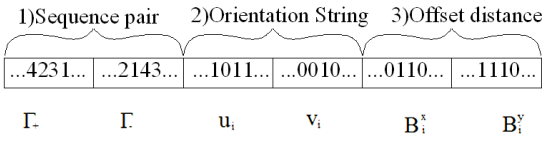


Fig. 1. Chromosome Structure

The chromosome string consists of three parts. For a layout problem of *N* cells, the first part is first and second sequence (Γ_+ and Γ_-) of sequence pair, the second part is binary code of 2*N* bits that represents u_i and v_i of each cell, and the last part is 2*N* bytes which helps to define the offset distances in the x direction and y direction for each cell.

3.1.1 Sequence Pair Representation

A cell system layout (*CSL*) can be represented by a unique sequence pair [10] describing the topology of the cell placement. A layout consisting of cells (*a*,*b*,*c*). The dimensions for every cells are: *a*(10 ×5), *b*(5×5), *c*(4×8) and it's corresponding *CSL* is shown in Fig. 2 which can be represented by a *SP* = (*bac* ; *abc*).

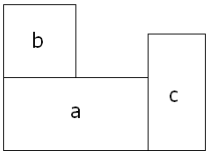


Fig. 2. CSL for *SP* = (*bac*; *abc*)

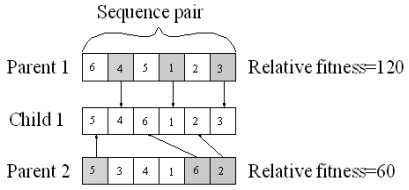


Fig. 3. Crossover operation on the first part *SP* = (Γ_+, Γ_-)

3.2 Fitness Evaluation

The decoding of a chromosome and finding the objective function value for a feasible solution is done in four steps.

Step 1: Finding the spatial coordinates of the each department. Using first and second part of the chromosome, and the sequence pair evaluation algorithm Algorthim1 found in the literature [11] the spatial coordinates of the lowest left corner of each cell in a *CSL* is computed.

Step 2: Finding the spatial coordinates I/O stations. Once the spatial coordinates of each department is obtained, the algorithm is able to generate any CSL that has the topology defined by the corresponding sequence-pair. Then the spatial coordinates of I/O stations can be determined using constraints 4 and 5.

Step 3: Computing the optimal travel distance between the departments (d_{ij}). At first the grid graph [5] is constructed and then through repeated applications of Dijkstra's algorithm [12] the shortest path distance along the department perimeter is determined. The obtained shortest path distance d*ij* along the department perimeter is unique for corresponding CSL.

Step 4: Objective function value calculation. TMHC and TCRS are the objective function values, OF_1 and OF_2, calculated based on the equations (1_a) and (1_b) respectively.

3.3 GA Operators

3.3.1 Selection

The selection module is constructed on the basis of Roulette wheel [13] mechanism. The probability of selection for each chromosome is based on a fitness value relative to the total fitness value of the population.The Fitness value for this work is (F) = 1/COF. Where COF= (TMHC+TCRS)/2.

3.3.2 Crossover

For the first part, a crossover operator similar to [14] was implemented for first and second sequence of the sequence pair independently. The first child is constructed by randomly picking a gene from the first parent and placed it in a child string at the same location as its position in the parent sequence. This process is continued for *k* cells where *k* is proportional to the relative fitness of the first parent. The missing integers in the first child are filled in the same order as they appear in the second parent. Similarly the second child string is created by reversing the selection order of two strings. For the last two parts of a chromosome, a heterosexual one-point crossover [15] was adopted. An example of this crossover operator is shown in Fig. 3 and Fig. 4

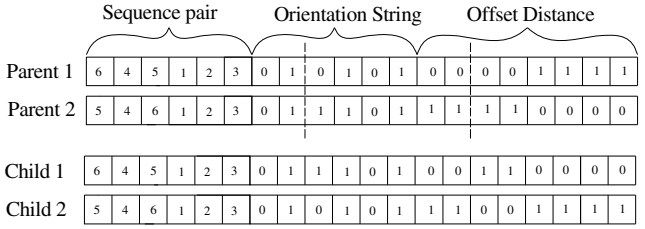


Fig. 4. Crossover operation on the second and third part

3.3.3 Mutation

For first and second sequence of the first part of the chromosome the mutation operator involves a random selection and swapping of two integers. For the second part, the mutation operator involves randomly altering one symbol to another. For the last part, the mutation operation involves replacing a randomly chosen byte with a new value generated at random with range of [0, 255]. An example of the three types of mutation operators is shown in Fig. 5

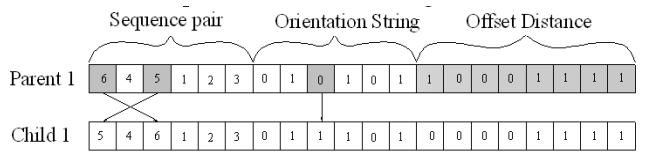


Fig. 5. Mutation operation on the first, second and third part of the chromosome

3.3.4 Neighborhood Generation

The neighborhood solution in the vicinity of current solution is generated by the mutation operation discussed in section 3.3.3.

3.3.5 Non-dominated Sorting of Solutions

The initial GA population is added to pareto-optimal set (*POS*). During the search in SAA, every solution is compared with solutions of the *POS* and tested for non-dominance. If the particular solution is not dominated by any of the solutions in the *POS*, it is added on to the *POS*. With the addition of any member to the *POS*, if any existing member of *POS* is found to be dominated with respect to OF_1 and OF_2 it is removed from *POS*.

3.4 Control Parameters

The control parameter values for GASAA was determined based on pilot study which produces satisfactory output are summarized as below. For GA, Population size (P_s) = 5 $\times N$, Cross over probability (p_c) = 0.6, Mutation probability (p_m) = 0.20 was set as parameter values. For SAA, Initial Temperature (T_i) =300, Final Temperature (T_f) =0, Cooling Rate (cr)=0.85, Number of perturbation at each temperature drop (N_p)=5.

4 Results and Discussion

The proposed genetic algorithm based procedure was coded in MATLAB and implemented in Dual core processor with 2GB RAM. Experiments were conducted using the bench mark problems [8] and [16] found in the literature. For each bench mark problem 10 different initial population set, each population set having 20 different initial solutions were generated at random. The final layout obtained by GA is reported in Fig 6. The pareto-optimal solution set obtained and the pareto frontier is shown in Table 1 and Fig 7 respectively. The average computational time taken by the algorithm to reach the optimal solution is given in Table 2.

(a) 7 department (b) 12 department

Fig. 6. An example of final layouts obtained by GA-SAA

Table 1. List of pareto-optimal solutions

7 department				12 department			
GA		GA-SAA		GA		GA-SAA	
OF_1	OF_2	OF_1	OF_2	OF_1	OF_2	OF_1	OF_2
388551	198000	250000	14800	8140	8244	8000	8118
458550	196900	265000	14603	8350	7685	8254	7356
466056	196300	278560	13700	8740	7206	8450	7012
471050	195505	289950	12504	9243	7118	8900	6923
473550	195200	296542	11689	10820	7044	9489	6856
478552	190401	305423	9502			9650	6755
506060	189400	312455	8545				
508550	188600	335666	7856				
516050	188000	348650	7562				
548550	184800	365220	6955				
646050	182000	385445	6854				
713550	181700	457650	5855				
756050	181200	550250	4980				

OF_1 – Objective Function 1 value (TMHC); OF_2 - Objective Function 2 value (TCRS)

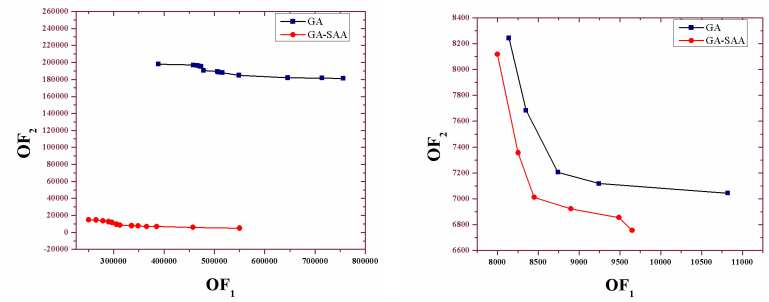


Fig. 7. Pareto-optimal curve for the test problems

Table 2. Average Computational time (s)

No of departments	7	12
Proposed GA	465	2910
Proposed GA-SAA	1150	7890

From figure 7 it is evident that proposed GA-SAA is good in finding efficient frontier than GA for both the test problem instances. The convergence characteristic of GA-SAA is improved by adopting a suitable local search mechanism using SAA.

5 Conclusion and Scope for Future Research

To overcome the limitations out of sequential design procedure in the Layout design, integrated design of *CSL* and MHS was adopted. Again, to simultaneously optimize two different objective of the layout design problem a Genetic Algorithm based procedure (GA-SAA) is proposed in this work. The proposed algorithm was tested with two different problems of different problem sizes to concurrently optimize two objectives namely TMHC and TCRS. It is found that the proposed algorithm is able to produce satisfactory paret-optimal solutions within an acceptable computational time limit than standard GA.

The outcome of this research leaves scope for further research towards employing a local search mechanism to further reduce the computational time. To provide wide range of alterative solutions to the implementers, a NSGA-II and AMOSA based multi objective optimization algorithm can also be developed to produce a pareto optimal front.

References

1. Montreuil, B.: Integrated design of cell layout, input/output station configuration, and flow network of manufacturing systems. Research Memorandum No. 87± 9, School of Industrial Engineering, Purdue University (1987)

2. Montreuil, B., Ratliff, H.D.: Optimizing the location of input/output stations within facilities layout. Engineering Cost and Production Economics 14, 177–187 (1988)
3. Kim, J.G., Kim, Y.D.: A space partitioning method for facility layout problems with shape constraints. IIE Trans. 30, 947–957 (1998)
4. Norman, B.A., Arapoglu, R.A., Smith, A.E.: Integrated facilities design using a contour distance metric. IIE Trans. 33, 337–344 (2001)
5. Hu, G.H., Chen, Y.P., Zhou, Z.D., Fang, H.C.: A genetic algorithm for the inter-cell layout and material handling system design. Int. J. Adv. Manuf. Technol. 34, 1153–1163 (2007)
6. Aiello, G., Enea, M., Galante, G.: An integrated approach to the facilities and material handling system design. Int. J. Prod. Res. 40, 4007–4017 (2002)
7. Ho, Y.C., Moodie, C.L.: A hybrid approach for concurrent layout design of cells and their flow paths in a tree configuration. Int. J. Prod. Res. 38, 895–928 (2000)
8. Wu, Y., Appleton, E.: The optimization of block layout and aisle structure by a genetic algorithm. Comput. Indust. Eng. 41, 371–387 (2002)
9. Khare, V.K., Khare, M.K., Neema, M.L.: Combined computer aided approach for the facilities design problem and estimation of the distribution parameter in the case of multigoal optimization. Comput. Ind. Eng. 14, 465–476 (1988)
10. Murata, H., Fujiyoshi, K., Kajitani, Y.: VLSI module placement based on rectangle-packing by the sequence-pair. IEEE Trans. CAD Integ. Circ. Syst. 15, 1518–1524 (1996)
11. Tang, X., Wong, D.F., Tian, R.: Fast Evaluation of Sequence Pair in Block Placement by Longest Common Subsequence Computation. In: Design, Automation and Test in Europe, p. 106 (2000)
12. Cormen, T.H., Leiserson, C.E., Rivest, R.L.: Introduction to algorithms. McGraw-Hill and MIT Press, New York (1990)
13. Goldberg, D.E.: Genetic algorithms in search, optimization, and machine learning. Addison-Wesley, USA (1989)
14. Kochhar, J.S., Foster, B.T., Heragu, S.S.: HOPE: A genetic algorithm for the unequal area facility layout problem. Comput. Oper. Res. 25, 583–594 (1998)
15. Poli, R., Langdon W.B.: Genetic programming with one-point crossover. In: Second On-line World Conference on Soft Computing in Engineering Design and Manufacturing, pp. 23–27. Springer, London, (1997)
16. Welgama, P.S., Gibson, P.R.: A construction algorithm for the machine layout problem with fixed pick-up and drop-off points. Int. J. Prod. Res. 11, 2575–2590 (1993)

Dynamic Network Traffic Data Classification for Intrusion Detection Using Genetic Algorithm

Rahul Mitra[1], Sahisnu Mazumder[1], Tuhin Sharma[1],
Nandita Sengupta[2], and Jaya Sil[1]

[1] Dept. of Computer Science and Technology,
Bengal Engineering and Science University,
Shibpur, Howrah -711 103, West Bengal, India
[2] University College of Bahrain, Bahrain
{sahisnumazumder,tuhinsharma121,rahulm9999}@gmail.com,
ngupta@ucb.edu.bh, js@cs.becs.ac.in

Abstract. Intrusion Detection System (IDS) classifies network traffic data either ′anomaly′ or ′normal′ to protect computer systems from different types of attacks. In this paper, data mining concepts and genetic algorithm have been applied to classify online traffic data efficiently by developing a rule based lazy classifier. The proposed method updates the rule set dynamically to accommodate the changing pattern in the traffic data in order to attain highest classification accuracy and at the same time maintaining consistency. The classifier is able to detect variants of common network traffic data patterns or modified existing security attacks based on the knowledge gained from its existing training data set with significant classification accuracy.

Keywords: discretization, cut generation, center-spread encoding, nsl-KDDCUP'99 network traffic data set classification accuracy.

1 Introduction

Intrusion Detection System (IDS) [1, 2], a quick evolving domain recently gained its importance in the field of advancement of network security solutions. The current IDSs fail to predict appropriate decision due to problem of 'data overload' resulting classification of normal traffic as malicious (false positive) or vice versa (false negative). Significant advancement has already been done to protect the systems by designing extremely complex security infrastructure including firewalls, identification and authentication systems, access control products, VPNs, encryption products, different antivirus software, scanners and many more. However, yet no system able to handle sufficient and optimum way to the ever changing and evolving innovative security attacks fired intentionally and/or deliberately by a serious attacker in a given network. Moreover, these systems are managed by human beings and so they are highly prone to human error.

Development of an online IDS is a challenging task that would classify network traffic data either ′*anomaly*′ or ′*normal*′ by efficiently analyzing the dataset.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 509–518, 2012.

The proposed system overcomes the limitations of existing classifiers [3, 4]. In the paper sampled training data set has been generated with as much variation as possible using probabilities proportional to size without replacement (PPSWR)[5] to overcome the data overload problem. The objective of the paper is to maximize classification accuracy[3] by building rule based lazy classifier while dealing with the problem of 'false positive' and 'false negative' [2, 3].The proposed classifier can adapt itself with the changing network environment and able to detect network traffic to cope up with variants of common vulnerable security attacks.

2 Proposed Work

The proposed work consists of several procedures to generate the rule set for dynamic classification of the intrusion domain data sets. To develop the classifier a part of the well-known KDDCUP'99 data set [6] has been used for modeling and testing the intrusion detection system[2]. First the data set is discretized by applying cut [7] operation and then intervals are generated considering two successive cut points. As a next step, the intervals are encoded by corresponding centre-spread value [8].Contribution of each interval to classifying the data is evaluated by measuring biasness of an interval to predict the decision class label. Finally, by applying genetic algorithm (GA) [9] best matching rules are learnt, using which maximum classification accuracy is obtained.

2.1 Pre-processing

Network traffic data is preprocessed to obtain discrete attribute values from continuous ones. Using discrete attributes, redundant and less important attributes are removed by applying feature selection algorithm, namely CfsSubsetEval [10] and RankSearch [11] which effectively selects significant attributes and reduces complexity of the system. Instead of all attributes, henceforth only reduced attribute set is used for further processing of data.

2.2 Generation of Intervals

The method generates intervals for each conditional attribute from the set of discrete values as described below.

(i) The set of distinct discrete values for each conditional attribute are sorted in ascending order.
(ii) Each pair of successive values is considered as an interval including the lower value and excluding the upper value.

Example: The set of cut points(S) of a conditional attribute in ascending order is {1, 1.5, 2.5, 4.5, 5.5, 9.5, 10.5, 15.5, 19}

Respective intervals are listed in figure 1.

[1, 1.5)	Interval # 1
[1.5, 2.5)	Interval # 2
[2.5, 4.5)	Interval # 3
[4.5, 5.5)	Interval # 4
[5.5, 9.5)	Interval # 5
[9.5, 10)	Interval # 6
[10.5, 15.5)	Interval # 7
[15.5, 19)	Interval # 8

Fig. 1. Intervals corresponding to the cut-points in set (S)

Intervals	*Centre*	*Spread*
[1, 1.5)	1.25	0.25
[1.5, 2.5)	2	0.5
[4.5, 5.5)	5	0.5
[9.5, 10.5)	10	0.5
[10.5, 15.5)	13	2.5
[16.5, 19)	17.75	1.25

Fig. 2. Encoding of Intervals using Centre-Spread Technique

The next step is to encode these intervals to represent them using discrete value which is used for discretization of the incoming test data.

Centre-Spread encoding technique [8] has been applied in the paper for encoding the intervals. In Centre-Spread encoding method, an interval is denoted as [*lw*,*up*] and encoded as the mid-point of the interval and the span of interval from its mid-point to either side of its end-points. The mid-point of the interval represents discretized value of any continuous value belonging to that interval and the span gives the limit of the continuous values on either side of the mid-point. The intervals of figure 1 are mapped according to the format as given in figure 2.

2.3 Evaluation of Contribution

Once the intervals are generated, as a next step its contribution towards classifying the data is calculated considering training data only.

Contribution of an interval *is calculated using equation* (1) *which refers to a measure based on which one can infer how strongly biased an interval is in predicting a particular decision class.*

$$\boldsymbol{contribution\_interval\_dec\_normal} = \frac{(\boldsymbol{dec_{normal}} \times \mathbf{100})}{(\boldsymbol{dec_{normal}} + \boldsymbol{dec_{anomaly}})} \quad (1)$$

Where dec_{normal}= number of data objects occurring in that interval having decision *normal.*

And dec_{anomaly}= number of data objects occurring in that interval having decision *anomaly*.

Similarly,

$$\boldsymbol{Contribution\_interval\_dec\_anomaly} = \frac{(\boldsymbol{dec\_anomaly} \times \mathbf{100})}{(\boldsymbol{dec\_normal} + \boldsymbol{dec\_anomaly})} \quad (2)$$

Deviation of contribution of a concerned interval

$$= \begin{cases} \text{(Contribution of the interval in predicting decision as "normal" - Contribution of the interval in predicting decision as "anomaly")} & \text{; when Contribution of the interval predicting decision as "normal" is greater than or equals to the contribution of the interval predicting decision as "anomaly".} \\ -\text{(Contribution of the interval in predicting decision as "anomaly")} & \text{; Otherwise} \end{cases} \quad (3)$$

Therefore, for predicting *normal* class label, the deviation is positive while for *anomaly* it is negative with an assumption that the population of data objects having decision *normal* is more than the decision as *anomaly*. So if two intervals having same contribution value while one predicting decision as *normal* and other predicting decision as *anomaly*, then the interval predicting decision as "*anomaly*" have a greater effect. This deviation of contribution of a concerned interval is nothing but the relative measure of strength of that interval on prediction of the decision value as *normal* or *anomaly*. More the value positive, more biased the interval is towards predicting decision as *normal*. Similarly, more the negative value of this measure, more is the interval biased towards predicting decision as *anomaly*.

2.4 Generation of Initial Rule Set

Once the intervals of all conditional attributes in the training dataset and their corresponding contribution towards the prediction of the decision of a particular class are generated, the next task is to generate the initial rule set of the classifier. In the subsequent steps GA [9] is applied to adapt the new test data set.

Algorithm. *Initial Rule Set Generation*

Step 1: The conditional attribute (A_{max}) with maximum number of intervals is obtained.

Step 2: For each interval (I) in A_{max} create a rule with
(i) Attribute A_{max} consists of interval I.
(ii)Rest of the attributes consist interval having closest contribution deviation value to I.
(iii) The intervals which are selected removed from the corresponding attributes. If an attribute contains no more intervals, then all the intervals are added back.

Step 3: Repeat Step 2 until A_{max} does not contain any more intervals.

2.5 Generation of the Best Matching Rule

For each incoming test data, the rule which matches with the maximum number of attribute values is obtained using the concepts of genetic algorithm.

Algorithm. *best matched rule generation*

Step 1: Copy the original rule set into a temporary rule set (T).
Step 2: Select a rule (R1) from T having maximum number of attribute matching with that of the test data set.
Step 3: Remove R1 from T.
Step 4: Select a rule (R2) at random from T.
Step 5: Create a new rule with attributes of R1 and R2, matches with that of the incoming data.
Step 6: Assign the newly created rule as R1. Remove R2 from T.
Step 7: Repeat Step 4 to Step 6 until T is empty.

The resultant rule has maximum number of attribute value, matching with that of the test data.

An instance rule set is shown in figure 3.

Rule no.	Attr-1	Attr-2	Attr-3	Attr-4	Attr-5	Attr-6	Attr-7	Attr-8	Attr-9	Attr-10
1	1.5	2	2	3.5	0.5	1	2	0	2	3.5
2	1.5	0	3	4	1.5	0.5	3	0.5	3	2.5
3	1	0.5	2.5	3	2	0.5	4	1.5	2.5	2
4	0	2	3	3.5	1.5	1	0	0.5	2.5	3.5
5	2	0.5	3.5	2.5	1.5	0	1	0.5	1	3

Fig. 3. Sample population of the Initial Rule set

The incoming test data is given in figure 4.

Attr-1	Attr-2	Attr-3	Attr-4	Attr-5	Attr-6	Attr-7	Attr-8	Attr-9	Attr-10
1.5	0	2	3	2	0.5	0	0	1.5	3.5

Fig. 4. Instance of Incoming Test Data

After applying the *best matched rule generation* algorithm we get the best matching rule, given in figure 5.

Attr-1	Attr-2	Attr-3	Attr-4	Attr-5	Attr-6	Attr-7	Attr-8	Attr-9	Attr-10
1.5	0	2	3	2	0.5	0	0	3	3.5

Fig. 5. Best Matching Rule

2.6 Predicting Decision

For predicting decision of an incoming test data, the proposed classifier employs the decision making procedure as described below.

Step 1: For each attribute in the *reduct* set, the interval which matches with the test data, scale the absolute value of contribution deviation of the interval (***abs***) from (0 ~ 100) to (0 ~ 20).

Step 2: The effect of this interval towards prediction of the decision is calculated using equation (4).

$$\boldsymbol{effect(e) = 2^{(abs)}} \tag{4}$$

Step 3: The prediction is made based on the net effect of all the attributes matching the test data, calculated using equation (5).

$$\begin{aligned} &\boldsymbol{Net\ Effect} = \\ &\boldsymbol{Net\ Effect} + \boldsymbol{e};\ \textit{if contribution deviation of the interval is positive} \\ &\boldsymbol{Net\ Effect} = \\ &\boldsymbol{Net\ Effect} - \boldsymbol{e};\ \textit{if contribution deviation of the interval is negative} \end{aligned} \tag{5}$$

Step 4: If Net Effect is positive or zero then the decision predicted is *normal* else *anomaly*.

Note: Here an interval having modulus of contribution deviation close to 100 have an effect which is exponentially greater than those having a value close to 0.

3 Experimental Results and Comparisons

3.1 Comparisons with Original KDD Data Set

Comparison with respect to classification accuracy of the common existing classifiers of different categories with the proposed classifier is demonstrated here. We have taken 10000 data objects from NSL-KDD Data set [12, 13] for network-based intrusion detection and used 10 fold cross validation technique [3] for measuring classification accuracy [3] of the different classifiers. We have used the modules provided by the WEKA tool [14, 15] to run different classifiers. The accuracy of the proposed classifier is **94.9**% on an average, as shown in figure 6.

The proposed classifier is primarily a predictive model and falls under lazy classifiers, comparison with different lazy and probability based classifiers are shown in figure 7.

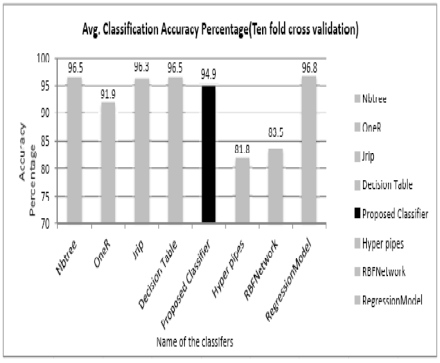


Fig. 6. Comparison of the Proposed Classifier with different types of well-known Classifiers

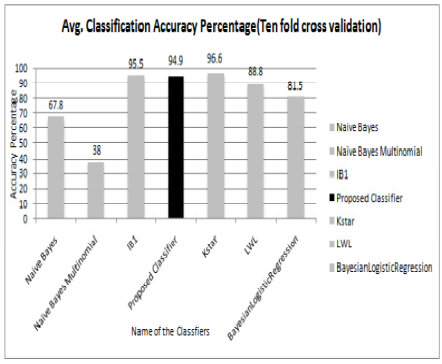


Fig. 7. Comparisons of the proposed classifier with lazy and probabilistic classifiers

By analyzing figure 6 and 7, one can observe that the proposed classifier provides slightly less classification accuracy from that of the well-known classifiers (like Nbtree, Jrip, Decision table and Regression Model as shown in figure 6), when applied on original NSL-KDD dataset (considering both training and test data set taken from the original one). This is due to the fact that we have not considered the concept of correlation between different conditional attributes in building the classifier while all of the well-known classifiers (giving better performance than our one) consider the correlation between different attributes during learning. However, we have tried to ensure that the proposed classifier gives better classification accuracy when it runs on a test dataset which is a variant of the original one and has less correlation.

3.2 Comparisons Using Evolutionary Test Dataset

The classification accuracy of different classifiers are obtained considering data set created using evolutionary method. To generate the data set concepts of crossover and mutation operations of genetic algorithm are applied. This new data set is a variant of the original network traffic data set and represents modified forms of intrusions or attacks that can be encountered by the IDS.

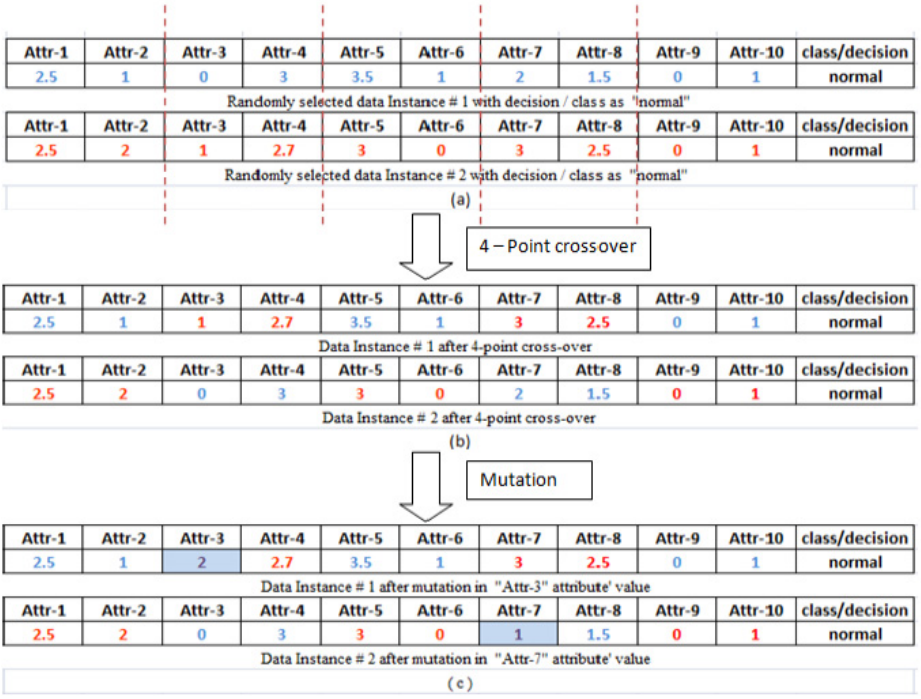


Fig. 8. Generation of evolutionary data instances

For creation of each data instance in this new data set, two data instances are sampled from the original data set having same class label(figure 8(a)) and performed 4 point cross over to generate the new instances, shown in figure 8(b). This new instance has the same class label as that of its parent chromosome. After performing crossover, mutation is applied by replacing the value of any randomly chosen conditional attribute by a value which lies in the domain of that conditional attribute, represented in figure 8(c).

The new data set maintains the ratio of number of *normal* instances to the number of *anomaly* instances same as that of the original data set. Here the correlation between different attributes in newly generated evolutionary data set has been shown in the figure 9(b),different from the correlation values in the original data set as shown in the figure 9(a). In figure9, we have shown top 4 attributes of rank as determined by rank search in both (a) and (b).

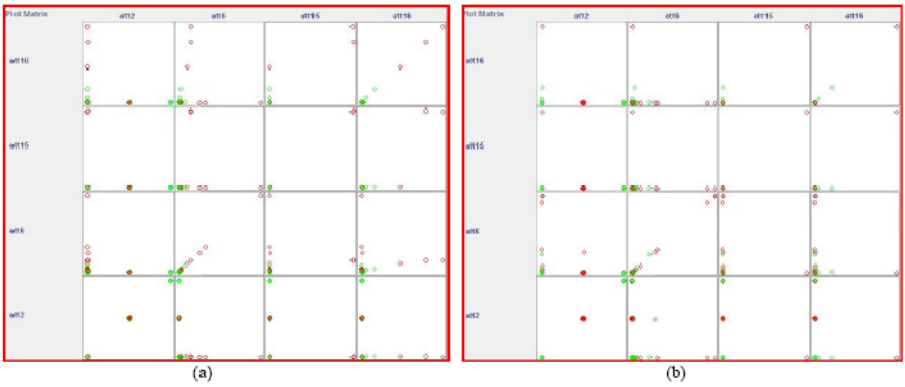


Fig. 9. (a) Correlation between different conditional attributes in original KDD data set; (b) Correlation between different conditional attributes in evolutionary (test) data set obtained from original NSL-KDD data set by crossover and mutation

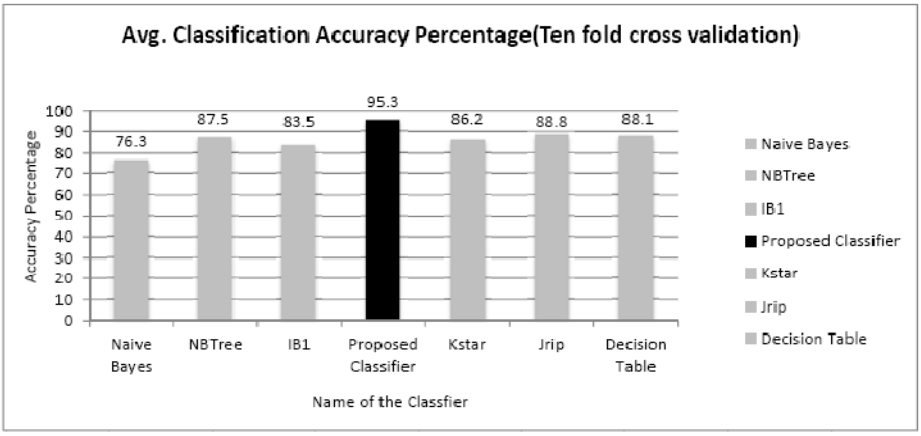


Fig. 10. Comparisons while detecting variants of common network traffic data patterns

Experiments with new test data set and comparing classification accuracies of different well-known efficient classifiers and our method, following results are obtained, shown in the figure10. It has been concluded that the proposed classifier outperforms all others giving the best average classification accuracy as **95.3**% while detecting variants of common network traffic data patterns.

4 Conclusions

In this paper, we have built a classifier which does not take correlation into account while predicting decisions of the incoming test data. The proposed classifier outperforms all the classifiers while detecting test data which are variants of the original data set representing modified form of common network security attacks. Since these test data patterns have less correlation than the original training data set, rule based and decision tree based classifiers gives lower classification accuracy compared to that of our proposed classifier obtained by 10 fold cross validation. Moreover, the proposed classifier shows impressive performance while detecting normal behavioral traffic data patterns compared to other efficient existing classifiers as discussed in the previous section. Therefore, our proposed classifier can efficiently meet the needs for designing an up-to-date intrusion detection system with providing solutions to the limitations of existing ones operating in modern dynamic network environment.

References

1. SANS Institute InfoSec Reading Room site: Understanding Intrusion Detection Systems
2. Scarfone, K., Mell, P.: Guide to Intrusion Detection and Prevention Systems (IDPS), Recommendations of the National Institute of Standards and Technology
3. Han, Kamber, M.: Data Mining: Concepts And Techniques. MorganKaufmann Publishers, San Francisco (2001)
4. Wu, X., Kumar, V., Ross Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Yu, P.S., Zhou, Z.-H., Steinbach, M., Hand, D.J., Steinberg, D.: Top 10 algorithms in data mining. Knowl. Inf. Syst. 14, 1–37 (2008), doi:10.1007/s10115-007-0114-2
5. Rosen, B.: On sampling with probability proportional to size. Journal of Statistical Planning and Inference 62, 159–191 (1997)
6. KDD Cup (1999), `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`
7. Komorowski, J., Polkowski, L., Skowron, A.: Rough Set: A tutorial
8. Wyatt, D.: Applying the XCS Learning Classifier System to Continuous-Valued Data-mining Problems, Technical Report UWELCSG05-001 (September 2004)
9. Goldberg, D.E.: Genetic algorithms in search, optimization and machine earning. Addison-Wesley, Reading (1989)
10. Selvakuberan, K., Indradevi, M., Rajaram, R.: Combined Feature Selection and classification–A novel approach for thecategorization of web pages
11. Data Mining Algorithm In R/Dimensionality Reduction/Feature Selection, `http://www.wikipedia.org/`

12. Nsl-kdd data set for network-based intrusion detection systems (March 2009), http://nsl.cs.unb.ca/NSL-KDD/
13. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A Detailed Analysis of the KDD CUP 99 Data Set
14. Weka 3: Data Mining Software in Java, http://www.cs.waikato.ac.nz/ml/weka/
15. Weka User Manual, http://www.gtbit.org/downloads/dwdmsem6/dwdmsem6lman.pdf, http://kent.dl.sourceforge.net/project/weka/documentation/3.6.x/WekaManual-3-6-2.pdf

2DOF PID Controller Tuning for Unstable Systems Using Bacterial Foraging Algorithm

K. Latha[1] and V. Rajinikanth[2]

[1] Department of Instrumentation Engineering, M.I.T. Campus,
Anna University, Chennai, Tamilnadu, India
[2] Department of Electronics and Instrumentation, St. Joseph's College of Engineering,
Jeppiaar Nagar, Chennai, Tamilnadu, India

Abstract. In this paper, a method to tune the two Degree Of Freedom PID controller for a class of time delayed unstable systems using Bacterial Foraging Optimization algorithm is proposed. In this algorithm, limits are assigned for the algorithm and search parameters to minimize the convergence time during the optimization exploration. The problem considered in this study is to aptly tune the controller in order to enhance the overall closed loop performance of the time delayed unstable process. The BFO algorithm is focused to search the best possible controller parameter values such as 'K_p, K_i, K_d' and controller weighting parameters 'α, β' by minimizing the multiple objective function which monitor the optimization search. A comparative study is presented between the feed forward and the feedback 2DOF PID structures. The robustness of the BFO tuned 2DOF PID controller is tested by introducing model uncertainty in the process model parameters. The simulation results evident that, the proposed procedure helps to accomplish enhanced system performance such as smooth reference tracking, satisfactory disturbance rejection, and error minimization for a class of unstable systems.

Keywords: 2DOF PID controller, Bacterial foraging, Model uncertainty, Performance measure, Unstable system.

1 Introduction

In the control literature, despite the significant developments in advanced process control schemes, Proportional + Integral + Derivative (PID) controllers are still widely used in industrial control system where reference tracking and disturbance rejection are the major task. PID and the modified structured PID controllers are widely used in industries because of their structural simplicity, reputation and easy implementation. In the literature, there are several papers dealing with the tuning and implementation of a single Degree Of Freedom (1DOF) PID controller for stable and unstable systems. For stable systems, 1DOF PID structure provides a viable result for both the reference tracking and disturbance rejection. But, for unstable systems, PID controller can capably work either for efficient reference tracking or disturbance rejection operation.

Most of the industrial process loops are represented by an equivalent higher order modeling equations because of its nonlinear nature. The nonlinear processes can be

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 519–527, 2012.

efficiently modeled as linear processes (stable / unstable) based on the operating regions. These linear models are then considered for the model based controller tuning operations. Sometimes, these higher order process loops cannot provide the expected result, when it is controlled using a lower order (1DOF) PID controller structure. Since it is always essential to use a higher order PID structures such as 2DOF.

2DOF PID controller was widely addressed by the researchers for stable and unstable process models [1-5]. The classical PID tuning methods proposed by most of the researchers for particular process model will not provide the fitting response for other process models. The classical PID tuning also requires computations in order to get the best controller parameters. Due to these reasons, in recent years, soft computing based PID controller tuning is greatly attracted the researchers.

In this paper, Bacteria Foraging Optimization (BFO) based 2DOF PID controller tuning is proposed for a class of time delayed unstable process models. BFO is a nature inspired heuristic search technique, introduced by Kevin M. Passino [6]. The previous research has reported the superiority of the BFO based search for finding the optimum solution for a class of engineering problems [7-9,15,16]. It is based on mimicking the foraging activities of Escherichia coli (E.coli) bacteria. During the optimization exploration, a collection of artificial bacteria cooperates to find the best possible solutions in the '*D*' dimensional search space. We propose a parameter normalized Bacteria Foraging Optimization (BFO) algorithm to discover the optimized PID controller parameters such as K_p, K_i, K_d and controller weighting parameters 'α, β' for a class of unstable process models by maintaining the guaranteed accuracy.

2 Bacteria Foraging Algorithm

Bacteria Foraging Optimization (BFO) algorithm is a new division of metaheuristic algorithm. It is a population based optimization technique developed by inspiring the foraging manners of E.coli bacteria. The basic operation of BFO algorithm has four key steps namely chemo-taxis, swarming, reproduction, and elimination-dispersal. A detailed description of the above procedures could be found in [10-13].

The parameters of the classical BFO algorithms are defined below:

D = the dimension of search space, N = the total number of E.Coli bacteria, N_c= total number of chemotaxis steps, N_s = swim length during the search, N_{re} = total number of reproduction steps, N_{ed} = total number of elimination – dispersal events, N_r = number of reproduced bacteria, P_{ed} = the probability that each bacterium will be eliminated / dispersed, n = the run length.

In the literature, there is no apparent guide line to allot the parameters of the BFO algorithm. In this work, we assigned the boundaries for algorithm parameters in order to minimize the complexity.

The algorithm parameters are assigned as follows [17]:

- The total number of E.Coli bacteria (N) = 22
- The total number of chemotactic steps (N_c) = $\frac{N}{2}$

- Swim length (N_s) = number of reproduction steps (N_{re}) $\approx \frac{N}{3}$
- The number of elimination – dispersal events (N_{ed}) $\approx \frac{N}{4}$
- The total number of bacterial reproduction (N_r) = $\frac{N}{2}$
- The probability of the bacterial elimination/dispersal (P_{ed}) = $\left(\frac{N_{ed}}{N + N_r}\right)$
- Total number of iterations during the search = N^2
- Swarming parameters can be assigned as follows:

$$d_{attractant} = W_{attractant} = \frac{N_s}{N}; \quad \text{and} \quad h_{repellant} = W_{repellent} = \frac{N_c}{N}$$

- Initial positions for the bacterium (bacterium 1 to N) is assigned as follows

*PI(value1: Dim1, Dim2, … Dim D) = SB for value1 * rand (N, Dim1)*

*PI(value2: Dim1, Dim2, … Dim D) = SB for value2 * rand (N, Dim2)*

. . . . . .

*PI(value D:Dim1, Dim2, …Dim D) = SB for valueD * rand (N, Dim5)*

Where, PI = Performance index which guides the algorithm, SB = Search boundary for the 'D' dimensional parameters , Dim= number of values to be optimized = total no of search dimension rand = random number (0 < rand <1).

The chief benefit of the BFO is the number of parameters to be assigned are only the values of N, D, PI, and SB.

3 2DOF PID Controller

The PID and modified structure PID controllers are used to shape the steady state as well as the transient response of the process control system. For unstable systems, the 1DOF controller fails to provide a smooth reference tracking performance due to the occurrence of proportional and derivative kick [14]. In order to reduce the effects of proportional and derivative kick and also to improve the time response characteristics, it is essential to consider a 2DOF PID structure. A detailed study on various 2DOF structures was clearly presented by Araki and Taguchi [4].

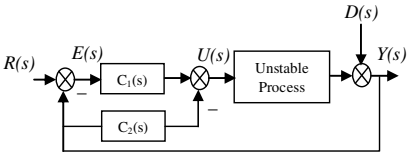


Fig. 1. Feedback structure

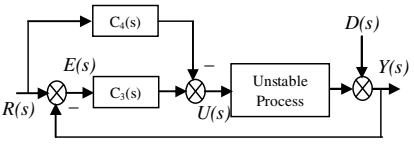


Fig. 2. Feed forward structure

In this paper, an attempt has been made with the Feedback (FB) and the Feed Forward (FF) 2DOF PID structures. Fig. 1 depicts a FB type controller structure with a PD controller in the inner loop and a PID controller in the outer loop. In this structure, the PID controller responds on error signal '*e(t)*' and the PD controller works on the process output '*y(t)*'. Since the controller '$C_2(s)$' is free from proportional and derivative kick effect.

The controller values for this structure are presented in Eqn. 1 and 2.

$$C_1(s) = K_p\left((1-\alpha) + \frac{1}{\tau_i s} + (1-\beta)\tau_d D_f(s)\right) = K_p(1-\alpha) + K_i + (1-\beta)K_d D_f(s) \quad (1)$$

$$C_2(s) = K_p\left(\alpha + \beta\,\tau_d D_f(s)\right) = K_p\,\alpha + \beta\,K_d D_f(s) \quad (2)$$

Where: $K_i = K_p / \tau_i$; $K_d = K_p\ \tau_d$ (τ_i = integral time constant, τ_d = derivative time constan); α and β are controller weighting parameters (0 to 1); $D_f(s)$ is the derivative filter term given by's */ (1+Ns)', (N*= filter constant = 10)

Fig. 2 shows a FF type controller structure with a PD controller in the feed forward loop and a PID controller in the closed loop. The PID controller responds on error signal '*e(t)*' and the PD controller works on the reference input '*r(t)*'.

The controller values for this structure are presented in Eqn. 3 and 4.

$$C_3(s) = K_p\left(1 + \frac{1}{\tau_i s} + \tau_d D_f(s)\right) = K_p + K_i + K_d D_f(s) \quad (3)$$

$$C_4(s) = K_p\left(\alpha + \beta\,\tau_d D_f(s)\right) = K_p\,\alpha + \beta\,K_d D_f(s) \quad (4)$$

3.1 Controller Tuning

The controller tuning process is employed to find the best possible values for K_p, K_i, K_d, α, and β. In order to achieve superior accuracy during the optimization search, it is necessary to assign appropriate Performance Index (PI) which guides the BFO algorithm.

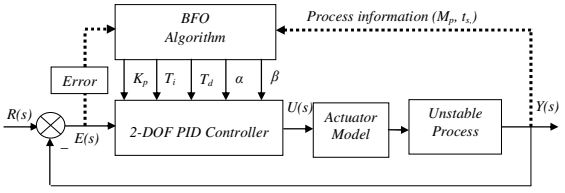


Fig. 3. BFO tuned 2DOF PID controller

In this work, the following PI (Eqn.5) with *ISE*, *IAE*, M_p, and t_s are considered.

$$J_{min}(K_p, K_i, K_d, \alpha, \beta) = (w_1.\ ISE) + (w_2.\ ITSE) + (w_3.\ M_p) + (w_4.\ t_s)) \quad (5)$$

Where w_1, w_2 ... w_4 are weighting functions used to set the priority of the PI parameters and the value of 'w' varies from 0 to 10. The criterion 'J_{min}' guides the algorithm towards the appropriate values of 2DOF PID controller parameters.

Prior to the optimization search, it is essential to assign the parameters for BFO and PI. In this study, the following values are assigned:

- Dimension of the search space (D) = 5 (ie. K_p, K_i, K_d ,α, β)
- The total number of E.Coli bacteria = 22
- Boundaries for the three dimensional search space is assigned as (If the numerator of the transfer functions has a positive sign (ie. +) the lower boundary is assigned as zero. Else the lower boundary will have a –ve value)

$$\begin{aligned} &Value\ 1 = 0\% < K_p < +50\%\ (ie.\ 0 < K_p < 5.0)\\ &Value\ 2 = 0\% < K_i < +25\%\ (ie.\ 0 < K_i < 2.5)\\ &Value\ 3 = 0\% < K_d < +30\%\ (ie.\ 0 < K_d < 3.0)\\ &Value\ 4 = 0\% < \alpha < +10\%\ (ie.\ \ 0 < \alpha < 1)\\ &Value\ 5 = 0\% < \beta < +10\%\ (ie.\ \ 0 < \beta < 1) \end{aligned}$$

- The weighting functions are assigned as $w_1=w_2=w_3= w4=10$
- Simulation time is 100 sec
- The overshoot (M_p) range is selected as <*50%* of the reference signal.
- The 't_s' is preferred as < *50%* of the maximum simulation time.
- The reference signal is considered as R(s) = 1.

4 Results and Discussions

The following simulation study demonstrates the proficiency of the BFO tuned 2DOF PID controller on a class of unstable process models.

Example 1: Let us consider the following First Order Plus Time Delayed (FOPTD) unstable process studied by Padhy and Majhi [5],

$$G(s) = \frac{1e^{-0.8s}}{s-1} = \frac{K}{\tau s-1}e^{-\theta s} \tag{6}$$

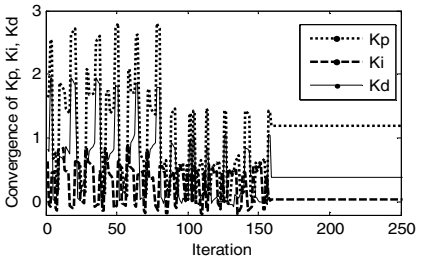


Fig. 4. Convergence of K_p, K_i, K_d

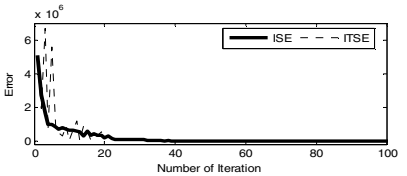


Fig. 5. Convergence of ISE, ITSE

The FOPDT system has a 'θ /τ' ratio of 0.8, hence it is very difficult to stabilize the above model with a classical PID controller. The BFO based 2DOF PID controller tuning is proposed for the above process model as in Fig 3. Initially the feedback type 2DOF structure is considered in the BFO based search. Five trials are executed during the search and the finest value among the trials is chosen as the best possible controller value. The convergence of the BFO algorithm towards the global optimal solution for the controller parameters and the errors are graphically represented in Fig 4 and Fig 5 respectively. The chosen search value is converging at 159th iteration and the optimized controller

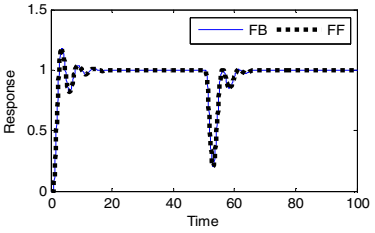


Fig. 6. Closed loop response for Ex. 1

parameters and the corresponding error values are presented in Table 1. The controller values such as K_p, K_i, K_d, α, and β obtained with the FB structure is tested on the FF structure, to ensure the reference tracking and the disturbance rejection performances. Fig 6 depicts the reference tracking and disturbance rejection case for Ex 1 with FB and FF structures. From the above result, it is confirmed that, both the 2DOF structures are similar and provides identical performance. The supply disturbance rejection feature of the controllers are tested by applying a disturbance signal of 0.25 (25% of the setpoint) at 50sec as depicted in Fig 6. The effect of the disturbance is completely eliminated at 60 sec. The error values for FB and FF structures are the same and it is presented in Table 1.

Example 2: Let us take an unstable second order process with one unstable pole as stated in Eqn 7 [2].

$$G_p(s) = \frac{1e^{-s}}{(2s-1)(0.5s+1)} \tag{7}$$

As discussed above, the controller tuning is initially attempted with the FB structure and the optimized controller values are considered for both the FB and FF structure.

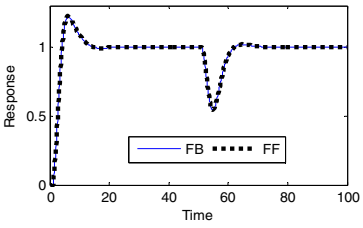


Fig. 7. Servo and regulatory response for Ex.2

Fig 7 shows the closed loop response of the BFO tuned 2DOF controller for FB and FF structures. The reference tracking response results in overshoot of 0.28, and a settling time of 18 sec. The supply disturbance rejection performance of the controller is tested by applying a disturbance signal of 0.25 (25% of R(s)) at 50sec as depicted in Fig 6. The effect of the disturbance is completely eliminated at 65 sec. The other important parameters such as ISE, IAE, ITSE, and ITAE values are presented in Table 1.

Example 3: This example considers an integrating process with one unstable pole as existing in [1].

$$G_p(s) = \frac{1e^{-0.2s}}{s(s-1)} \tag{8}$$

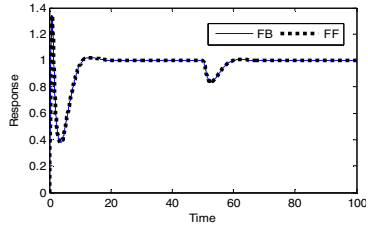


Fig. 8. Closed loop response for Ex. 3

For the above process, Liu et al. [1] proposed a traditionally designed 2DOF control scheme. Due to the presence of the integral value, this process may generate an unnecessary over shoot, when it is controlled with the basic PID controller.

The integral term makes the system more complex compared to other process models.

In this work, the controller tuning procedure is executed as discussed in Example 1. The iteration number and the final values of controller parameters are presented in Table 1. Fig 8 depicts the reference tracking and supply disturbance rejection performance of the process model. During the reference tracking, the proposed controller shows an overshoot of 32%. The supply disturbance rejection performance for this process model shows a satisfactory result with a reduced error values.

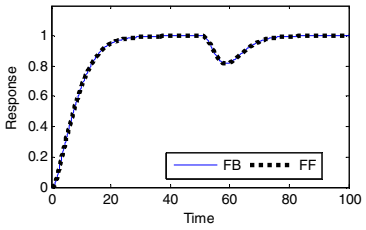


Fig. 9. Servo and regulatory response for Ex.4

Example 4: Let us take a third order process with one unstable pole as given in [2],

$$G_p(s) = \frac{1e^{-0.5s}}{(5s-1)(0.5s+1)(2s+1)} \quad (9)$$

Initially the controller parameter optimization is proposed with the FB structure. The performance of the FB and FF structures are tested on the process model and the results are recorded in Table 1. Later, the robustness of the BFO tuned 2DOF PID controller is tested by assuming perturbation in the gain 'K' and delay 'θ'.

Fig 10 shows the reference tracking performance of Ex 4 for uncertainty in K ± 25% (0.75 < K < 1.25) and θ ± 15% (0.35 < θ < 0.65).

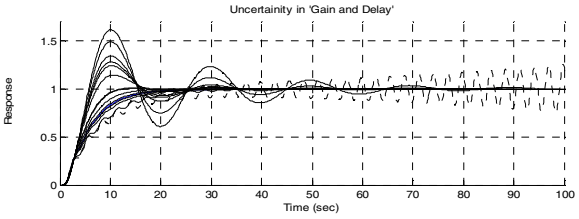


Fig. 10. Closed loop step response of Ex.4 with perturbed model parameters

When the model perturbation is within the specified limit, the process offers a satisfactory result. The decrease in 'K' and/or increase in 'θ' will initiate unwanted overshoot and oscillations as depicted in Fig 10.

Table 1. Optimized controller parameters and its performance measure

	Iteration	Controller parameters					Performance measure			
		K_p	K_i	K_d	α	β	ISE	IAE	ITSE	ITAE
Ex.1	159	1.416	0.097	0.291	0.838	0.910	2.882	5.115	71.12	147.0
Ex.2	106	1.572	0.111	0.998	0.751	0.844	3.155	6.256	46.81	150.6
Ex.3	227	1.134	0.331	2.883	0.903	0.870	2.424	5.081	11.88	61.17
Ex.4	113	2.294	0.112	3.608	0.864	0.818	5.807	11.00	38.82	194.4

From the above study, it is noted that, even though there is a structural mismatch, the FB and FF 2DOF PID offers similar result on unstable process models.

5 Conclusion

In this work, we discussed a Bacterial Foraging algorithm tuned 2DOF PID controller for a class of unstable process models. It is a five dimensional search, in order to minimize the algorithm complexity guidelines are provided to assign the BFO algorithm parameters and controller parameters. A comparative study is executed between Feedback and Feed Forward controller structures. From the result, it is confirmed that both the 2DOF structures are similar and provides an identical performance. Reference tracking and supply disturbance rejection procedures are performed on the process models. The proposed method shows a robust performance for the system with the model perturbation. The results show that, the BFO tuned controller supports smooth reference tracking, satisfactory disturbance rejection, and error minimization for the unstable systems considered in this study.

References

1. Liu, T., Zhang, W., Gu, D.: Analytical design of two-degree-of-freedom control scheme for open-loop unstable processes with time delay. Journal of Process Control 15, 559–572 (2005)
2. Chen, C.-C., Huang, H.-P., Liaw, H.-J.: Set-Point Weighted PID Controller Tuning for Time-Delayed Unstable Processes. Ind. Eng. Chem. Res. 47(18), 6983–6990 (2008)
3. Vijayan, V., Panda, R.C.: Design of a simple setpoint filter for minimizing overshoot for low order processes. ISA Trans. 51(4), 514–521 (2012), doi:10.1016/j.isatra.2011.10.006
4. Araki, M., Taguchi, H.: Two-Degree-of-Freedom PID Controllers. International Journal of Control, Automation, and Systems 1(4), 401–411 (2003)
5. Padhy, P.K., Majhi, S.: Relay based PI-PD design for stable and unstable FOPDT processes. Computers and Chemical Engineering 30, 790–796 (2006)
6. Passino, K.M.: Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Systems Magazine, 52–67 (2002)
7. Dasgupta, S., Das, S., Biswas, A., Abraham, A.: Automatic circle detection on digital images with an adaptive bacterial foraging algorithm. Soft. Comput. 14(11), 1151–1164 (2010)
8. Rajinikanth, V., Latha, K.: Bacterial Foraging Optimization Algorithm based PID controller tuning for Time Delayed Unstable System. The Mediterranean Journal of Measurement and Control 7(1), 197–203 (2011)
9. Rajinikanth, V., Latha, K.: I-PD Controller Tuning for Unstable System Using Bacterial Foraging Algorithm: A Study Based on Various Error Criterion. Applied Computational Intelligence and Soft Computing (2012), doi:10.1155/2012/329389
10. Chen, H., Zhu, Y., Hu, K.: Cooperative Bacterial Foraging Optimization. Discrete Dynamics in Nature and Society (2009), doi:10.1155/2009/815247
11. Dasgupta, S., Das, S., Abraham, A., Biswas, A.: Adaptive Computational Chemotaxis in Bacterial Foraging Optimization: An Analysis. IEEE Trans. Evolutionary Computation 13(4), 919–941 (2009)
12. Das, S., Biswas, A., Dasgupta, S., Abraham, A.: Bacterial Foraging Optimization Algorithm: Theoretical Foundations, Analysis, and Applications. Foundations of Computational Intelligence (3), 23–55 (2009)

13. Biswas, A., Das, S., Abraham, A., Dasgupta, S.: Stability analysis of the reproduction operator in bacterial foraging optimization. Theor. Comput. Sci. 411(21), 2127–2139 (2010)
14. Johnson, M.A., Moradi, M.H.: PID Control: New Identification and Design Methods. Springer-Verlag London Limited (2005)
15. Panigrahi, B.K., Ravikumar Pandi, V.: Congestion management using adaptive bacterial foraging algorithm. International Journal on Energy Conversion and Management 50, 1202–1209 (2009)
16. Pandi, V.R., Biswas, A., Dasgupta, S., Panigrahi, B.K.: A hybrid bacterial foraging and differential evolution algorithm for congestion management. Euro. Trans. Electr. Power 20, 862–871 (2010), doi:10.1002/etep.368
17. Rajinikanth, V., Latha, K.: Controller Parameter Optimization for Nonlinear Systems Using Enhanced Bacteria Foraging Algorithm. Applied Computational Intelligence and Soft Computing (2012), doi:10.1155/2012/214264

Generation of Sufficient Cut Points to Discretize Network Traffic Data Sets

Sahisnu Mazumder[1], Tuhin Sharma[1], Rahul Mitra[1],
Nandita Sengupta[2], and Jaya Sil[1]

[1] Dept. of Computer Science and Technology,
Bengal Engineering and Science University,
Shibpur, Howrah -711 103, West Bengal, India
[2] University College of Bahrain, Bahrain
{sahisnumazumder,tuhinsharma121,rahulm9999}@gmail.com,
ngupta@ucb.edu.bh, js@cs.becs.ac.in

Abstract. Classification accuracy and efficiency of an intrusion detection system (IDS) are largely affected by the discretization methods applied on continuous attributes. Cut generation is one of the methods of discretization and by applying variable number of cuts (in a partition) to the continuous attributes, different classification accuracy are obtained. In the paper to maximize accuracy of classifying network traffic data either 'normal' or 'anomaly', the proposed algorithm determines the set of cut points for each of the continuous attributes. After generation of appropriate and necessary cut points, they are mapped into corresponding intervals following centre-spread encoding technique. The learnt cut points are applied on the test data set for discretization to achieve maximum classification accuracy.

Keywords: discretization, cut generation, center-spread encoding, nsl-KDDCUP'99 network traffic data set classification accuracy.

1 Introduction

The process of partitioning continuous variables into categories is termed as discretization. Discretization is a potential time-consuming bottleneck, since the number of possible discretization is exponential in the number of interval threshold candidates within the domain. The goal of discretization is to bind the domain of all continuous conditional attributes into some finite set of values. The finite values help to observe the patterns of objects and associated decision class labels appearing in the discretized training data set. Many of the Machine Learning algorithms [1,2,3] produce better models by discretizing continuous attributes. For example, Naive Bayes classifier [4,5] requires probability estimations with the help of continuous attributes. But it is difficult to handle as they often take too many different values for a direct estimation of frequencies. To alleviate this problem, normal distribution of the continuous values can be assumed, but it is not always realistic. The same phenomenon leads rule extraction techniques to build poorer sets of rules. Decision

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 528–539, 2012.

Tree based algorithms [6,7,8] cannot handle continuous attribute directly rather nominal attributes. As a result, machine learning and statistical techniques are applied on the data sets to compose nominal variables. However, a very large proportion of real data sets include continuous variables i.e. variables measured at the interval or ratio level. One solution is to partition numeric variables into a number of sub-ranges [9] and treat each such sub-range as a category. In the paper, based on the observations, contribution of a given interval corresponding to a particular decision (normal or anomaly) has been measured. In addition, by maximizing the interdependence between class labels and attribute values, the paper aims at to develop an ideal discretization method with a secondary goal to minimize the number of intervals without significant loss of class-attribute mutual dependence.

Cut generation [10] is one of the widely used methods of discretization. By varying number of cuts, applied on continuous attributes classification accuracy is also varied. So, in order to maximize classification accuracy of the network traffic data, the classifier system should learn to determine the set of cut points for each of the continuous attributes. In the proposed classifier system, a heuristic based cut generation algorithm has been developed which generates a set of cut points that are sufficient to distinguish between a discernible pair of data objects.

The paper has been divided into four sections. Section 2 presents the proposed work while section 3 comprises of experimental results and comparisons with other discretization methods considering nsl-KDDCUP'99 network traffic data set [11,12]. Finally, conclusions are summarized in section 4.

2 Proposed Work

Discretization is performed prior to the learning process [13] by dividing the total task into three sub-modules. The first task is to find the number of discrete intervals, unlike other discretization algorithms where user must specify the number of intervals [14] or provide a heuristic rule [15]. The second task is to find the width or the boundaries of the intervals depending on the range of values of each continuous attribute. Finally, the attribute values from the continuous domain are mapped to the discrete domain.

2.1 Cut Points Generation

Heuristic based solution to determine the set of all necessary and sufficient cut points has been described below:-

Algorithm Cut Generation

Step 1: Consider, k^{th} conditional attribute values in a decision system and arrange them in ascending order.

Step 2: For each particular value, group the objects based on the decision attribute value (d) either ***normal*** (say, d=1) or ***anomaly*** (say, d=2).

Step 3: The ordered k^{th} conditional attribute values are marked as per following rule:

(i) Mark a value by "tick" if it corresponds to an object with decision attribute value *normal* (d=1).

(ii) Mark a value by "Encircle" if it corresponds to an object with decision attribute value *anomaly* (*d*=2).

Step 4: Scan the marked k^{th} attribute values and set the cut points between any two successive values (say, attr-*k*_value-1 and attr-*k*_value-2) accordingly:

(i) If attr-*k*_value-1 is marked as "encircled" (or "ticked") and attr-*k*_value-2 is marked as "ticked" (or "encircled"), select a cut point as the mid-point of the interval between attr-*k*_value-1 and attr-*k*_value-2.

(ii) If one of them (say, attr-*k*_value-1) is marked as both "encircled" and "ticked" (i.e. in both ways) and other one (say, attr-*k*_value-2) is marked as either "encircled" or "ticked" (i.e., only one of the two ways), select a cut point as the mid-point of the interval between attr-*k*_value-1 and attr-*k*_value-2, which are marked differently.

(iii) If attr-*k*_value-1 is marked as both "encircled" and "ticked" and attr-*k*_value-2 is also marked as both "encircled" and "ticked", select a cut point as the mid-point of the interval between attr-*k*_value-1 and attr-*k*_value-2.

Step 5: Do step 1 to step 4 for all conditional attributes.

The proposed method of generating a cut point between two successive attribute values is based on their marking either different way or in both ways. ***Therefore, the data objects having those values of the k^{th} attribute forms a discernible pairs belonging to that training data set.***

***Example 1*:** Consider the following continuous data set in decision system, given in the table 1. In this decision system, there are 3 conditional attributes (Attr-1, Attr-2, Attr-3) and one decision attribute (Decision) with two possible decision attribute values (Decision =1, representing *normal* and Decision =2, representing *anomaly*). There are 21 data objects or instances in the given continuous data set in table 1.

Table 1. A Continuous Decision System

Attr-1	Attr-2	Attr-3	Decision
13	1	1	1
0	2	2	1
0	2	2	2
7570	1	1	1
0	1	4	1
0	1	5	2
10	1	8	1
0	3	12	1
5	1	8	1
0	2	10	2
0	1	14	1
0	1	3	1
282	1	9	1
0	1	15	1
0	1	6	1
0	1	16	2
0	1	11	1
0	3	13	1
7951	1	1	1
0	1	7	1
0	1	17	1

Now, the proposed cut generation algorithm is applied on the data set in table 1, producing ordered set of values for **"Attr-1"** as:-{0, 5, 10, 13, 282, 7570, 759 }. The result of the proposed heuristic based solution is given in fig. 1 for Attr-1.

***Attr-1*:**

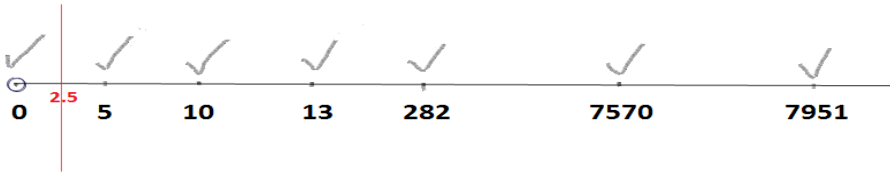


Fig. 1. Plotted Continuous Attribute values and Possible Cut Points for *Attr*-1

So, the only sufficient cut point we get is 2.5 and each one of the set of discernible pairs [16] (with respect to attr-1) consists of two data objects, one having the value from set {0} and the other having any value from the set {5,10,13, 282, 7570, 7951}.

Similarly, for two more conditional attributes, the proposed algorithm generates following cut point as shown in fig. 2 (for Attr-2) and fig. 3 (for Attr-3):-

***Attr-2*:**

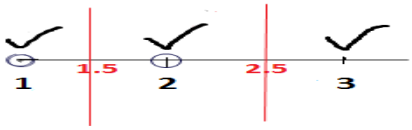


Fig. 2. Plotted Continuous Attribute values and Possible Cut Points for *Attr*-2

So, from fig. 2, the generated cut points for "*Attr-2*" are: **{1.5, 2.5}** and the discernible pairs [16] (with respect to attr-2) obtained from Fig. 2., are – {data object having *"Attr-2"* value as 1, data object having *"Attr-2"* value 2} ; { data object having *"Attr-2"* value as 2, data object having *"Attr-2"* value 3} and { data object having *"Attr-2"* value as 1, data object having *"Attr-2"* value as 3}.

***Attr-3*:**

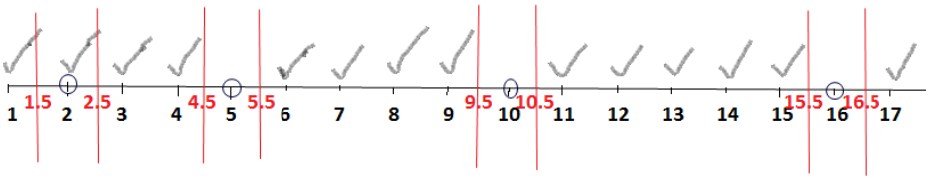


Fig. 3. Plotted continuous attribute values and possible cut points for attr-3

So, from fig. 3, the generated cut points for *Attr*-3 are: **{1.5, 2.5, 4.5, 5.5, 9.5, 10.5, 15.5, 16.5}.**

2.2 Centre-Spread Encoding Method

From the cut points of the decision system, intervals are generated and as a next step, the intervals are encoded with discrete values using Centre-Spread encoding technique. In the proposed approach, the mid-point of the interval represents the *centre* which is a discretized value of any continuous value belonging to that interval. The other variable *spread* provides limit of the continuous values on either side of the mid-point in discretized form. So, an interval is represented as (*centre*, *spread*) as described below:-

Step 1(Interval Generation):-The cut points are traced and two successive cut points are represented as an interval [*lw*,*up*) where lower one (*lw*) is included and upper one (*up*) is excluded in the interval.

Step 2(Centre-Spread encoding):- An interval denoted as [*lw*,*up*) is encoded as the centre (mid-point) of the interval and spread (span) of the interval from its mid-point to either side of the end-points.

Example 2: *The set of cut points of conditional attribute* ***"Attr-3"*** *is arranged in ascending order as:-* {1.5, 2.5, 4.5, 5.5, 9.5, 10.5, 15.5,16.5}. *Inclusion of minimum and maximum value of the attribute results ----* {**1,** *1.5, 2.5, 4.5, 5.5, 9.5, 10.5, 15.5,* 16.5**, 17**}.

From this set, following intervals are generated–

[1, 1.5):-Interval # "1"; [1.5, 2.5):-Interval # "2"; [2.5, 4.5):-Interval # "3"; [4.5, 5.5):-Interval # "4"; [5.5, 9.5):-Interval # "5"; [9.5, 10.5):-Interval # "6"; [10.5, 15.5):-Interval # "7"; [15.5, 16.5):-Interval # "8" and [16.5, 17):-Interval # "9"

Now, the intervals of ***"Attr-3"*** *are encoded using* ***Centre-Spread encoding*** *technique as follows given in table 2:-*

Table 2. Centre-Spread Encoding of Intervals

Intervals	Centre	Spread
[1, 1.5)	1.25	0.25
[1.5, 2.5)	2	0.5
[2.5, 4.5)	3.5	1
[4.5, 5.5)	5	0.5
[5.5, 9.5)	7.5	2
[9.5, 10.5)	10	0.5
[10.5, 15.5)	13	2.5
[15.5, 16.5)	16	0.5
[16.5, 17)	16.75	0.25

2.3 Discretization of Data Set

Data set has been discretized to bind the domain of continuous conditional attributes into some finite set of values so that patterns of all conditional attributes corresponding to each data object and its associated decision class has been observed. Using the Centre-Spread encoding technique, each attribute value of a data set is guaranteed to be in at most and at least one interval. The corresponding centre value is assigned as the discrete value for the respective attribute of the decision system.

Example 4: After encoding of intervals, all the center values of the set of intervals are obtained for each of the 3 continuous attributes and used to discretize corresponding continuous attributes. Finally, the discretized form of the given decision system is obtained and given in the table 3.

Table 3. Discretized form of the Decision System

Attr-1	Attr-2	Attr-3	Decision
3976.75	1.25	1.25	1
1.25	2	2	1
1.25	2	2	2
3976.75	1.25	1.25	1
1.25	1.25	3.5	1
1.25	1.25	5	2
3976.75	1.25	7.5	1
1.25	2.75	13	1
3976.75	1.25	7.5	1
1.25	2	10	2
1.25	1.25	13	1
1.25	1.25	3.5	1
3976.75	1.25	7.5	1
1.25	1.25	13	1
1.25	1.25	7.5	1
1.25	1.25	16	2
1.25	1.25	13	1
1.25	2.75	13	1
3976.75	1.25	1.25	1
1.25	1.25	7.5	1
1.25	1.25	16.75	1

3 Time Complexity Analysis

Assume that the algorithm runs on a continuous valued decision system with *"m"* numbers of conditional attributes (say, attr-1,…, attr-*m*), one decision attribute (say, *d*) and "*n*" number of objects or instances (say, obj-1,…, obj-*n*). The decision system is the input to the proposed discretization algorithm.

3.1 Complexity of the cut generation algorithm

In step 1 of the cut generation algorithm, k^{th} conditional attribute ($1 \leq k \leq m$) is considered for discretization. Now, according to their decision attribute (d) values, k^{th} conditional attribute values along with marking (step 2 and step 3) are copied into a two dimensional array (with two fields value and class) of length n. Time complexity is O(n) to perform this task. Then, the array is sorted using heap sort algorithm because it is an in-place sort and has both average and worst case time complexity is O($n$lg$n$) for sorting "n" number of elements. In step 4, the sorted array is traversed to search the cut points and simultaneously storing these cut points into an array. To perform this operation, the algorithm takes O(n) running time.

So, for one conditional attribute, the cut generation algorithm takes {O(n)+O($n$lg$n$)+O(n)} running time which is O($n$lg$n$). Now, for "m" number of conditional attributes, the time complexity of the proposed algorithm is O(mnlg$n$). However, for a given decision system m is constant and does not change with time for a given data set, only no of instances may change. So, if we treat m as a constant, then the overall running time comes out to be O($n$lg$n$).

3.2 Complexity of the Centre-Spread Encoding Method

To encode the cut-points into intervals, the array which stores the cut-points has been traversed. To perform this task, the running time is O(n) and for all conditional attributes it becomes O(mn). If m is considered as constant, then time complexity of the center-spread encoding method is O(n).

3.3 Complexity of the Discretization Algorithm

To discretize the data set using encoded intervals, each continuous value of a conditional attribute is mapped into the center point of the corresponding interval. So to search that specific interval, binary search algorithm is employed with time complexity O(lg$n$). Therefore, to discertize all values of that attribute corresponding to n instances, it takes O($n$lg$n$) time and for the whole decision system, O(mnlg$n$) running time. If m is constant then time complexity of the discretization algorithm is O($n$lg$n$).

Therefore, time complexity of the proposed discretization algorithm is O(mnlg$n$) and if m is constant, it is O($n$lg$n$).

4 Experimental Results and Comparisons

Comparisons are performed between the discretized data set obtained using the proposed discretization process and the original continuous data set in terms of classification accuracy, mean absolute error, root mean square error, relative absolute error and root relative squared.

We have considered 10000 data objects from nsl-KDD Data set [11, 12], continuous in nature. Then the proposed discretization method is applied and 10 fold cross validation technique is used for measuring classification accuracy of different classifiers. The average classification accuracy in percentage of different well known classifiers considering both discretized and continuous data set has been shown in Fig. 4.

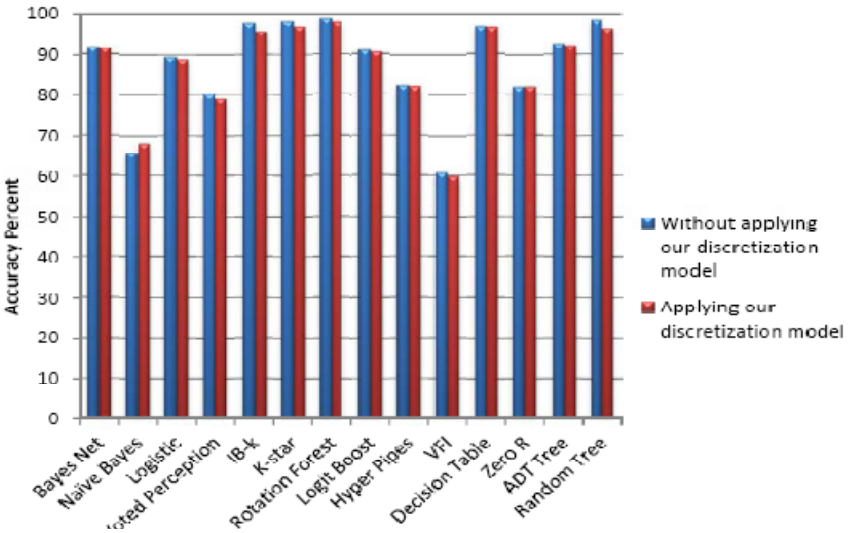


Fig. 4. Comparisons of classification accuracy considering both discretized and continuous data set

Maximum **1.9%** drop in accuracy is observed in random tree classifier while there is **2.4%** increase in average classification accuracy considering Naïve Bayes Classifier. Due to information loss in the discretization process, reduction of accuracy is inevitable. The original data set has more information than the discretized one. When the training patterns are stored in multi-dimensional feature space (for lazy classifier) or represented using the rule set (for rule based classifier), or decision tree (for tree based classifier) is generated or the best component classifier for each data point (for meta-classifier [22]) is selected then it covers much more attribute domain than discretized data set. After discretization, several attribute values (on the basis of which decisions are made) which were in the rule set or decision tree or in the multidimensional feature space are lost due to the discretization and for those attribute values decisions made may be wrong. Similarly, for meta-classifier if some of the data points are lost then the selection of best component classifier may not be optimal. This explains the drop of classification accuracy.

Discretization provides an alternative to the probability density estimation when quantitive attributes are involved in naive-Bayes learning and under this probability density estimation, if the assumed density is not a proper estimate of the true density, the naïve-Bayes classification accuracy degrades [20]. Since for real world data, the true density is usually unknown, often unsafe assumptions are made resulting less

classification accuracy. A proper discretization method can circumvent this problem by tuning interval size and interval number to find a good trade-off between discretization bias [21] and discretization variance [21] and thus can achieve low learning error and higher classification accuracy. This explains the increase in classification accuracy of naïve-Bayes classifier (shown in figure. 4.) when it runs on the discretized data set generated from the continuous one by our proposed discretization method.

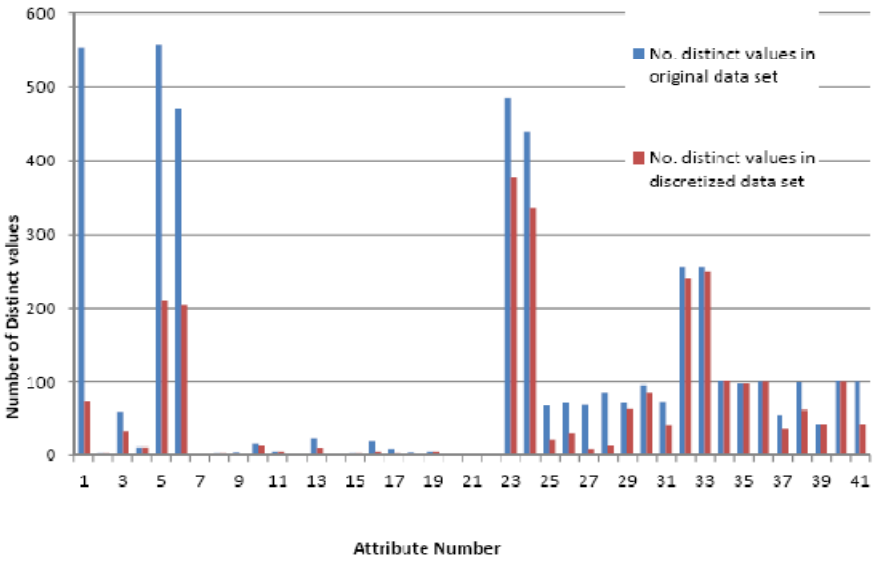


Fig. 5. Shows the difference between the distinct values in the original data set of different attributes with that of the discretized data set

Fig. 5 shows the comparisons of continuous data in the original data set into less number of distinct values in the discretized data set. Among the different attributes, the maximum reduction in data points occurred for Attribute 1 in the original data set having **550** distinct values and the discretized having only **70** distinct values. So the maximum ratio of reduction is (**550/70**) i.e. **7.85**.

In table 4, along with average classification accuracy [22] other statistics are provided showing error values which are nearly the same. Here, maximum difference in root mean square error in classification between discretized and continuous data set is 0.06. Furthermore, true positive [23], false positive [23] and F-measure [23] values are provided while classifying both the discretized and continuous data set. From the analysis of the results given in these figures, we can conclude that there is very little data loss in our proposed discretization process of a given continuous data set. Weka tool [24,25] is used for calculation of statistical parameters.

Here Precision of a class is the number of True Positives divided by the total number of elements belonging to the positive class. Recall is defined as the number of

True Positives divided by the number of elements actually belong to the positive class.

$$Prediction(p) = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad ;$$

$$Recall(r) = \frac{True\ Positive}{True\ Positive + False\ Negetive} \quad ;$$

$$F\ measure(F) = \frac{2(p * r)}{p + r}$$

Table 4. Comparisons of Correctly classified percentage, Root Mean Squared error in classification, true positive, false positive, and F-measure values

Classifier Type	Classifier Name	Without Applying our Discretization model					Applying our Discretization model				
		Correctly Classified Instance (%)	Root Mean Squared error	True Positive Rate (Class)	False Positive Rate (Class)	F – measure (Class)	Correctly Classified Instance (%)	Root Mean Squared error	True Positive Rate	False Positive Rate	F – measure
BAYES	Bayes Net	91.62	0.27	0.92(1)	0.10(1)	0.92(1)	91.45	0.27	0.92(1)	0.19(1)	0.94(1)
				0.89(2)	0.07(2)	0.79(2)			0.89(2)	0.08(2)	0.73(2)
	Naïve Bayes	65.47	0.57	0.97(1)	0.46(1)	0.74(1)	67.8	0.55	0.64(1)	0.19(1)	0.76(1)
				0.83(2)	0.38(2)	0.46(2)			0.80(2)	0.35(2)	0.47(2)
FUNCTION BASED	Logistic	89.38	0.28	0.97(1)	0.46(1)	0.93(1)	88.85	0.29	0.97(1)	0.47(1)	0.93(1)
				0.53(2)	0.02(2)	0.53(2)			0.52(2)	0.03(2)	0.63(2)
	Voted Perception	80.09	0.44	0.88(1)	0.58(1)	0.87(1)	79.05	0.45	0.91(1)	0.76(1)	0.87(1)
				0.41(2)	0.11(2)	0.43(2)			0.23(2)	0.08(2)	0.28(2)
LAZY	IB-K	97.59	0.15	0.98(1)	0.06(1)	0.98(1)	95.69	0.20	0.97(1)	0.13(1)	0.97(1)
				0.93(2)	0.01(2)	0.93(2)			0.86(2)	0.02(2)	0.88(2)
	K-Star	98.03	0.11	0.98(1)	0.05(1)	0.98(1)	96.91	0.15	0.98(1)	0.10(1)	0.98(1)
				0.94(2)	0.01(2)	0.94(2)			0.89(2)	0.01(2)	0.91(2)
META	Rotation Forest	98.46	0.10	0.99(1)	0.03(1)	0.99(1)	97.74	0.13	0.98(1)	0.06(1)	0.98(1)
				0.96(2)	0.01(2)	0.96(2)			0.93(2)	0.01(2)	0.93(2)
	Logic Boost	90.89	0.27	0.97(1)	0.38(1)	0.94(1)	90.49	0.27	0.97(1)	0.40(1)	0.94(1)
				0.61(2)	0.02(2)	0.71(2)			0.59(2)	0.02(2)	0.69(2)
MISC	Hyper Pipes	82.01	0.50	1.00(1)	0.98(1)	0.90(1)	81.81	0.5	1.00(1)	0.99(1)	0.90(1)
				0.01(2)	0.00(2)	0.03(2)			0.01(2)	0.00(2)	0.01(2)
	VFI	60.63	0.49	0.52(1)	0.04(1)	0.68(1)	59.86	0.49	0.52(1)	0.05(1)	0.67(1)
				0.95(2)	0.47(2)	0.47(2)			0.95(2)	0.48(2)	0.45(2)
RULE BASED	Decision Table	96.64	0.16	0.98(1)	0.12(1)	0.98(1)	96.5	0.16	0.98(1)	0.13(1)	0.97(1)
				0.87(2)	0.01(2)	0.90(2)			0.86(2)	0.01(2)	0.90(2)
	Zero R	81.68	0.38	1.00(1)	1.00(1)	0.89(1)	81.71	0.38	1.00(1)	1.00(1)	0.89(1)
				0.00(2)	0.00(2)	0.00(2)			0.00(2)	0.00(2)	0.00(2)
TREE BASED	ADT Tree	92.56	0.24	0.97(1)	0.30(1)	0.95(1)	92.22	0.24	0.96(1)	0.28(1)	0.95(1)
				0.69(2)	0.02(2)	0.77(2)			0.71(2)	0.03(2)	0.77(2)
	Random Tree	98.24	0.13	0.98(1)	0.04(1)	0.98(1)	96.34	0.18	0.97(1)	0.10(1)	0.97(1)
				0.95(2)	0.01(2)	0.95(2)			0.89(2)	0.02(2)	0.89(2)

5 Conclusions

In the paper, we have proposed a discretization method and applied on network traffic data set. Results are analyzed and classification accuracy is compared using different error values. It has been observed that in the proposed discretization method very little data loss is occurred. Here, sufficient number of cut points has been generated for each continuous attributes and maintains consistency in the decision system after

getting discretized using encoded intervals generated from those cut points. Therefore, each data objects in the discretized data set preserves its integrity [20] even after the proposed discretization process as it has been in continuous data set. Therefore, from this discussion, we conclude that using the proposed discretization method cut points are generated and using encoded intervals, an efficient technique of discretization has been achieved.

References

1. Williams, N., Zander, S., Armitage, G.: A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification
2. McGregor, A., Hall, M., Lorier, P., Brunskill, J.: Flow Clustering Using Machine Learning Techniques. In: Passive & Active Measurement Workshop, France (April 2004)
3. Dunnigan, T., Ostrouchov, G.: Flow Characterization for Intrusion Detection, Technical Report, Oak Ridge National Laboratory (November 2000)
4. http://software.ucv.ro/~cmihaescu/ro/teaching/AIR/docs/Lab4-NaiveBayes.pdf
5. Chai, K., Hn, H.T., Chieu, H.L.: Bayesian Online Classifiers for Text Classification and Filtering. In: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 97–104 (August 2002)
6. Badulescu, L.A.: Data Mining Algorithms Based On Decision Trees, Annals of the Oradea University. Fascicle of Management and Technological Engineering, vol. V(XV), pp. 1621–1628. Publishing House of Oradea University (2006) ISSN:1583 - 0691
7. Chaudhuri, S., Fayyad, U., Bernhardt, J.: Scalable Classification over SQL Databases. In: Proc. ICDE 1999, Sydney, Australia, pp. 470–479. IEEE Computer Society (1999)
8. Du, W., Zhan, Z.: Building Decision Tree Classifier on Private Data. In: IEEE International Conference on Data Mining Workshop on Privacy, Security, and Data Mining, Conferences in Research and Practice in Information Technology, Maebashi City, Japan, vol. 14. Australian Computer Society, Inc. (2002)
9. Kotsiantis, S., Kanellopoulos, D.: Discretization Techniques: A recent survey. GESTS International Transactions on Computer Science and Engineering 32(1), 47–58 (2006)
10. Xu, T., Yingwu, C.: Half-global discretization algorithm based on rough set theory. Journal of Systems Engineering and Electronics 20(2) (April 1, 2009)
11. Nsl-kdd data set for network-based intrusion detection systems (2009), http://nsl.cs.unb.ca/NSL-KDD/
12. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A Detailed Analysis of the KDD CUP 99 Data Set
13. http://werner.yellowcouch.org/phd03/PhdOnTheWeb/node8.html
14. Ching, J.Y., Wong, A.K.C., Chan, K.C.C.: Class-Dependent Discretization for Inductive Learning from Continuous and Mixed Mode Data. IEEE Transactions on Pattern Analysis and Machine Intelligence 17(7), 641–651 (1995)
15. Ren, Z., Hao, Y., Wen, B.: A Heuristic Genetic Algorithm for Continuous Attribute Discretization in Rough Set Theory
16. Komorowski, J., Polkowski, L., Skowron, A.: Rough Set: A tutorial
17. Kruse, R.L., Ryba, A.J.: Data structures and program design in C++. Prentice Hall (1998) ISBN-13: 9780137689958
18. Boritz, J.E.: IS Practitioners' Views on Core Concepts of Information Integrity. International Journal of Accounting Information Systems (retrieved August 12, 2011)

19. Morariu, D.I., Vintan, L.N., Tresp, V.: Meta-Classification using SVM Classifiers for Text Documents World Academy of Science, Engineering and Technology 21 (2008)
20. Dougherty, J., Kohavi, R., Sahami, M.: Supervised and unsupervised discretization of continuous features. In: Proc. of the Twelfth International Conf. on Machine Learning, pp. 194–202 (1995)
21. Yang, Y., Webb, G.I.: On Why Discretization Works for Naive-Bayes Classifiers
22. Han, Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, San Francisco (2001)
23. Neyman, J., Pearson, E.S.: The testing of statistical hypotheses in relation to probabilities a priori. Joint Statistical Papers, pp. 186–202. Cambridge University Press (1933, 1967)
24. Weka 3: Data Mining Software in Java, http://www.cs.waikato.ac.nz/ml/weka/
25. Weka User Manual http://www.gtbit.org/downloads/dwdmsem6/dwdmsem61man.pdf, http://kent.dl.sourceforge.net/project/weka/documentation/3.6.x/WekaManual-3-6-2.pdf

Parameters Optimization of Continuous Casting Process Using Teaching-Learning-Based Optimization Algorithm

Ravipudi Venkata Rao and Vivek D. Kalyankar

S.V. National Institute of Technology, Surat, Gujarat – 395007, India
ravipudirao@gmail.com, vivekkalyankar@yahoo.co.in

Abstract. In the present work, continuous casting process is considered for its parameters optimization using a recently developed optimization algorithm known as teaching-learning-based optimization algorithm. The example considered is a multi-objective multi-constrained problem having three objectives and two constraints. The objectives of the process under consideration are maximization of lubrication index and minimization of peak friction and oscillation marks. Four process parameters are involved in the process namely casting speed, stroke, frequency and deviation from sinusoid. The same model was earlier attempted by researchers using genetic algorithm and differential evolution techniques. The results obtained in the present work outperformed the previous results in terms of fitness functions and number of generations.

1 Introduction

Casting is a very age old manufacturing process used for production of various complicated shapes that cannot be easily manufactured by any machining process. Based on the basic working principle, various types of castings processes are now available such as sand casting, die casting, continuous casting, investment casting, etc. and each type of casting process is having its particular application area. Due to low cost and higher production rates, continuous casting process is now getting widely used for the mass production of various billets, blooms, thin sheets, etc. Regularly used metals in industries such as steel, copper and aluminium can be easily casted by using continuous process.

In continuous casting process, molten metal is poured from one end and the solidified slabs come out from other end through various rollers. However, control over various process parameters is very important in this process to achieve the desired quality product. Various input process parameters involved in continuous casting process are: casting speed, flux viscosity, flux density, flux solidification temperature, mold thickness, mold stroke length, cooling water temperature, cooling water flow rate, etc. Each of these parameters directly affects the process outputs. Non-metallic inclusion in the form of oxides is one of the major problems in continuous steels that can lead to excessive casting repairs or rejected castings. To avoid such problems, close control over the casting speed, cooling water flow rate,

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 540–547, 2012.

etc. is very important. By setting these various important parameters by trial may lead to more rejections and thereby decreasing the production rate and increasing the cost. Selection of optimum process parameters will help to avoid all these problems and this can be achieved by using advanced optimization algorithms.

In the literature it is observed that few researchers had carried out research on the continuous casting process. Zhou et al. [1] used nondominated sorting genetic algorithm for multiobjective optimization of a continuous casting process. The process was operated for producing a film of methyl methacrylate. Two objective functions were attempted in the work which involved maximization of the cross-section average value of the monomer conversion of the product and minimization of the length of the film reactor. Seven decision variables were used in the study which included the temperature, feed flow rate, the film thickness, the monomer conversion at the output, and three coefficients describing the wall temperature used in the film reactor. Effect of various decision variables was discussed and the result in the form of Pareto set was presented. Cheung and Garcia [2] carried out optimization of the quality of a SAE 1010 steel billet by using heuristic search technique. A knowledge base was proposed in the work by interacting a numerical heat transfer model and an artificial intelligence (AI) heuristic search method. The work was focused on obtaining the optimized cooling conditions for the continuous casting process which results in defect-free billet production.

Chakraborti et al. [3] used pareto-converging genetic algorithm for optimizing the casting velocity. Two different objectives were considered for casting velocity subjected to three constraints due to a number of physical considerations. The work was restricted only to the mold region. Subsequently, Chakraborti et al. [4] extended their work on the spray and radiation cooling regions. In another work, Chakraborti et al. [5] used finite volume approach along with genetic algorithm to solve the pertinent transport equations of a continuous caster mold and reported improvement in casting velocity and solidified shell thickness.

Santos et al. [6] used genetic algorithm and a knowledge base of operational parameters to develop a mathematical model and a computational algorithm to maximize the quality of steel billets produced by a continuous casting process. The optimization strategy was concerned with achieving a set of optimal cooling conditions in order to attain highest product quality. In another work, Santos et al. [7] developed a numerical code based on genetic algorithm for an industrial data of a continuous caster machine and determined optimum settings of water flow rates in different sprays zones.

Ghosh et al. [8] discussed the problems occurred in a continuous casting of a thin slab due to meniscus-level fluctuation. Genetic algorithm was used to find the optimum combination of spray cooling. The total bulging was minimized and also the surface temperature at the slab exit was also maintained minimum. Kulkarni and Babu [9] attempted the multiobjective optimization problem for producing quality products in a continuous casting system. Simulated annealing algorithm was used to obtain the optimum process parameters in order to satisfy 17 critical quality conditions. Cheung et al. [10] determined an improved cooling condition of mold and spray cooling zones of a continuously billet casting machine and suggested suitable modification in the secondary cooling zone by proposing provision of two spray zones. A heuristic search

technique supported by a Knowledge Base and a heat flow mathematical model was used for determining the results and the suggested modification resulted in a shorter metallurgical length and lower surface reheating compared with the set of original spray zones.

Filipic et al. [11] used differential evolution for multi-objective optimization technique for optimizing the coolant flows in continuous casting of steel in order to satisfy multiple objectives associated with temperature and core length. Miettinen [12] described an interactive classification-based multiobjective optimization method: NIMBUS which converts the original objective functions together with preference information coming from the decision maker into scalar-valued optimization problems. A real-life problem related to continuous casting of steel was attempted in order to minimize the defects in the final product.

Bhattacharya et al. [13] used genetic algorithm to obtain the optimum parameters of the mould oscillation system in the continuous casting process. The important parameters considered were casting speed, stroke, frequency and deviation from sinusoid. The main focus of the work was to maximize the lubrication index and minimize depth of oscillation marks and friction. The comparison of results was made with the conditions described by the original equipment manufacturer. Subsequently, Bhattacharya and Sambasivam [14] used differential evolutionary algorithm for the same problem and compared the results with those obtained by genetic algorithm.

Ye et al. [15] described an Engineering-Driven Rule-Based Algorithm (ERD) and various challenges related to the bleeds detection were explained. An attempt was made to solve bleed detection problem using a real case study in which miss detection rate and false alarm rate were optimized by considering the four important process variables related to the geometry and two important coefficients of the process. A full factorial three-level experimental design was used for determining the process parameters to conduct the experiments. Lopez et al. [16] created a simulator for the continuous casting process using a computational algorithm based on the numerical method. The results obtained by the simulator were validated for the three different steel casters produced in an industry. Jabri et al. [17] used particle swarm optimization for tuning the process parameters of a real plant in order to improve the bulging effect rejection.

It is observed from the literature that among the various optimization techniques available, genetic algorithm (GA) [13], differential evolutionary (DE) [14] and particle swarm optimization (PSO) [17] algorithms were used by some researchers for the parameters optimization of continuous casting process. However, subsequently it is proved by many researchers of different fields that the results given by GA, DE, PSO, etc. are not always optimum. Thus it can be concluded that there is a big scope for making use of recently developed advanced optimization techniques for optimizing the continuous casting process parameters. Hence, in the present work, an attempt is made to optimize the parameters of continuous casting process by using a recently developed advanced optimization algorithm named as teaching-learning-based optimization (TLBO) algorithm. In the next section the important steps of TLBO algorithm are described and then the same is applied to an example of continuous casting process.

2 Teaching-Learning-Based Optimization Algorithm

Teaching-learning-based optimization algorithm (TLBO) is a teaching-learning process inspired algorithm recently proposed by Rao et al. [18-19] and Rao and Patel [20] based on the effect of influence of a teacher on the output of learners in a class. The algorithm mimics teaching-learning ability of teacher and learners in a class room. In this algorithm a group of learners is considered as population and different subjects offered to the learners are considered as different design parameters and a learner's result is analogous to the 'fitness' value of the optimization problem. The best solution in the entire population is considered as the teacher. The design parameters are actually the parameters involved in the objective function of the given optimization problem and the best solution is the best value of the objective function. The teacher phase is the first part of the algorithm where learners learn through the teacher. During this phase a teacher tries to increase the mean result of the class room to his or her level. Learner phase is the second part of the algorithm where learners increase their knowledge by interaction among themselves. A learner interacts randomly with other learners for enhancing his or her knowledge. A learner learns new things if the other learner has more knowledge than him or her. The working of the TLBO algorithm is described in detail by Rao et al. [18-19] and the same explanation is referred here for the working of TLBO algorithm. The next section presents the applications of this advanced algorithm for the parameter optimization of continuous casting process.

3 Parameters Optimization of Continuous Casting Process

This example is taken from the literature based on the research work of Bhattacharya and Sambasivam [14] which was related to the optimization of parameters in continuous casting process of steel manufacturing. In their work, Bhattacharya and Sambasivam [14] used differential evolution technique on the mathematical models of continuous casting process and compared their result with those obtained by genetic algorithm. The mathematical model used by them consisted of four input parameters: casting speed 'v', stroke's', frequency 'f' and deviation from sinusoid 'τ'. The work attempted by Bhattacharya and Sambasivam [14] was a multi-objective problem and it consisted of three objectives namely maximization of lubrication index 'LI', minimization of peak friction 'PF' and minimization of oscillation marks 'OM'. These three objectives were represented by mathematical models in terms of input parameters. The same models as used by Bhattacharya and Sambasivam [14] are given below by equations (1)-(6).

Maximization of lubrication index 'LI',

$$LI = (R_{NA})^{0.3} * (t_p)^{0.5} \tag{1}$$

$$R_{NA} = f * (N_d/V_c) * 100 \tag{2}$$

$$N_d = s* \sin(\Pi * f * t_{neg}) - V_c * t_{neg} \tag{3}$$

$$t_{neg} = (1/\Pi * f) * \cos^{-1}(V_c/ \Pi * 2 * s * f) \quad (4)$$

$$t_p = (1/f) - t_{neg} \quad (5)$$

Where N_d is the negative strip distance, V_c = v (casting speed), t_p is the positive strip time and t_{neg} is the negative strip time (NST).

Equation (1) represents the mathematical model for lubrication index whereas equations (2)-(5) can be used to obtain the required terms. A single equation can also be obtained for lubrication index by putting the related terms directly in equation (1).

Minimization of peak friction 'PF',

$$PF = \eta * (V_{max}^{up} - V_c)/d_t \quad (6)$$

Where V_{max}^{up} is the maximum upward speed in a cycle, η is the viscosity of lubricant and d_t is the thickness of lubricant film in the gap between strand and mould.

Bhattacharya and Sambasivam [14] combined all the three objectives by assigning some weightage to each objectives and formed a single objective referred as performance metric 'PM' which is given below by equation (7)

$$PM = w_1 \text{ x } (LI)^2 - w_2 \text{ x } (PF)^2 - w_3 \text{ x } (OM)^2 \quad (7)$$

Where, w_1, w_2 and w_3 are the weightages assigned to the each objective and LI, PF and OM are the respective objectives of lubrication index, peak friction and oscillation marks. The lubrication index 'LI' can be directly obtained by using equation (1), however, Bhattacharya and Sambasivam [14] showed in their work that there is a direct relationship for PF with stroke 's' and OM with frequency '*f*'. Hence the performance metric 'PM' was then further modified by them as given below by equation (8) – (9).

$$PM = w_1 \text{ x } (LI)^2 - w_2 \text{ x } (s)^2 + w_3 \text{ x } (f)^2 \quad (8)$$

$$PM = w_1 \text{ x } (LI)^2 - w_2 \text{ x } (NST)^2 - w_3 \text{ x } (PFF)^2 \quad (9)$$

Where, NST is the negative strip time also denoted as t_{neg} and PFF is the peak friction factor calculated as PFF=(V_{max}-V). The process under consideration was also having 2 constraints. The constraints and parameter bounds as given below:

$$(s * \Pi * f)/ 2*(1- \tau) \leq 0.8 * V_{max} \quad (10)$$

$$(s * \Pi^2 * f^2)/ 2*(1- \tau)^2 \leq 0.8 * \alpha_{max} \quad (11)$$

$$80 \leq f \leq 200 \quad (12)$$

$$2 \leq s \leq 10 \quad (13)$$

$$0.5 \leq \tau \leq 0.7 \quad (14)$$

Even though casting speed 'v' is an input parameter, but Bhattacharya and Sambasivam [14] used five different casting speeds from 0.5 – 1.7 and obtained the optimized parameter setting of remaining three parameters for each set of casting speed. Differential evolution technique was used by them for the parameters optimization of the process under consideration. In another work, Bhattacharya et al.

[13] attempted the same models by using genetic algorithm and compared their results with those given by original equipment manufacturer (OEM).

In the present work, same mathematical models, constraints and bounds, as used by Bhattacharya and Sambasivam [14] are used so as to have the fair comparison with previous results. Initially the TLBO algorithm is tried with different combinations of population sizes and number of generations and finally a population size of 10 is used throughout while attempting various combinations of casting speeds. The results obtained by TLBO algorithm along with its comparison with GA and DE are given in Table 1.

Table 1. Comparison of results of TLBO with those of GA and DE

	GA [14]				**DE** [13]				**TLBO**			
Casting speed (m/min)	Stroke (mm)	Frequency (cpm)	Fitness (Z)	No. of Generations	Stroke (mm)	Frequency (cpm)	Fitness (Z)	No. of Generations	Stroke (mm)	Frequency (cpm)	Fitness (Z)	No. of Generations
0.5	10	113	0.2011	3835	10	113	0.2011	46	10	102	**0.2518**	**20**
0.8	10	125	0.157	1240	10	125	0.157	75	10	100	**0.2273**	**20**
1.1	10	115	0.159	1731	10	115	0.159	67	10	105	**0.1932**	**20**
1.4	10	106	0.1651	2830	10	106	0.1651	49	10	105	**0.2138**	**20**
1.7	10	115	0.145	1350	10	115	0.145	70	10	82	**0.5147**	**20**

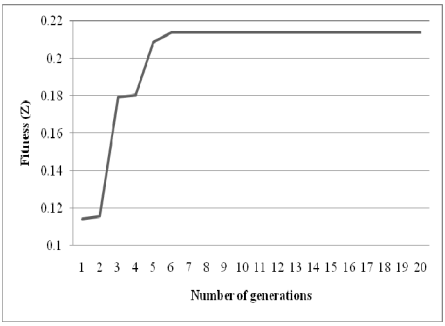


Fig. 1. Convergence of result obtained by TLBO algorithm

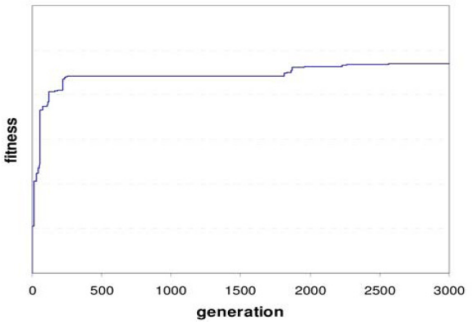


Fig. 2. Convergence of result obtained by GA [14]

From the Table 1, it is observed that the TLBO algorithm has taken very less number of generations for convergence as compared to that taken by GA and DE. In case of GA, the numbers of generations taken were in the range of 1240 to 3835, while in case of DE the numbers of generation taken were varying from 46 to 75. But in case of TLBO algorithm, 20 generations are sufficient to obtain the consistent result. However, the TLBO algorithm has given the optimum results in less than six generations in various cases, but 20 generations are used in all the cases to check for constant result. This indicates the faster convergence capability of the TLBO algorithm which helps to reduce the computational efforts as compared to other optimization algorithm like GA, DE, etc. The convergence of result obtained by TLBO algorithm for casting speed of 1.4 m/min is shown in Fig. 1 whereas for the

same case, the convergence obtained by GA, as given by Bhattacharya and Sambasivam [14], is produced in Fig. 2 for the comparision purpose.

The results given by TLBO algorithm is also very significant compared to that of GA and DE. The TLBO algorithm has given considerable improvement in result in case of all the five casting speed combinations considered for optimization. The frequency suggested by the TLBO algorithm is different than that of GA and DE. For lower casting speed of 0.5 m/min, the fitness function is improved from 0.2011 to 0.2518 thereby achieving more than 25% improvement in the result over that of GA and DE. In case of higher casting speed of 1.7 m/min, the improvement in result is more than 350%. Similarly, for other casting speeds, the TLBO algorithm has given improvement ranging from 20-45% as shown in Table 1. Thus the TLBO algorithm is proved better in terms of results and computational time over that of GA and DE. The TLBO algorithm has also proved its suitability in the field of parameters optimization of continuous casting process.

4 Conclusion

The continuous casting process parameters optimization problem considered in the present work is a multi-objective multi-constrained problem having three objectives and two constraints. Previous researchers had used genetic algorithm and differential evolution techniques to optimize the process parameters of the continuous casting process. Same models and constraints are used in the present work to have a fair comparison of the results. The TLBO algorithm has given a significant improvement in the results for all the cases of casting speed by improving the results ranging from 20% upto 350% for various cases. Only 20 generations are required for the TLBO algorithm for convergence of result as compared to 3835 generations taken by GA and 75 generations taken by DE. Hence TLBO algorithm is proved superior to GA and DE in terms of fitness function and result convergence. The output of this work will also help the designer and process engineers to make effective use TLBO algorithm for the parameters optimization of continuous casting process in order to satisfy their objectives and constraints. Similarly, the TLBO algorithm can also be tried for parameter optimization of other casting processes.

References

1. Zhou, F., Gupta, S.K., Ray, A.K.: Multiobjective Optimization of the Continuous Casting Process for Poly (methyl methacrylate) Using Adapted Genetic Algorithm. Journal of Applied Polymer Science 78, 1439–1458 (2000)
2. Cheung, N., Garcia, A.: The Use of a Heuristic Search Technique for the Optimization of Quality of Steel Billets Produced by Continuous Casting. Engineering Applications of Artificial Intelligence 14, 229–238 (2001)
3. Chakraborti, N., Kumar, R., Jain, D.: A Study of the Continuous Casting Mold Using a Pareto-Converging Genetic Algorithm. Applied Mathematical Modeling 25, 287–297 (2001)

4. Chakraborti, N., Gupta, R.S.P., Tiwari, T.K.: Optimisation of Continuous Casting Process using Genetic Algorithms: Studies of Spray and Radiation Cooling Regions. Iron Making and Steel Making 30(4), 273–278 (2003)
5. Chakraborti, N., Kumar, K.S., Roy, G.G.: A Heat Transfer Study of the Continuous Caster Mold using a Finite Volume Approach Coupled with Genetic Algorithms. Journal of Materials Engineering and Performance 12, 430–435 (2003)
6. Santos, C.A., Spim, J.A., Garcia, A.: Mathematical Modelling and Optimization Strategies (Genetic Algorithm and Knowledge Base) Applied to the Continuous Casting of Steel. Engineering Applications of Artificial Intelligence 16, 511–527 (2003)
7. Santos, C.A., Cheung, N., Garcia, A.: Application of a Solidification Mathematical Model and a Genetic Algorithm in the Optimization of Strand Thermal Profile along the Continuous Casting of Steel. Materials and Manufacturing Processes 20, 421–434 (2005)
8. Ghosh, S., Mitra, K., Basu, B., Jategaonkar, Y.A.: Control of Meniscus-Level Fluctuation by Optimization of Spray Cooling in an Industrial Thin Slab Casting Machine Using a Genetic Algorithm. Materials and Manufacturing Processes 19(3), 549–562 (2004)
9. Kulkarni, M.S., Babu, A.S.: Managing Quality in Continuous Casting Process Using Product Quality Model and Simulated Annealing. Journal of Materials Processing Technology 166, 294–306 (2005)
10. Cheung, N., Santos, C.A., Spim, J.A., Garcia, A.: Application of a Heuristic Search Technique for the Improvement of Spray Zones Cooling Conditions in Continuously Cast Steel Billets. Applied Mathematical Modelling 30, 104–115 (2006)
11. Filipic, B., Tusar, T., Laitinen, E.: Preliminary Numerical Experiments in Multiobjective Optimization of a Metallurgical Production Process. Informatica 31, 233–240 (2007)
12. Miettinen, K.: Using Interactive Multiobjective Optimization in Continuous Casting of Steel. Materials and Manufacturing Processes 22, 585–593 (2007)
13. Bhattacharya, A.K., Sambasivam, D., Roychowdhury, A., Das, J.: Optimization of Continuous Casting Mould Oscillation Parameters in Steel Manufacturing Process Using Genetic Algorithms. In: IEEE Congress on Evolutionary Computation, pp. 3998–4004 (2007)
14. Bhattacharya, A.K., Sambasivam, D.: Optimization of Oscillation Parameters in Continuous Casting Process of Steel Manufacturing: Genetic Algorithms versus Differential Evolution. In: Evolutionary Computation, pp. 77–102 (2009)
15. Ye, E.P.L., Shi, J., Chang, T.S.: On-Line Bleeds Detection in Continuous Casting Processes Using Engineering-Driven Rule-Based Algorithm. Journal of Manufacturing Science and Engineering 131, 610081–610089 (2009)
16. Lopez, A.R., Cortes, G.S., Pardave, M.P., Romo, M.A.R., Lopez, R.A.: Computational Algorithms to Simulate the Steel Continuous Casting. International Journal of Minerals, Metallurgy and Materials 17(5), 596–607 (2010)
17. Jabri, K., Dumur, D., Godoy, E., Mouchette, A., Bele, B.: Particle Swarm Optimization Based Tuning of a Modified Smith Predictor for Mould Level Control in Continuous Casting. Journal of Process Control 21, 263–270 (2011)
18. Rao, R.V., Savsani, V.J., Vakharia, D.P.: Teaching–Learning-Based Optimization: A Novel Method for Constrained Mechanical Design Optimization Problems. Computer Aided Design 43, 303–315 (2011)
19. Rao, R.V., Savsani, V.J., Vakharia, D.P.: Teaching–Learning-Based Optimization: An Optimization Method for Continuous Non-Linear Large Scale Problem. Information Sciences 183, 1–15 (2012)
20. Rao, R.V., Patel, V.: An Elitist Teaching–Learning-Based Optimization Algorithm for Solving Complex Constrained Optimization Problems. International Journal of Industrial Engineering Computations 3(4) (2012), doi:10.5267/j.ijiec.2012.03.007

Genetic Algorithm Based Approach for Optimal Allocation of TCSC for Power System Loadability Enhancement

Almoataz Y. Abdelaziz[1], M.A. El-Sharkawy[1], M.A. Attia[1], Bijaya Ketan Panigrahi[2]

[1] Department of Electrical Power & Machines,
Faculty of Engineering, Ain Shams University, Cairo, Egypt
[2] Department of Electrical Engineering,
Indian Institute of Technology, Delhi, India

Abstract. This paper presents an approach to find the optimal location of thyristor controlled series compensators (TCSC) in a power system to improve the loadability of its lines and minimize its total loss. Also the proposed approach aims to find the optimal number of devices and their optimal compensation levels by using genetic algorithm (GA) based approach with taking into consideration the thermal and voltage limits. Examination of the proposed approach is carried out on a modified IEEE 30-bus system.

Keywords: TCSC, Optimal Location, Genetic Algorithms.

1 Introduction

Flexible AC Transmission Systems, called FACTS, got in the recent years a well-known term for higher controllability in power systems by means of power electronic devices. FACTS-devices provide a better adaptation to varying operational conditions and improve the usage of existing installations. The basic applications of FACTS-devices are [1]:

- Power flow control.
- Increase of transmission capability.
- Voltage control.
- Reactive power compensation.

One of these applications - deregulated power systems - suffers from congestion management problems. Also they cannot fully utilize transmission lines due to excessive power loss that it could cause. FACTS devices such as thyristor-controlled series compensators (TCSC) can, by controlling the power flow in the network, help reducing the flows in heavily loaded lines. Also they can minimize the power loss of the systems. However, because of the considerable cost of FACTS devices, it is important to minimize their number and obtain their optimal locations in the system.

The basic operation of TCSC can be easily explained from circuit analysis. It consists of a series compensating capacitor shunted by a thyristor-controlled reactor

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 548–557, 2012.

(TCR). TCR is a variable inductive reactor X_L controlled by firing angle α. Variation of X_L with respect to α is given by equation (1) [2].

$$X_L(\alpha) = X_L \frac{\pi}{\pi - 2\alpha - \sin 2\alpha} \tag{1}$$

Net reactance of TCR, X_L (α) is varied from its minimum value X_L to maximum value infinity. Likewise effective reactance of TCSC starts increasing from TCR X_L value till occurrence of parallel resonance condition $X_L(\alpha) = X_C$, theoretically X_{TCSC} is infinity. This region is inductive region. Further increasing of $X_L(\alpha)$ gives capacitive region, starts decreasing from infinity point to minimum value of capacitive reactance X_C. Thus, impedance characteristics of TCSC shows, both capacitive and inductive region are possible through varying firing angle (α) as shown in Figure 1 [2].

Also, it can be noted from Figure 1 that:

$90 < \alpha < \alpha_{Llim}$	Inductive region
$\alpha_{Clim} < \alpha < 180$	Capacitive region
$\alpha_{Llim} < \alpha < \alpha_{Clim}$	Resonance region

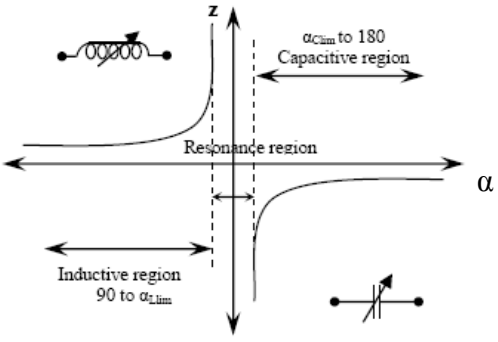


Fig. 1. Impedance Vs firing angle characteristic curve [2]

While selecting inductance, X_L should be sufficiently smaller than that of the capacitor X_C to get both effective inductive and capacitive reactance across the device [2].

In [3], the TCSC may have one of the two possible characteristics: capacitive or inductive, respectively to decrease or increase the overall reactance of the line X_L. It is modeled with three ideal switched elements connected in parallel. In order to avoid resonance, only one of the three elements can be switched at a time. Moreover, in order to avoid overcompensation of the line, the maximum value of the capacitance is fixed at -0.8 X_L. For the inductance, the maximum is 0.2 X_L. The TCSC model presented in [3] is shown in Figure 2.

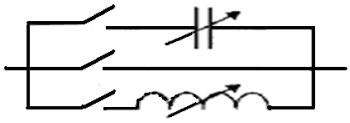


Fig. 2. Model of the TCSC [3]

In [4], the TCSC is a capacitive reactance compensator which consists of a series capacitor bank shunted by a thyristor-controlled reactor to provide a smooth control of the series capacitive reactance. Model of the TCSC presented in [4] is shown in Figure 3.

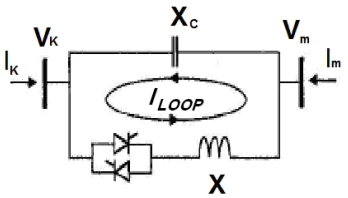


Fig. 3. Model of the TCSC [4]

Another TCSC model has been used in [5]. According to this model, a variable reactance is inserted in series with the line to be compensated which is similar to the model used in [2]. This model which is shown in Figure 4 is used in this paper and the reactance is assumed to vary in the range from -0.3 X_L to -0.7 X_L.

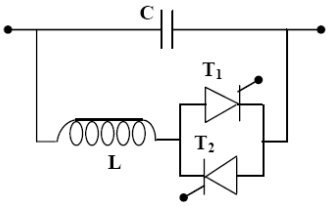


Fig. 4. The TCSC model used in this paper [2]

Several research works are carried out to solve the optimal location problem of the TCSC. Optimization techniques applied in most of these works cannot be accepted as general optimization techniques as they used a fixed pre-specified number of FACTS devices as in [3, 6, 7] i.e. they have for example 5 devices to select location for them but to be generalized, the number of FACTS should be optimized also. Some other works did not select the proper type or the proper working range of FACTS devices used in the optimization problem. For example in [8, 9], shunt FACTS devices are used while the objective is to increase loadability. The proper type in this case is series or (shunt & series) devices. Other works select range of FACTS devices which is not a function in line reactance. For example in [10], the range of TCSC was taken from -0.05 to 0.05 p.u. which may cause resonance. Thus the proper working range should be a function of the line reactance to avoid resonance.

Power system can, in general, be measured by system loadability and/or system losses at a condition that nodal voltage magnitudes are kept within acceptable limits and thermal constraints of system elements are not violated. According to [3] such optimization problem can be solved by using heuristic methods such as genetic algorithms (GAs) [11, 12]. It belongs to the category of random search algorithms which simulate the evolution process based on the theory of survival of the fittest.

In this paper, an approach to find the optimal location of thyristor-controlled series compensator (TCSC) in the power system to improve the loadability of the lines and minimize the total loss using GA is presented. The proposed approach aims to find the optimal number of devices and their optimal compensation levels with taking into consideration the thermal and voltage limits. Examination of the proposed approach is carried out on a modified IEEE 30-bus system.

2 The Proposed Optimization Technique

The problem is to find the optimum numbers, locations and reactances of the TCSCs to be used in the power system. This problem is a nonlinear multi-objective one. The GA method will be used in this paper where it only uses the values of the objective function and less likely to get trapped at a local optimum. The selected method is to use two genetic algorithms with number of generations of 30, fitness limit of zero and the other parameters are taken as the default values in MATLAB (e.g. population size = 20). The first one is to find the location and number of TCSC devices by computing the minimum total loss after inserting TCSC in the system. After location and number of TCSC are obtained they have been given to another genetic algorithm to obtain the best rating of TCSC by also computing the total loss. Details of program are as follows:

• The program starts with a group of random population for the location in binary, then this random population is multiplied by the values of TCSCs in a specified range and the result of the multiplication will change in the reactance of the system.
• After that power flow is carried out for the system with TCSC all over the range.
• Then, the total loss is calculated and fitness function is computed.
• Finally stopping criteria is checked if it is not reached, another generation will start by reproduction, crossover and mutation.

Two optimization techniques using GA are used to solve the TCSC optimal location problem.

- **First technique**

In the first technique the objective is to minimize the total losses without taking into consideration limitations on the number of devices, i.e. it is required to minimize the objective function:

$$\text{Total system losses} = \text{Sum of real loss of all system lines} \tag{2}$$

- **Second technique**

Same as the first technique, but the number of devices is considered, i.e. it is required to minimize the objective function:

(Total system losses after applying the TCSCs) / (Total system losses before applying the TCSCs) + (Number of TCSC devices / Total number of locations available for connecting the TCSCs) (3)

Calculation of total loss is obtained by using MATLAB m-files in MATPOWER [14] to calculate the power flow of the system and compute the sum of real losses.

In this paper the reactance of each branch in MATPOWER case is replaced by variable reactance function of the value of TCSC reactance added as in equation (3).

$$\text{New reactance} = \text{Old reactance} + X_{TCSC} \quad (4)$$

3 Case Study

The proposed optimization technique has been applied to the IEEE 9-bus system given in [13] and the obtained results are compared with the work in [14]. The comparison showed that the proposed optimization technique gave better results than [14] with little number of devices. More details of this comparison are presented in [15]. In this paper, the modified IEEE 30-bus system is taken as the system under study. A one line diagram of the system is shown in Figure 5. The data of the system is given in [13]. The system consist of 30 buses, 41 branches and 6 generators at buses 1, 2, 22, 23, 27 and 13. The case under study is an outage of the line connecting bus1 and bus 2 where it is the most sever line [4] and increasing the load of bus 8 by 50% where it is the bus with largest load of the system. This study is carried out for range of TCSC from -30% to -70% of the line reactance. The two previous techniques of optimal allocation of TCSC are considered.

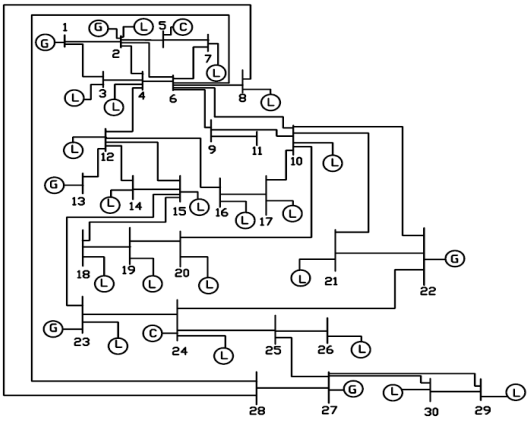


Fig. 5. Modified IEEE 30-bus system

4 Simulation Results

After running the program with the 41 locations (number of branches) available for the TCSC allocation, the power flow results with and without TCSC show that:

- Line (6-8) power is more than its rated value by 18.5% without any TCSC.
- After TCSC insertion, the power of line (6-8) returns from 118.5% of its rated value to 97.9 % in case of first technique and to 98.6% in case of second technique.
- The voltage of bus 8 (minimum voltage of the system) = 0.944 without any FACTS devices.
- After TCSC insertion, the voltage of bus 8 increases to 0.957 in first technique and to 0.958 in second technique i.e. system stability is increased.

Table 1 shows the system performance summary before and after TCSC insertion.

Table 1. System performance summary before and after TCSC insertion

	Without TCSC	**First technique after TCSC insertion**	**Second technique after TCSC insertion**
Actual Generation (MW)	208	207.8	207.7
Load (MW)	204.2	204.2	204.2
V_{min} (p.u.)	**0.944**	**0.957**	**0.958**
Min Power angle (Delta $_{min}$) (degrees)	-8.05	-8.24	-7.79
V_{max} (p.u.)	1	1	1
Max Power angle ($Delta_{max}$) (degrees)	0	0	0
$P_{loss\ max}$ (MW)	0.9	0.89	0.89
$Q_{loss\ max}$ (MVAr)	3.41	3.37	3.36
Total loss (MW)	**3.76**	**3.58**	**3.52**
Line (6-8) power as a percentage of its rated power	**118.5**	**97.9**	**98.6**
Number of devices	------	21	20

From the previous analysis, it can be noticed that although small improvement in system performance after TCSC insertion has been obtained, still large number of TCSC devices is used. Therefore, increasing the number of population size is another trial to enhance the performance of GA in order to minimize the number of devices. Table 2 shows the system performance summary before and after TCSC insertion with increasing the population size to 50.

It is noticed that after increasing the population size, results of the first technique have not changed a lot, but by using the second technique, the number of devices is reduced from 20 to 11 in the expense of total loss where it increased from 3.52 MW to 3.686 MW. It is also noticed that the number of devices is still large, thus by removing the devices that are located on the lines which are far from the line connecting buses 6 and 8, it can be noticed that only two devices are needed in lines (6-7, 8-28) and the results of system performance is given in Table 3.

Table 2. System performance summary before and after TCSC insertion with increasing the population size

	Without TCSC	**First technique after TCSC insertion with increasing population size to 50**	**Second technique after TCSC insertion with increasing population size to 50**
Actual Generation (MW)	208	207.7	207.9
Load (MW)	204.2	204.2	204.2
V_{min} (p.u.)	**0.944**	**0.957**	**0.95**
Min power angle (Delta $_{min}$) (degrees)	-8.05	-6.14	-5.7
V_{max} (p.u.)	1	1	1
Max power angle ($Delta_{max}$) (degrees)	0	0	0
$P_{loss\ max}$ (MW)	0.9	0.88	0.89
$Q_{loss\ max}$ (MVAr)	3.41	2.35	1.91
Total loss (MW)	**3.76**	**3.49**	**3.686**
Line (6-8) power as a percentage of its rated power	**118.5**	**99.1**	**99.59**
Number of devices	------	18	11

Table 3. System performance summary with only two TCSC devices

Actual Generation (MW)	208
Load (MW)	204.2
V_{min} (p.u.)	0.946
Delta $_{min}$ (degree)	-8.09
V_{max} (p.u.)	1
Delta $_{max}$ (degree)	0
$P_{loss\ max}$ (MW)	0.9
$Q_{loss\ max}$ (MVAr)	3.42
Total loss (MW)	3.81
Number of devices	2
Line (6-8) power = 118.5% of its rated value before insertion of devices	After insertion of TCSC, Line (6-8) power = 95.9% from its rated value

4.1 Minimum Number of Devices

It is clear from the previous results in Table 3 that optimization of the total loss is still not obtained. Therefore a modification to the program has been proposed to make GA search for only the locations in the line which is out of its rated value and the lines around it i.e. (8 available locations instead of 41 locations). The 8 locations are lines (4-6, 2-6, 7-6, 6-8, 6-9, 6-10, 6-28, 8-28) and population size will be returned to the default value of the MATLAB.

Now, it is required to minimize the new proposed objective function:

(total loss / total loss without devices) + (Number of devices / 8) (4)

Also observation of power limit and voltage limit are considered in the program but comparing between V_{min} as 0.9 p.u. and 0.95 p.u. is carried out.

After running the program with 8 available locations for the TCSC allocation as mentioned before, the power flow results with and without TCSC show that:

- Only 3 devices are needed to return system to stability rather than 20 devices in other results. The locations and compensation levels of the 3 devices are as in Table 4.
- Power of line (6-8) has returned within its rated value. Power of Line (6-8) decreases from 118.5% to 98.14 % in case of minimum voltage of the system is limited to 0.95 p.u. and to 99.63% in case of minimum voltage of the system is limited to 0.9 p.u.
- Voltage of bus 8 (minimum voltage of the system) was 0.944 p.u. without any FACTS devices, but after TCSC insertion the voltage of bus 8 increases to 0.95 p.u. in both of the previous cases of voltage limits.

Table 4. Locations and compensation levels of the TCSC devices

from bus	to bus	% p from rated	X_{old} (p.u.)	Compensation level % with minimum number of devices and V > 0.95 (p.u.)	Compensation level % with minimum number of devices and V > 0.9 (p.u.)
4	6	45.2	0.04	70.00	70.00
6	10	9.3	0.56	70.00	70.00
8	28	-23.1	0.2	62.41	59.57

Table 5 shows the final summary of the results.

Table 5. Final summary of the results

TCSC from $-0.7X_L$ to $-0.3X_L$	Without TCSC	First technique after TCSC insertion	Second technique after TCSC insertion	First technique after TCSC insertion - population size = 50	Second technique after TCSC insertion - population size = 50	Min No. of devices V > 0.95	Min No. of devices V > 0.9
Actual Generation (MW)	208	207.8	207.7	207.7	208	207.9	207.9
Load (MW)	204.2	204.2	204.2	204.2	204.2	204.2	204.2
V_{min} (p.u.)	0.944	0.957	0.958	0.96	0.95	0.950	0.95
P% (percentage power) from rated power of line (6-8)	118.5	97.9	98.6	99.1	99.59	98.14	99.63
Delta $_{min}$ (degree)	-8.05	-8.24	-7.79	-6.14	-5.71	-7.31	-7.31
V_{max} (p.u.)	1	1	1	1	1	1	1
Delta $_{max}$ (degree)	0	0	0	0	0	0	0
$P_{loss\ max}$ (MW)	0.9	0.89	0.89	0.88	0.89	0.9	0.9
$Q_{loss\ max}$ (MVAr)	3.41	3.37	3.36	2.35	1.91	3.4	3.4
Total loss (MW)	3.76	3.58	3.52	3.49	3.686	3.717	3.711
Number of devices	……..	21	20	18	11	3	3

5 Conclusions

The obtained results show that optimum number of TCSC devices to be inserted in a power system in addition to their locations and compensation levels can be found in the presence of constraints on nodal voltage deviations and thermal capability of transmission lines. The obtained results show also that TCSCs can, when optimally sized and selected, improve system stability by increasing minimum voltage of the system. Optimal locations of the TCSCs can also return the system to its operational limits after exceeding them due to line outages. Finally, proper selection of FACTS devices and their locations can effectively improve the overall system performance.

References

[1] Zhang, X., Rehtanz, C., Pal, B.: Flexible AC Transmission Systems: Modeling and Control. Springer (2006)
[2] Meikandasivam, S., Nema, R.K., Jain, S.K.: Behavioral Study of TCSC Device - A Matlab/Simulink Implementation. World Academy of Science, Engineering and Technology 45 (2008)
[3] Gerbex, S., Cherkaoui, R., Germond, A.J.: Optimal Location of Multi-Type FACTS Devices in a Power System by Means of Genetic Algorithms. IEEE Trans. Power Syst. 16(3), 537–544 (2001)
[4] Banu, R.N., Devaraj, D.: Genetic Algorithm Approach for Optimal Power Flow with FACTS Devices. In: 4th International IEEE Conference Intelligent Systems (September 2008)
[5] Saravanan, M., Slochanal, S., Venkatesh, P., Abraham, J.: Application of Particle Swarm Optimization Technique for Optimal Location of FACTS Devices Considering Cost of Installation and System Loadability. Electric Power Syst. Res. 77(3/4), 276–283 (2007)
[6] Cai, L., Erlich, I., Stamtsis, G.: Optimal Choice and Allocation of FACTS Devices in Deregulated Electricity Market Using Genetic Algorithms. In: Proceeding of the IEEE PES General Meeting, pp. 201–207 (2004)
[7] Feng, W., Shrestha, G.: Allocation of TCSC Devices to Optimize Total Transmission Capacity in a Competitive Power Market. In: Proceedings of the IEEE Power Engineering Society Transmission and Distribution Conference (Winter Meeting), Columbus, OH, pp. 587–593 (January 2001)
[8] Kazemi, A., Shahnazari, M., Naghshbandi, A.: A Genetic Algorithm Based Approach to Allocation of SVC Considering System Loadability. In: Proceedings of the 41st International Universities Power Engineering Conference, UPEC (September 2006)
[9] Mahdad, B., Bouktir, T., Srairi, K.: Strategy of Location and Control of FACTS Devices for Enhancing Power Quality. In: IEEE MELECON, Benalmádena, Málaga, Spain (May 2006)
[10] Rashed, G., Shaheen, H., Cheng, S.: Optimal Location and Parameter Setting of TCSC by Both Genetic Algorithm and Particle Swarm Optimization. In: Second IEEE Conference on Industrial Electronics and Applications, Art. No. 4318586, pp. 1141–1147 (2007)
[11] Sait, S., Youssef, H.: Iterative Computer Algorithms with Application in Engineering: Solving Combinatorial Optimization Problems. IEEE Computer Society (1999)
[12] Goldberg, D.E.: Genetic Algorithms in Search Optimization and Machine Learning. Addison-Wesley Publishing Company Inc. (1989)

[13] Zimmerman, R.D., Murillo-Sanchez, E.C.: Matpower A Matlab™ Power System Simulation Package Version 3.2, User's Manual (September 21, 2007)
[14] Yang, G.Y., Hovland, G., Majumder, R., Dong, Z.Y.: TCSC Allocation based on Line Flow Based Equations Via Mixed-Integer Programming. IEEE Trans. Power Syst. 22(4), 2262–2269 (2007)
[15] Abdelaziz, A.Y., El-Sharkawy, M.A., Attia, M.A.: Optimal Location of Thyristor-Controlled Series Compensators in Power Systems for Increasing Loadability by Genetic Algorithm. Electric Power Components and Systems 39(13), 1373–1387 (2011)

Wavelet-ANN Model for Nutrient Load Predictions in Rivers

Raj Mohan Singh

Department of Civil Engineering, Motilal Nehru National Institute of Technology, Allahabad-211004, India
rajm@mnnit.ac.in, rajm.mnnit@gmail.com

Abstract. Rise in nutrients concentrations have been a persistent problem in streams and rivers throughout the world. Estimates of nutrient fluxes are necessity as well as challenges for water quality management. The observation of nutrient loads (of nitrogen and/or phosphorous) from watershed into river or stream system is not straight forward but complex function of hydrology, geology, and land use of the region. There are statistical approaches to predict the nutrients loads in rivers. Development of models based on temporal observations may improve understanding the underlying hydrological processes complex phenomena of nutrient concentrations in river. Present work utilized temporal patterns extracted from temporal observations of monthly flow data and nitrogen loads using wavelet theory. These patterns are then utilized by an artificial neural network (ANN). The wavelet-ANN conjunction model is then able to predict the monthly nutrient load.

Keywords: Wavelet-ANN, Time series modeling, Wavelet transforms, Nutrients, Nitrogen load prediction.

1 Introduction

Rivers have been central to the growth of societies throughout recorded history. Their fertile floodplains produced, and continue to produce, high yields from agricultural crops. The rivers yielded high harvests of fish, and river ecologists have learned that fish production, too, is linked to the productive floodplain [1]. Both nitrogen and phosphorus have complex cycles that are mediated by physical, chemical, and biotic processes in the water and in the soil [2].These processes are modulated by temperature and water conditions. Therefore, nitrogen and phosphorus cycles are expected to be affected by the timing of floodplain inundation. Excessive addition of nutrients, usually nitrogen and phosphorus (N and P), to natural water is usually refers as eutrophication [3].

Rise in nutrient concentrations can result in locally high algal, phytoplankton, and macrophyte biomass that have negative impacts on aquatic habitat and biota. Additionally, the transport of these nutrients can lead to degradation of downstream water bodies, such as lakes, reservoirs, and estuaries [4]. Noteworthy examples of problems associated with elevated nutrient concentrations vary from eutrophication of

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 558–565, 2012.

lakes and reservoirs to hypoxia (low concentrations or absence of dissolved oxygen) [5].

Nutrient enrichment results in the excessive growth of plants including phytoplankton in surface waters. In recent years, eutrophication has been recognized as an important issue for environmental concern. It becomes one of the most serious water pollution problems [6,7]. Eutrophication can stimulate rapid algal growth, resulting in massive algal glooms. High biomass during algal blooms may initiate significant ecological problems and cause harmful effects in the biota of the region [8]. Some typical problems include anoxia condition of the water, change in bio-community and food-web, deterioration of the water quality and adversary effects on the recreational use of the water. Anoxia or hypoxia induced by algal blooms would cause declines in fishery production, major changes in species composition, and distortions of sex ratio of some fish [9]. In the water supply source the organic matter excreted from algae cannot be effectively removed by enhanced coagulation and filtration, and hence it is difficult to control the formation of disinfection by-products in drinking water [10]

The gradual accumulation of water quality data records over the past few decades has increased the value of these data for examining long-term trends. However, on many major rivers of different countries, infrequent sampling of most pollutants makes flux estimates and their analysis difficult. The present work utilized the monthly data for the Idaho River to estimate the nitrogen loads using Wavelet ANN conjunction model.

Transport of nutrient into river or streams is complex function of hydrology of the region and land use patterns in a given river or stream basin which are difficult to quantify accurately. Development of models based on temporal observations may improve understanding the underlying hydrological processes in such complex phenomena. Recently, artificial neural network (ANN) as a non-linear inter-extrapolator is extensively used by hydrologists [11]. ANNs may not be able to cope with nonstationary data if preprocessing of the input and/or output data is not performed. Therefore, in this study, a new combined model was developed for monthly nitrogen load prediction based on wavelet decomposition and ANN techniques. The aim of combining the wavelet and ANN models is to improve the accuracy of nutrient load prediction.

Present work utilized temporal patterns extracted from temporal observations of monthly nutrient load series using wavelet theory. These patterns are then utilized by an artificial neural network (ANN). The wavelet-ANN conjunction model is then utilized to predict the monthly nitrogen load in a stream. The application of the proposed methodology is illustrated with real data pertaining to Indian River system.

2 Artificial Neural Network

The ANN is a broad term covering a large variety of network architecture, the most common of which is a multilayer perceptron feedforwrd network with back propagation algorithm [12]. There is no definite formula that can be used to calculate

the number of hidden layer(s) and number of nodes in the hidden layer(s) before the training starts, and usually determined by trial-and-error experimentation. The back propagation algorithm is used for training of the feed forward multi-layer perceptron using gradient descent [13, 14]. Present paper utilized Levenberg- Marquardt (LM) backpropagation algorithm to optimize the weights and biases in the network. LM algorithm is more powerful and faster than the conventional gradient descent technique [15, 16]. Basics and details of ANN are available in literature [17].

3 Wavelet Analysis

Wavelet analysis allows the use of long-time intervals for low frequency information and shorter intervals for high frequency information. Wavelet analysis is capable of revealing aspects of original data like trends, breakdown points, and discontinuities that other signal analysis techniques might miss. Furthermore, it can often compress or denoise a signal. Basics of wavelets are available in literature [18, 19, 20]. Discrete wavelet transform (DWT) operates two sets of function (scaling and wavelets) viewed as high-pass and low-pass filters (Fig. 1). The original time series are passed through high-pass and low-pass filters and separated at different scales. The time series is decomposed into one comprising its trend the approximation and one comprising the high frequencies and the fast events (the detail). In the present study, the detail coefficients (D) and approximation (*A*) subtime series are obtained using MATLAB wavelet tool box [21].

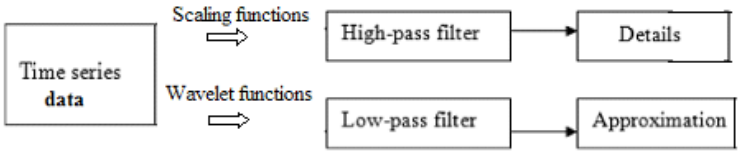


Fig. 1. Decompositions of time series data into DWT components

Wavelet function $\psi(t)$ is called the mother wavelet, can be defined as:

$$\int_{-\infty}^{\infty} \psi(t)\, dt = 0 \tag{1}$$

Mathematically, the time-scale wavelet transform of a continuous time signal, *f(t)*, is defined as

$$W_{a,b}\,(t) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} \psi * \left(\frac{(t-b)}{a}\right) f(t)\, dt \tag{2}$$

where * corresponds to the complex conjugate of ψ; $W_{a,b}\,(t)$ = successive wavelet and presents a two-dimensional picture of wavelet power under a different scale; a=scale or frequency factor, b=position or time factor; $a \in R, b \in R, a \neq 0$ and R= domain of real number.

Let $a = a_0{}^{j}$, $b=kb_0a_0{}^{j}$, $a_0>1$, $b_0 \in R$; j, k are integer numbers that control the wavelet dilation and translation respectively; a_0 is a specified fined dilation step greater then 1; and b_0 is the location parameter and must be greater than zero. The discrete wavelet transform (DWT) of $f(t)$ can be written as:

$$W(a,b) = a_0{}^{-\frac{j}{2}} \int_{-\infty}^{\infty} \psi^*(a_0{}^{-\frac{j}{2}} - kb_0) f(t) d \tag{3}$$

Now, the original time series may be represented (reconstructed) as:

$$f_i = \bar{C} + \sum_{j=1}^{J} \sum_{k=0}^{2^{J-k}-1} W(a,b)_D \tag{4}$$

It may be further simplified as:

$$f_i = \bar{C} + \sum_{j=1}^{J} \sum_{k=0}^{2^{J-k}-1} W(a,b)_D \tag{5}$$

where first term, C, is called approximation sub-signal at level J and second term, $W_j(t)$, are details subsignals (low scale, high frequency) at levels j = 1, 2, . . .,J.

Wavelet-ANN model is assumed to perform satisfactory when its performance evaluation statistics is not improving or constant. Developed methodology is implemented in MATLAB 7.0 platform. Model evaluations criteria are utilized to judge the predictive capability of the best performing ANN models.

4 Wavelet-ANN Conjunction Model for Nitrogen Nutrients

Wavelet-ANN conjunction model utilized wavelet decomposed coefficients ob-tained from wavelet analysis to the ANN technique for daily sediment load prediction as shown in Fig.2.

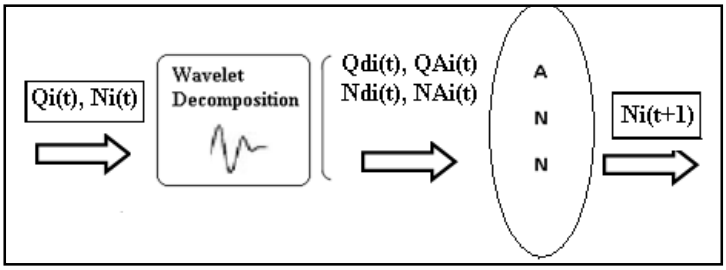


Fig. 2. Wavelet-ANN conjunction Model for Nitrogen Nutrients

First, the measured daily time series, Q (discharge) (m3/s) and/or N(Nitrogen nutrients, NO_2 +NO_3) (mg/l) were decomposed into smulti-frequency time series comprising of details (low scale, high frequency) - Qd1(t); Qd2(t);…; Qdi(t) for discharge and Nd1(t); Nd2(t);…; Ndi(t); and approximate (high scale, low frequency) – QA(t) for discharge and NA(t) by DWT. The representation di presents the level 'i' decomposed details time series and 'A' denotes approxi-mation time series. The

decomposed Q(t) and N(t) time series were utilized as in-puts to the ANN model and the original time series of observed suspended sedi-ment load at the next step N(t+1) is output to the Wavelet-ANN model. The schematic representation of methodology is presented in Fig. 2.

5 Model Evaluation Criteria

The performances of the developed models are evaluated based on some performance indices in both training and testing set. Varieties of performance evaluation criteria are available which could be used for evaluation and inter comparison of different models. Following four performance indices represented as correlation coefficient (R), root mean square error (RMSE), model efficiency (Nash–Sutcliffe Coefficient, ME_{Nash}) [22], and Index of agreement (IOA) are selected in this study based on relevance to the evaluation process.

$$R = \frac{\sum_{i=1}^{n}(Xai - \overline{X}ai)(Xpi - \overline{X}pi)}{\sqrt{\sum_{i=1}^{n}(Xai - \overline{X}ai)^2 \sum_{i=1}^{n}(Xpi - \overline{X}pi)^2}} \tag{6}$$

$$RMSE = \sqrt{\frac{1}{n}\left(\sum_{i=1}^{n}(Xai - Xpi)^2\right)} \tag{7}$$

$$\mathrm{ME}_{\mathrm{Nash}}(E) = 1.0 - \frac{\sum_{i=1}^{n}(Xa_i - X_{pi})^2}{\sum_{i=1}^{n}(X_{ai} - \overline{X}ai)^2} \tag{8}$$

$$\mathrm{IOA} = 1.0 - \frac{\sum_{i=1}^{n}(Xa_i - X_{pi})^2}{\sum_{i=1}^{n}\left[\left|(X_{ai} - \overline{X}ai)\right| + \left|(X_{pi} - \overline{X}pi)\right|\right]^2} \tag{9}$$

where *Xai* and *Xpi* are measured and computed values of atrazine concentration values in streams; $\overline{X}ai$ and $\overline{X}pi$ are average values of *Xai* and *Xpi* values respectively; *i* represents index number and *n* is the total number of years of measurement.

6 Application Wavelet-ANN Conjunction Model

The developed models require uninterrupted time series data pertaining to flow and Nitrogen load at a gauging station for calibration and verification periods. The data derived from the Iowa River at the Wapello, IA site (USGS Station Number 05465500; basin area (sq mi): 12,499; latitude: 41°10′48″; longitude: 091°10′57″)

were employed to train and test all the models developed in this study. The monthly time series of Q and N (NO2 +NO3) for this station were downloaded from the USGS web server (http://toxics.usgs.gov/pubs/of-2007-1080/sub_basins/IOWAWAPE. Monthly. xls). Monthly data from October 1978 to September 1997 (20 yrs) and the data from October 1997 to September 2005 (8 years) were used as training and testing sets, respectively. The actual monthly time series data at the site related to discharge and nitrogen load employed for wavelet transform and performance evaluations of the methodology are shown in Fig. 3 and Fig. 4 respectively.

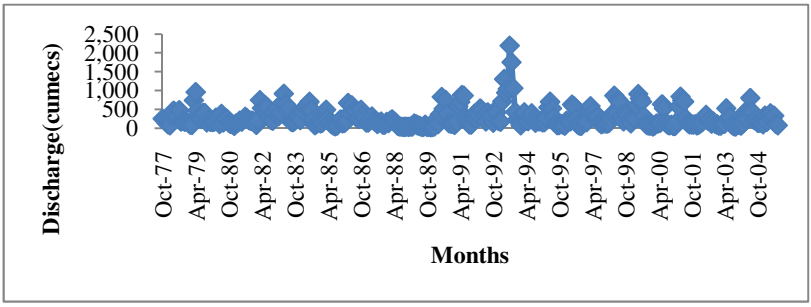


Fig. 3. Monthly discharge values

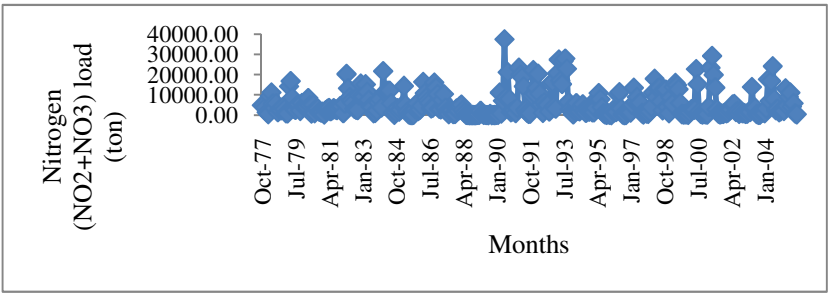


Fig. 4. Monthly nitrogen nutrient (NO2+NO3) values

7 Results and Discussions

Wavelet-ANN conjunction model is implemented on MATLAB platform. Back propagation ANN training algorithm is implemented for obtaining the optimal ANN architecture with the internal parameters as: number of epoch=1000; momentum coefficient=0.8. In addition, optimum combinations of transfer functions in the hidden and output layer are obtained with 'trainlm' function for Levenberg-Marquardt (LM). Performance of Wavelet–ANN (WANN) conjunction model is compared with actual time series (without wavelet decomposition) ANN model (TANN). Two WANN models, WANN 1 and WANN 2 are developed. WANN-1 model has two inputs (wavelet decomposition-approximation and details coefficients of discharge data at time t by wavelet transform DWT Haar wavelet level 1). The

output for WANN 1 model is Nitrogen concentration in the river at time t+1. The WANN-2 model has four inputs (wavelet decomposition-approximation and details coefficients of both discharge and nitrogen concentration data at time t by wavelet transform DWT Haar wavelet level 1). The output is same as WANN1 model. Two TANN models, TANN 1 and TANN 2, are also developed. TANN 1 model has one in-put i.e. actual discharge at time t and one output (nitrogen load at time t+1. The TANN 2 model has two inputs i.e. both actual discharge and nitrogen load at time t. The output is same as TANN 1 model. Thus, output of all the models is same.

Experimentation with varying number of hidden nodes and training algorithm are performed. The error statistics of best performing WANN and TANN models in training and testing are shown in Table 1. When both discharge and sediment data are employed in nitrogen prediction, results improve considerably. WANN2 model (4-3-1) outperforms TANN2 (2-2-1) as evident from performance statistics presented in Table 1.

Table 1. Training and testing errors for WANN conjunction model and TANN

Models (Trainlm)	Performance statistics				
	Training /Testing	R	RMSE	E	IOA
WANN 1 2-2-1	Training	0.671	4.6E+03	0.450	0.777
	Testing	0.578	5.4E+03	0.326	0.700
WANN 2 4-3-1	Training	0.788	3.8E+03	0.622	0.871
	Testing	0.724	4.6E+03	0.515	0.806
TANN 1 1-2-1	Training	0.556	5.18+03	0.309	0.669
	Testing	0.498	5.77E+03	0.238	0.596
TANN 2 2-2-1	Training	0.651	4.7E+03	0.424	0.762
	Testing	0.612	5.2E+03	0.370	0.706

8 Conclusions

The study presents the general framework for evaluating nutrients load in a river system. Methodology for wavelet-ANN conjunction model for nitrogen load prediction in rivers is demonstrated through illustrative real monthly discharge and nitrogen concentration data. The values of statistical performance evaluation criteria indicate the WANN time series model is able to simulate the complex nitrogen transport event in rivers. Wavelet decomposition improved the results considerably. Multi levels of wavelet decomposition may further improve the testing results.

References

1. Junk, W.J., Bayley, P.B., Sparks, R.E.: The flood pulse concept in river-floodplain ecosystems. In: Dodge, D.P. (ed.) Proceedings of the International Large Rivers Symposium, pp. 110–127. Canadian Journal of Fisheries and Aquatic Sciences Special Publication 106, Ottawa (1989)

2. Brady, N.C., Weil, R.R.: The Nature and Properties of Soils, 14th edn. Pearson, Columbus (2008)
3. Jorgensen, B.B., Richardson, K.: Eutrophiciation in Coastal Marine Ecosystems. American Geophysical Union, USA (1996)
4. USEPA (U.S. Environmental Protection Agency), Nutrient Criteria Technical Guidance Manual—Lakes and Reservoirs: Office of Water, Office of Science and Technology, EPA-822-B-00-001, Washington, D.C. (2000)
5. Saad, D.A., Schwarz, G.E., Robertson, D.M., Booth, N.L.: A multi-agency nutrient dataset used to estimate loads, improve monitoring design, and calibrate regional nutrient sparrow models. Journal of the American Water Resources Association (JAWRA) 47(5), 933–949 (2002)
6. Lee, J.H.W., Arega, F.: Eutrophication Dynamics of Tolo Harbour, Hong Kong. Marine Pollution Bulletin 39(1-2), 187–192 (1999)
7. Su, J.L., Dong, L.X.: Application of Numerical Models in Marine Pollution Research in China. Marine Pollution Bulletin 39(1-12), 73–79 (1999)
8. Scholten, M.C.T., Foekema, E.M., Dokkum, H.P.V., Kaag, N.H.B.M., Jak, R.G.: Eutrophication Management and Ecotoxicology. Springer, Heidelberg (2005)
9. Shang, E.H.H., Yu, R.M.K., Wu, R.S.S.: Hypoxia Affects Sex Differentiation and Development, Leading to a Male-Dominated Population in Zebrafish (Danio rerio). Environmental Science and Technology 40, 3118–3122 (2006)
10. Cheng, W.P., Chi, F.H.: Influence of eutrophication on the coagulation efficiency in reservoir water. Chemosphere 53, 773–778 (2003)
11. Nourani, V., Komasi, M., Mano, A.: A Multivariate ANN-Wavelet Approach for Rainfall–Runoff Modeling. Water Resour. Manage. 23, 2877–2894 (2009)
12. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representation by error propagation. In: Parallel Distributed Processing, vol. 1, pp. 318–362. MIT Press, Cambridge (1986)
13. Bishop, C.M.: Neural Networks For Pattern Recognition. Oxford University Press, India (1995)
14. Singh, R.M., Datta, B., Jain, A.: Identification of unknown groundwater pollution sources using artificial neural network. Journal of Water Resources Planning and Management, ASCE 130(6), 506–514 (2004)
15. Hagan, M.T., Menhaj, M.B.: Training feed forward networks with the Marquaradt algorithm. IEEE Trans. Neural Netw. 6, 861–867 (1994)
16. Kisi, O.: Streamflow forecasting using different artificial neural network algorithms. J. Hydrol. Eng. 12(5), 532–539 (2007)
17. Haykin, S.: Neural networks: A comprehensive foundation, p. 696. Mac-Millan, New York (1994)
18. Daubechies, I.: Orthonormal bases of compactly supported wavelets. Commun. Pure and Appllied Mathmatics XLI, 901–996 (1988)
19. Kang, S., Lin, H.: Wavelet analysis of hydrological and water quality signals in an agricultural watershed. Journal of Hydrology 338, 1–14 (2007)
20. Mallat, S.G.: A wavelet tour of signal processing. Academic, San Diego (1998)
21. MATLAB, The Language of Technical Computing. The Math Works Inc., Natick (2004)
22. Nash, J.E., Sutcliffe, J.V.: River flow forecasting through conceptual models. Part 1-A: Discussion principles. Journal of Hydrology (1970)

Performance Analysis and Design of Proportional Integral Derivative Controlled Automatic Voltage Regulator System Using Local Unimodal Sampling Optimization Technique

Pradeep Kumar Mohanty[1,*], Binod Kumar Sahu[1],
Sidhartha Panda[2], Sanjeeb Kumar Kar[1], and Nandan Mishra[1]

[1] Depatment of Electrical Engg. , Institute of Technical Education and Research,
S'o'A University, Bhubaneswar, India
{pkmohanty68,binoda,skkariter}@rediffmail.com,
aksanebs@gmail.com
[2] Department of Electrical & Electronics Engg., VSSUT, Burla, Burla, Odisha, India
panda_sidhartha@rediffmail.com

Abstract. Proportional–Integral–Derivative (PID) controllers are mainly used in industrial control systems, motor drives, process control, and instrumentation. Optimum tuning of PID controller parameters is a big challenge for researchers and plant operators. This paper describes the 'Local Unimodal Sampling (LUS)' algorithm to tune the PID controller for an Automatic Voltage Regulator (AVR) system to determine the optimum controller parameters. Many optimization techniques use local sampling with a fixed sampling range. Therefore, there is a risk of getting stuck in the local optima. This problem can be overcome by using LUS algorithm which decreases the sampling range as optimization progresses. Compared with the artificial bee colony (ABC), the proposed method is found to be more robust and efficient in improving the step response of an AVR system. Transient response analysis, root locus analysis and bode analysis are used to compare the performance of both the algorithms.

Keywords: Automatic Voltage Regulator (AVR), Proportional Integral Derivate (PID) controller, Local Unimodal Sampling (LUS), transient response analysis, robustness analysis.

1 Introduction

Most important factors of an electric power generating system are the quality and stability of electric power. To maintain the power system parameter such as voltage, frequency, oscillation etc., at their nominal values, control system plays a very important and distinct role. In power system AVR plays an important role in maintaining the terminal voltage of the alternator at its rated value. The AVR of an

* Corresponding author.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 566–576, 2012.

alternator controls the terminal voltage and reactive power simultaneously. It also helps in proper sharing of the reactive power amongst all the parallel connected alternators. The real power loss in the transmission lines depend on the flow of reactive power which depends upon system voltage. The real power loss in transmission lines can be minimised by controlling the voltage level of the system. Therefore, the AVR must be properly designed by taking both normal and abnormal conditions (e.g. faults) in to account. AVR with fixed-gain decreases the performance in abnormal conditions making the AVR to introduce negative damping and there by degrading the stability of the system [1].

Various control techniques such as neural control, adaptive control and fuzzy control have been studied [2]–[6]. But the best controller technique is the PID controller as it has a wide range of applications in instrumentation industries and in process control because of its easy installation, simple in structure and robust in performance. A PID controller consists of three control parameters i.e. proportional gain, integrating gain, derivative gain respectively. However, due to the significant changes in the daily load cycle, fixed gain PID controlled AVR system will fail to provide its best performance under wide range of operating conditions. It has been very difficult to properly tune the gains of PID controllers, because most of the components of industrial plants are of nonlinear in nature having higher order and time delays. Many stochastic optimization techniques have been proposed to determine the optimum parameters of PID controllers used in AVR system to improve the controller performance over a wide range of operating conditions. Stochastic techniques such as artificial bee colony (ABC), particle swarm optimization (PSO) and differential evolution (DE) have been applied to properly tune the PID controller parameters [7]. One of the modern heuristic optimization techniques is the optimization based on LUS algorithm.

Especially Pattern Search (PS) [8] and LUS are preferred for short optimization processes. LUS algorithm is an extension of the PS algorithm. In this paper, LUS algorithm is implemented to tune the PID controller in AVR. It was introduced in 2008 by Pedersen [9]. LUS algorithm samples all dimensions while decreasing its sampling range simultaneously in a similar manner as that of PS. The sampling range is reduced in the process of optimization as a fixed sampling range has a risk of convergence to local optima. In LUS algorithm, the search range is exponentially decreased if the samples fail to improve on the fitness of the current position. In this paper LUS algorithm is used to optimize the gains of PID controller. The objective of this paper is to design an efficient LUS-PID controller to control the magnitude of the voltage intelligently in an autonomous power generating system. The proposed algorithm reveals better performance than ABC algorithm [7].

The paper is organised as follows: Description of AVR model in 2, PID controller in 3, LUS algorithm in section 4, simulation results and description in 5 and finally the conclusion in section 6.

2 Description of AVR Model

The excitation control of synchronous generator plays a vital role for improvement of stability and quality of electric power system. An AVR system consists of four basic components such as amplifier, exciter, generator, and sensor. Transfer functions of the

above components are assumed to be linear for analysing the mathematical model of AVR system. The transfer functions [10], range of various gains, time constants and exact values of different parameters [7] of all the four components are depicted in table 1.

Table 1. Parameters used in the AVR system [7]

Elements	Transfer function	Range of Gains	Range of time constants	Gain	Time Constants
Amplifier	$TF_A = \frac{K_A}{1+sT_A}$	$10 \le K_a \le 40$	$0.02 \le T_a \le 0.1$	$K_a = 10$	$T_a = 0.1$
Exciter	$TF_E = \frac{K_E}{1+sT_E}$	$1 \le K_e \le 10$	$0.4 \le T_e \le 1.0$	$K_e = 1.0$	$T_e = 0.4$
Generator	$TF_G = \frac{K_G}{1+sT_G}$	$0.7 \le K_g \le 1.0$	$1.0 \le T_g \le 2$	$K_g = 1.0$	$T_g = 1.0$
Sensor	$TF_S = \frac{K_S}{1+sT_S}$	$K_s = 1.0$	$0.001 \le T_s \le 0.06$	$K_s = 1.0$	$T_s = 0.01$

Fig. 1. shows the transfer function model of an AVR system without PID controller.

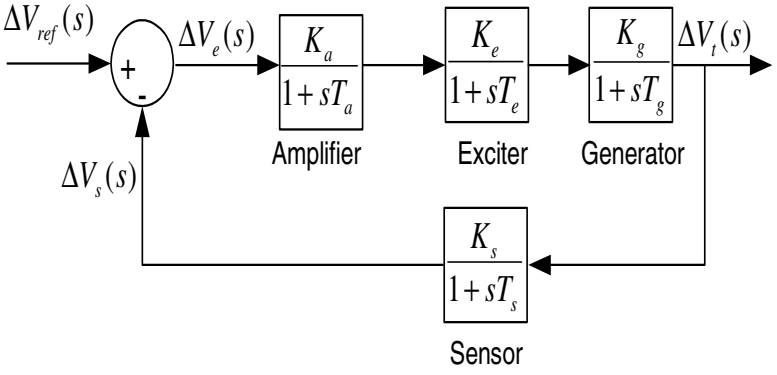


Fig. 1. Transfer function model of AVR system (without PID controller)

The transfer function of the above AVR system (without PID controller) is given by

$$\frac{\Delta V_t(s)}{\Delta V_{ref}(s)} = \frac{0.1s + 10}{0.0004s^4 + 0.0454s^3 + 0.555s^2 + 1.51s + 11} \tag{1}$$

The response of the AVR without PID controller is shown in Fig.2. The system is stable but oscillating in nature. The closed loop transfer function has two real poles at (-100, 0) and (-12.5, 0) and two complex poles at (-0.52 + 4.66i) and (-0.52 - 4.66i). The terminal voltage reaches a steady value of 0.91 pu at about 10 sec which is not acceptable. The response of the AVR system can be improved by using PID controller. An AVR system with PID controller tuned by LUS algorithm is shown in

Fig.3.With PID controller as shown in Fig.3, the transfer function of AVR system becomes

$$\frac{\Delta V_t(s)}{\Delta V_{ref}(s)} = \frac{0.1K_d s^3 + (0.1K_p + 10K_d)s^2 + (0.1K_i + 10K_p)s + 10K_i}{0.0004s^5 + 0.0454s^4 + 0.555s^3 + (1.51 + 10K_d)s^2 + (1 + 10K_p)s + 10K_i} \quad (2)$$

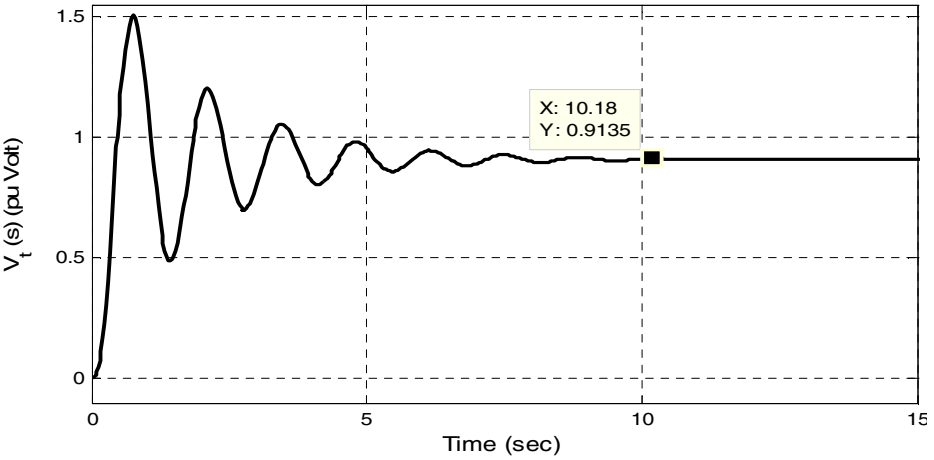


Fig. 2. Unit step response of AVR system without PID controller

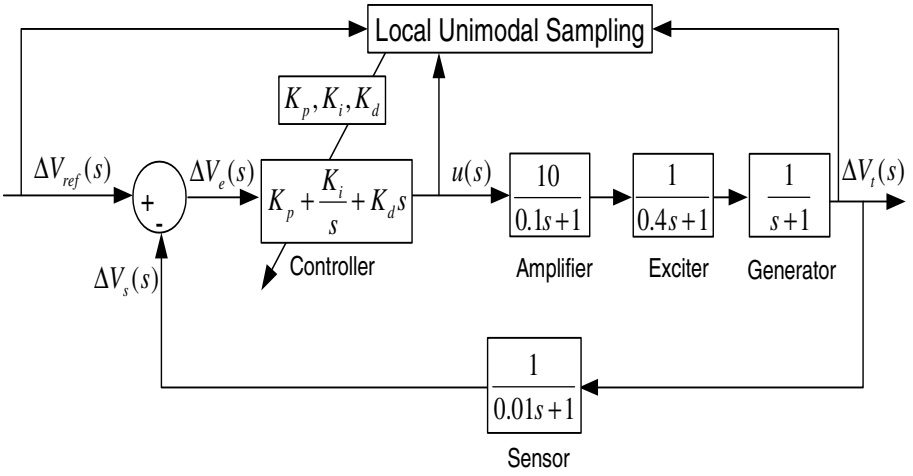


Fig. 3. Transfer function model of AVR system with PID controller tuned by LUS algorithm

3 Proportional Integral Derivative (PID) Controller

The PID controller consists of three basic modes, the proportional, the integral and the derivative modes. The proportional controller reduces the rise time (t_r), but cannot eliminate the steady-state error. An integral controller eliminates the steady-state

error, but worsens the transient response. The derivative controller increases the stability of the system by reducing overshoot and thereby improving the transient response [11]. To design the PID controller three basic required parameters are Proportional gain (K_p), Integral gain (K_i) and Derivative gain (K_d). Transfer function of PID controller in Laplace domain is given by

$$TF_{PID} = K_p + \frac{K_i}{s} + K_d s \tag{3}$$

To properly design the parameters of the PID controller, a suitable tuning algorithm must be adopted in order to improve the transient response and minimise the steady-state error. In this paper LUS algorithm is applied to tune the parameters of PID controller.

4 Local Unimodal Sampling (LUS) Algorithm Used to Tune the Parameters

Throughout the optimization, hill climber (HC) algorithm [12] and simulated annealing SA algorithm [13] uses local sampling with and a fixed sampling range. Therefore, there is a risk factor of getting stuck in local optima. This problem can be alleviated by using Local Unimodal Sampling (LUS) algorithm [9] which decreases the sampling range during optimization.

The initial current position $\vec{x} \in R^n$ is randomly selected from the search-space where n is the dimension of search space. The new position $\vec{y}$ is sampled from the neighbourhood of x. $(-\vec{d}, \vec{d})$ is taken as the initial sampling range, where $\vec{d} = \vec{B}_{up} - \vec{B}_{low}$. $\vec{B}_{up}$ is the upper boundaries and $\vec{B}_{low}$ is the lower boundaries of the entire search space. The sampling rate is decreased by multiplying a factor Q, on every failure of $\vec{y}$ in order to improve the fitness of $\vec{x}$.

Therefore, the new sampling range becomes $\vec{d} = Q \times \vec{d}$. Where Q = $\sqrt[\gamma n]{1/2}$ is known as the decrease factor.

In the above expression, n is the dimension of the search space and γ is the user defined parameter which governs the behaviour of the LUS algorithm. Usually, the value of γ is taken as 3 in many optimization algorithms. If the fitness is improved, the LUS algorithm moves from its old position $\vec{x}$ to a new position $\vec{y}$.

Different steps involved in LUS algorithm are

1. Randomly initialize the current position $\vec{x}$ in the search space.
2. Set the initial search range $\vec{d}$ over the entire search space:
$$\vec{d} \leftarrow \vec{B}_{up} - \vec{B}_{low}$$
3. Repeat the following steps until the fitness threshold is achieved.
   i. Randomly select a vector $\vec{a} \sim U(-\vec{d}, \vec{d})$
   ii. The new position $\vec{y} = \vec{x} + \vec{a}$
   iii. If the fitness at position $\vec{y}$ is less than the fitness at position $\vec{x}$, update the position $\vec{x} \leftarrow \vec{y}$
   iv. Else decrease the search range $\vec{d}$ as $\vec{d}_{new} = Q \times \vec{d}$

5 Simulation Results and Discussion

A. Application of LUS Algorithm

LUS algorithm is used to tune the PID controller parameters. Fig.3 shows the block diagram representation of an AVR system in which LUS algorithm is used for tuning of PID controller parameters. The lower and upper limits of the controller parameters are selected in the range of 0.2 to 2.0 [7]. The integral time absolute error (*ITAE*), integral absolute error (*IAE*), integral time square error (ITSE) and integral square error (ISE) functions are taken as objective functions in order to determine the optimum values of the controller gains. The expressions for different objective functions are given below.

$$ITAE = \int_0^t t|V_r - V_t|dt \tag{4}$$

$$IAE = \int_0^t |V_r - V_t|dt \tag{5}$$

$$ITSE = \int_0^t t|V_r - V_t|^2 dt \tag{6}$$

$$ISE = \int_0^t |V_r - V_t|^2 dt \tag{7}$$

Where V_r is the reference voltage, V_t is the terminal voltage and t is the time range of simulation. The optimization process is run 20 times for each objective function and the best PID controller parameters corresponding to the minimum fitness value among 20 runs are given in table 2 for each objective functions.

Table 2. Parameters of PID controller with different objective functions using LUS algorithm

Parameters	ITAE	IAE	ITSE	ISE
K_p	**0.8463**	1.4527	1.3614	1.2587
K_i	**0.5860**	0.9981	0.9995	0.9990
K_d	**0.2643**	0.6132	0.6860	0.9997

B. Effect of Objective Function

Simulation studies are carried out in order to compare the performance of the PID controller using various objective functions. The different error functions and respective gains are depicted in table 2. Results obtained from transient response analysis for different error functions are given in table 3 for comparison. Table 3 also shows the corresponding values with other recently published modern heuristic optimization technique such as the ABC algorithm [7]. It is clear from table 3 that better performance is obtained by LUS algorithm as compared to that of ABC algorithm in terms of maximum overshoot, settling time, peak time and rise time, when IAE and ITSE are used as objective functions. However peak overshoot and settling time are minimum when ITAE is used as objective function, which are the major factors for analysing the stability of a system. Transfer functions of the system according to the above gains corresponding to ITAE using LUS algorithm is given by;

$$\frac{\Delta V_t(s)}{\Delta V_{ref}(s)} = \frac{0.02543s^3 + 2.628s^2 + 8.522s + 5.86}{0.0004s^5 + 0.0454s^4 + 0.555s^3 + 4.053s^2 + 9.463s + 5.86} \tag{8}$$

Table 3. Result of the transient response for different objective functions

Different Algorithms	Maximum overshoot in pu	Settling time (2 % band) in s	Rise time in s	Peak time in s
LUS/ITAE	**1.0776**	**0.6721**	0.2307	0.4773
LUS/IAE	1.2137	0.7047	0.1202	0.2741
LUS/ITSE	1.1783	1.0582	0.1127	0.2473
LUS/ISE	1.2753	1.3586	**0.0877**	**0.2075**
ABC/ITSE[7]	1.25	3.19	0.156	0.3628

C. Transient Response Analysis

Fig.4 shows the transient and steady state behaviour of the AVR system for two algorithms. Table 3 shows the comparison study of different algorithms. Taking ITAE as objective function, it is found that the maximum overshoot in LUS algorithm is 16 % less than that of ABC algorithm. The settling time in LUS algorithm is 374.63 % less than that of ABC algorithm. The major factors to compare the stability analysis of a system are maximum overshoot and settling time [11]. The system having minimum overshoot and minimum settling time gives better performance which is satisfied by the AVR system tuned by LUS algorithm.

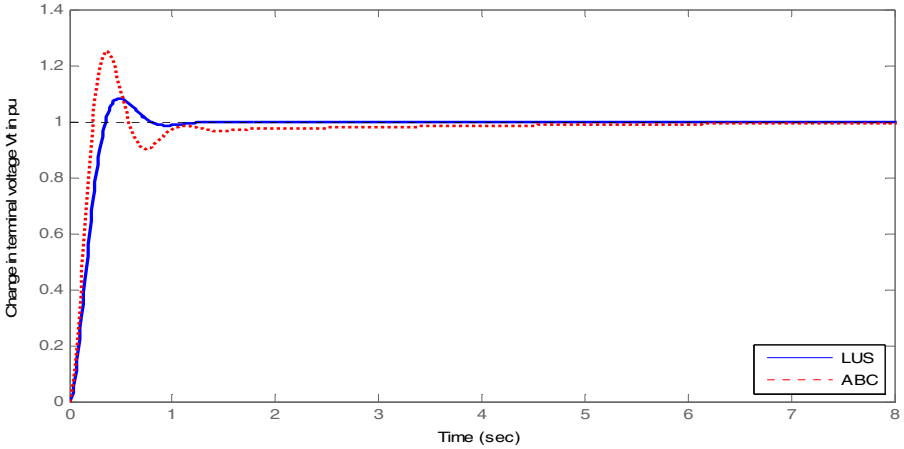


Fig. 4. Terminal Voltage curves of the AVR system for different algorithms

D. Root Locus Analysis

The root locus curves for LUS algorithm is shown in Fig.5. The closed loop poles and damping ratios for both algorithms are given in the Table 4. The AVR systems tuned by both the algorithms are stable, since for all algorithms the closed loop poles lie to left of the s-plane. Compared to ABC algorithm, the conjugate poles obtained by using LUS algorithm are located away from the imaginary axis which improves the performance of the AVR system. Damping ratio of AVR system tuned by LUS algorithm is 46 % more than ABC algorithm.

Table 4. Closed loop poles and damping ratio of the AVR system tuned by LUS, ABC algorithm

LUS		ABC [7]	
Closed loop poles	**Damping ratio**	**Closed loop poles**	**Damping ratio**
-101	1	-100.98	1
-2.09	1	-4.74	1
-1.02	1	-0.25	1
-4.84+6.73i	**0.584**	-3.75+8.4i	0.4
-4.84-6.73i	**0.584**	-3.75-8.4i	0.4

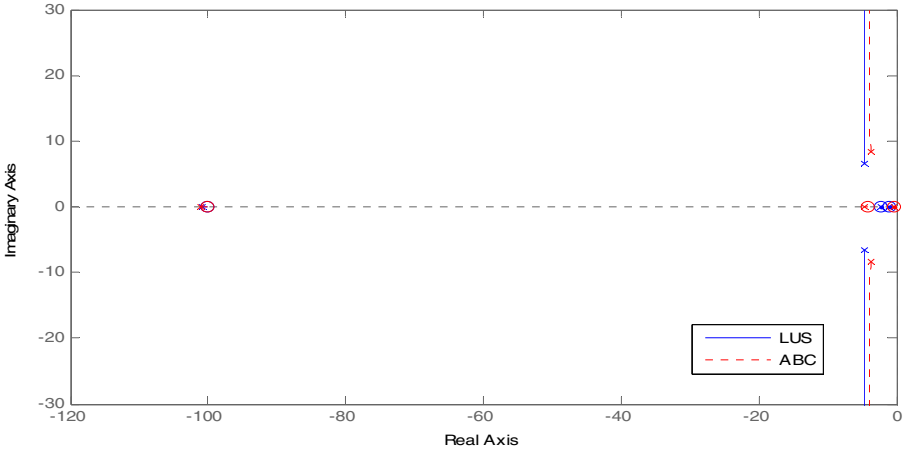


Fig. 5. Root locus curve of the system tuned by LUS algorithm

E. Bode Analysis

Bode plot is an important tool for stability analysis of closed-loop systems [11]. The magnitude and phase plot of the AVR system tuned by LUS algorithm is shown in Fig.6. The peak gain, phase margin, delay margin and bandwidth obtained using two algorithms from bode plots are given in Table 5. The performance of the system is better with minimum peak gain; maximum phase margin; and maximum delay margin. It is observed from Table 5, that AVR system tuned by LUS algorithm gives better frequency response.

Table 5. Bode analysis

Different algorithm	Peak gain (dB)	Phase margin (deg)	Delay margin (sec)	Bandwidth
LUS	**0.131 (4.86 Hz)**	**122**	**0.3721**	9.3862
ABC [7]	2.87 (7.51 Hz)	69.4	0.1109	**12.8791**

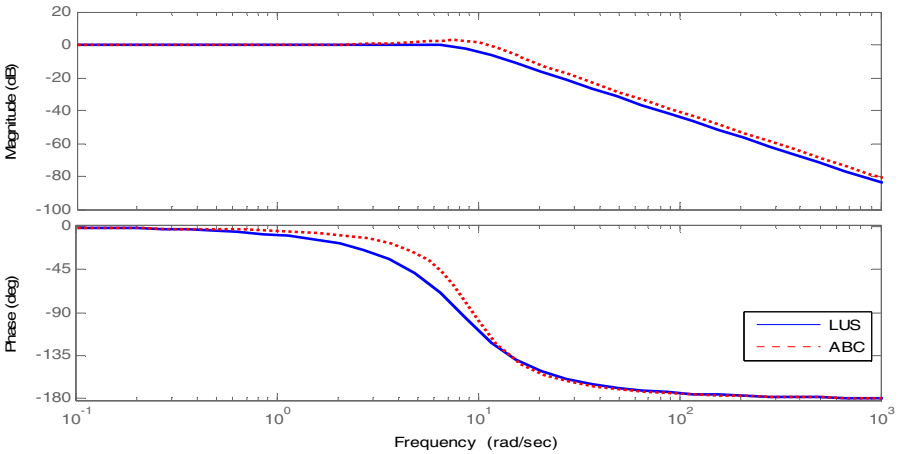


Fig. 6. Bode plot of system tuned by LUS algorithm

F. Robustness Analysis

Robustness of AVR system tuned by LUS algorithm is analysed and depicted in table 6. It is performed by varying the time constants of amplifier, exciter, generator and sensor in the range of -50 % to +50 % in steps of 25%. The range of deviations and percentage of total deviation in peak value, settling time, rise time and peak time are given in table 7. From table 7 it is clear that the deviations are in small range. The average deviation of peak value is 4.16 % and for settling time (2 % band), rise time and peak time are 44.58 %, 15 % and 33.3 % respectively. The ranges of total deviation are within limit, and therefore, AVR system with PID controller tuned by LUS algorithm is robust.

Table 6. Results of robustness analysis of PID controller tuned by LUS algorithm

Parameter	Rate of change (%)	Peak value (pu)	Settling time (2% band) (s)	Rise time (s)	Peak time (s)
T_a	-50	0.9985	0.3989	0.2337	1.7230
	-25	1.0324	0.5429	0.2262	0.4529
	+25	1.1169	1.0965	0.2385	0.5099
	+50	1.1507	1.2327	0.2478	0.5422
T_e	-50	1.0489	1.2019	0.1550	0.2963
	-25	1.0627	0.9785	0.1948	0.3832
	+25	1.0942	0.8993	0.2637	0.5635
	+50	1.1117	1.1649	0.2947	0.6594
T_g	-50	1.1462	1.6023	0.1375	0.2954
	-25	1.1018	1.0260	0.1844	0.3870
	+25	1.0665	0.8844	0.2767	0.5775
	+50	1.0652	1.7483	0.3221	0.7093
T_s	-50	1.0654	0.6678	0.2370	0.4801
	-25	1.0714	0.6700	0.2338	0.4786
	+25	1.0841	0.6739	0.2278	0.4763
	+50	1.0910	0.9463	0.2251	0.4642

Table 7. Range of total deviations and percentage of maximum deviations of the system

Time constants	Parameter	Range of total deviations	Percentage of total deviation (%)
T_a	Peak (pu)	0.1522	6.35
	Settling time (sec)	0.8338	45.47
	Rise time (sec)	0.0216	6.9
	Peak time (sec)	1.2701	72.3
T_e	Peak (pu)	0.0628	3.06
	Settling time (sec)	0.3026	42.38
	Rise time (sec)	0.1397	21.71
	Peak time (sec)	0.3631	27.61
T_g	Peak (pu)	0.0810	6.0
	Settling time (sec)	0.8639	61.5
	Rise time (sec)	0.1846	28.37
	Peak time (sec)	0.4139	32.7
T_s	Peak (pu)	0.0256	1.23
	Settling time (sec)	0.2785	28.97
	Rise time (sec)	0.0119	2.65
	Peak time (sec)	0.0159	0.58

6 Conclusion

Local Unimodal Sampling (LUS) algorithm is applied to tune the gains of PID controller for an AVR system. These parameters are utilized to design the AVR system. The proposed algorithm results better dynamic performance of the AVR system. It is found to be more robust and efficient. The result of the suggested algorithm is compared with that of ABC algorithm. Superior performance, robustness, and efficiency of the LUS algorithm have been studied through step response, root locus and bode plots. The extensive studies carried out clearly prove that the suggested algorithm is an excellent method for tuning the AVR system.

References

1. Kundur, P.: Power System Stability and control, TMH, 8th reprint (2009)
2. Visioli, A.: Tuning of PID controllers with fuzzy logic. Proceedings of Institute of Electrical Engineering. Control Theory Application 148(1), 1–8 (2001)
3. Seng, T.L., Khalid, M.B., Yusof, R.: Tuning of a neuro-fuzzy controller by genetic algorithm. IEEE Trans. System, Man, Cybernetics. B 29, 226–236 (1999)
4. Krohling, R.A., Rey, J.P.: Design of optimal disturbance rejection PID controllers using genetic algorithm. IEEE Trans. Evolutionary Computing 5, 78–82 (2001)
5. Mitsukura, Y., Yamamoto, T., Kaneda, M.: A design of self-tuning PID controllers using a genetic algorithm. In: Proc. Amer. Contr. Conf., San Diego, CA, pp. 1361–1365 (June 1999)
6. Kawabe, T., Tagami, T.: A real coded genetic algorithm or matrix inequality design approach of robust PID controller with two degrees of freedom. In: Proc. 12th IEEE Int. Symposium Intelligent Control, Istanbul, Turkey, pp. 119–124 (July 1997)

7. Gozde, H., Taplamacioglu, M.C.: Comparative Performance Analysis of Artificial Bee Colony Algorithm for Automatic Voltage Regulator (AVR) System. Journal of Franklin Institute 348, 1927–1946 (2011)
8. Hooke, R., Jeeves, T.: Direct Search solution of numerical and statistical problems. Journal of the Association for Computing Machinery (ACM) 8(2), 212–229 (1961)
9. Pedersen, M.E.H., ChipperField, A.J.: Local Unimodal Sampling, Hvass Laboratories Technical Report no. HL0801 (2008)
10. Sadat, H.: Power System Analysis, TMH, 17th reprint (2009)
11. Ogata, K.: Modern Control Engineering, 4th edn. Prentice-Hall Inc., USA (2002)
12. Vaughan, D.E.: Simultaneous Generalized Hill Climbing Algorithms for Addressing Sets of Discrete Optimization Problems, Dissertation submitted to the Faculty of the Virginia Polytechnic Institute and State University in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Industrial and Systems Engineering (2000)
13. Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P.: Optimization by Simulated Annealing. Science 220(4598), 671–680 (1983)

Particle Swarm Optimization Trained Auto Associative Neural Networks Used as Single Class Classifier

Vadlamani Ravi[1,*], Naveen Nekuri[1], and Manideepto Das[2]

[1] Institute for Development and Research in Banking Technology,
Castle Hills Road #1, Masab Tank,
Hyderabad - 500057, AP, India
[2] Computer Science Engineering,
IIT Hyderabad, Hyderbad – 502205, AP, India
vravi@idrbt.ac.in,
{naveen.nekuri,dasmanideepto}@gmail.com

Abstract. We propose the particle swarm optimization (PSO) trained auto associative neural network (AANN) as a single class classifier (PSOAANN). The proposed architecture consists of three layers namely input layer, hidden layer and output layer unlike that of the traditional AANN. The efficacy of the proposed single class classifier is evaluated on bankruptcy prediction datasets namely Spanish banks, Turkish banks, US banks and UK banks; UK credit dataset and the benchmark WBC dataset. PSOAANN achieved better results when compared to Modified Great Deluge Algorithm trained auto associative neural network (MGDAAANN) [1]. It is concluded that PSOAANN as a single class classifier can be used as an effective tool in classifying datasets, where the class of interest (usually the positive class) is either totally missing or disproportionately present in the training data, which is the case in many real life problems for e.g. financial fraud detection.

Keywords: Single class classifier, Auto Associative Neural Networks, Particle swarm optimization, Credit Scoring, Bankruptcy prediction.

1 Introduction

It is well known that almost all the intelligent binary classifiers notwithstanding their diverse backgrounds, do share a common problem in that they are all 'statistical' in nature. In other words, they can be trained best if there is sufficient number of samples in both the classes. However, in many real life scenarios such as bankruptcy prediction, fraud detection, intruder detection, churn prediction, money laundering and medical diagnosis, the samples are disproportionately distributed between positive and negative classes. The problem of interest in always the positive class and that is where the number of samples is very small in proportion to the negative class. This phenomenon is called data imbalance problem. There are some data preparation

* Corresponding author.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 577–584, 2012.

methods such as under-sampling, over-sampling etc. to solve this difficulty. An alternative way to solve this problem is to train single-class classifiers usually with negative class. In literature, Baek and Cho [2] and Pramodh and Ravi [1] proposed two such single-class classifiers with identical auto associative neural network architecture but trained by different training algorithms. Both studies solved bankruptcy prediction problems. Baek and Cho [2] concluded that the proposed neural network outperformed the 2-class neural networks. Pramod and Ravi [1] developed a soft computing framework where a global optimization meta-heuristic viz., MGDA is employed to train an Auto associative neural network. In the current study, we employed the particle swarm optimization (PSO) to train an auto associative neural network with simplified architecture as a single class classifier. In addition, PSO is a population based search technique unlike MGDA, which performs point-based search.

The remainder of the paper is aligned as: brief described about literature review in section 2. Brief introduction of PSO in section 3.In section 4, architecture of PSOAANN is presented. Section 5 presents the training algorithm of PSOAANN. In section 6, results and discussion are presented and finally section 7 conclusion of the paper.

2 Literature Review

In the following, we briefly review the works reported in bankruptcy prediction in banks and firms and credit scoring.

The bankruptcy prediction for financial firms especially banks has been the extensively researched area since late 1960s [4]. Creditors, auditors, stockholders and senior management are all interested in bankruptcy prediction because it affects all of them alike [5]. The most precise way of monitoring banks is by on-site examinations. These examinations are conducted by regulators on a bank's premises every 12–18 months, as mandated by the Federal Deposit Insurance Corporation Improvement Act 1991. Six part rating system are used to indicate the safety and soundness of the institution. This rating, referred to as the CAMELS rating, evaluates banks according to their basic functional areas: ***C****apital adequacy,* ***A****sset quality,* ***M****anagement expertise,* ***E****arnings strength,* ***L****iquidity,* and ***S****ensitivity to market risk*. While CAMELS ratings clearly provide regulators with important information, Cole and Gunther [4] reported that these CAMELS ratings decay rapidly. Fraser [6] noted that banks performed better by holding relatively more securities and fewer loans in their portfolios.

A variety of statistical techniques such as regression analysis, logistic regression have been used to solve the problem of bankruptcy prediction. These techniques typically make use of the company's financial data to predict the financial state of the company (healthy, distressed, high probability of bankruptcy). Altman pioneered the work of using financial ratios and multiple discriminant analysis (MDA) to predict financially distressed firms. However, the usage of MDA or statistical techniques, in general, relies on the restrictive assumption on *linear separability*, *multivariate*

normality and *independence of the predictive variables* [7-9]. Unfortunately, many of the common financial ratios violate these assumptions. Bankruptcy prediction problem for financial firms can also be solved using various other types of classifiers. Tam explored a backpropagation trained neural network (BPNN) for this problem and compared its performance with methods such as *MDA*, logistic regression, k-nearest neighbour (k-NN) method and *ID3* [10]. He concluded that neural network outperformed other prediction techniques. Salchenberger et al. [11] find that the neural network produces fewer or equal number of classification errors for each of the forecast periods compared to the logit model. This conclusion holds for total errors, type I errors, and type II errors. Tam and Kiang [12] found that a neural network performs better than statistical methods and decision trees. As a result, many researchers view the neural network as a better alternative to statistical techniques for bankruptcy prediction**.** Atiya [13] surveyed all the prediction techniques including neural networks applied to the bankruptcy prediction problem and proposed more financial indicators, in addition to the traditions ones, which was used by him in the design of a new neural network model. Shin et al. [14] applied SVM to the problem of corporate bankruptcy prediction. They concluded that SVM outperformed the MLFF-BP in terms of accuracy and generalization, as the training dataset size gets smaller. Canbas et al. [15] proposed a framework for constructing the integrated early warning system (IEWS) for detection of banks with serious problems. A fuzzy rule was proposed by Ravikumar and Ravi [16] based classifier for bankruptcy prediction. They reported that fuzzy rule based classifier outperformed the well-known technique, MLFF-BP in the case of US banks data. Ravi et al. [17] proposed a semi-online training algorithm for the radial basis function neural networks (SORBF) and applied it to bankruptcy prediction in banks. Semi Online RBFN without linear terms performed better than techniques such as ANFIS, SVM, MLFF-BP, RBF and Orthogonal RBF. Ravikumar and Ravi [18] proposed an ensemble classifier for the bankruptcy prediction problem based on a host of intelligent techniques. The ensemble classifier was developed in [19] using simple voting scheme and as part of the ensemble they employed seven classifiers such as ANFIS, SVM, RBF, SORBF1, SORBF2, Orthogonal RBF and MLFF-BP. Ravikumar and Ravi [19] conducted a comprehensive review of all the statistical and intelligent techniques applied to the problem of bankruptcy prediction in banks and firms. Similarly, Ravi et al. [20] developed a soft computing system for bank performance prediction.

Research in credit scoring models proliferated during the past 20 years with the application of linear discriminant analysis, logistic regression (LR), decision tree, Bayes network, linear programming, backpropagation trained neural network (BPNN), support vector machines (SVM) etc. these models is the need to increase the scoring accuracy of the credit decision. An improvement in accuracy of even a fraction of a percent translates into significant future savings. Most recently, Farquad et al [21] reviewed the existing credit scoring models and proposed a PCA-SVM hybrid for analyzing the UK credit dataset. For more information, the reader is referred to Farquad et al [21].

3 Particle Swarm Optimization (PSO)

Kennedy and Eberhart [22] proposed the PSO algorithm which is a population based optimization technique in the family of evolutionary algorithms. PSO algorithm mimics the behavior of bird flocking, the social behavior of group of people. Each individual particle in PSO is considered to be a point in the N dimensional search space. The procedure of PSO mainly consists of initialization and velocity update. In initialization phase, randomly generate a population size of solutions called particles with each particle assigned with random velocity (V_{id}^{Old}). The neighborhood best (p_{id}) is the best path traveled by each of the particle in the population. The global best (p_{gd}) is the best path from the entire population. In velocity updation, each particle's velocity is updated with respect to its position (x_{id}^{old}) using neighborhood best (p_{id}) and global best particle (p_{gd}). The velocity (V_{id}^{New}) and position (x_{id}^{New}) of each particle are updated by using the equations 1 and 2 given below:

$$V_{id}^{New} = w * V_{id}^{Old} + c_1 * rand * \left(p_{id} - x_{id}^{old}\right) + c_2 * rand * \left(p_{gd} - x_{id}^{old}\right) \quad (1)$$

$$x_{id}^{New} = x_{id}^{Old} + V_{id}^{New} \quad (2)$$

Where c_1 and c_2 are two predefined positive constants (usually c_1=2, c_2 =2), w is the inertia weight value, which is continuously decreased as the iterations pass, $rand$ is a random number generated from uniform distribution U(0,1).

4 Architecture of the Particle Swarm Optimization Auto Associative Neural Network

The proposed architecture consists of three layers namely input layer, hidden layer and output layer. The input and output layers consist of equal number of nodes and also represent the same input variables. The number of hidden nodes in the hidden layer is a user defined parameter. Each input node in the input layer is connected to each node in the hidden layer and each hidden node is connected to the each of the node in the output layer. Data flow diagram is shown in Figure 1. Sigmoid activation function is used in the hidden layer and output layer.

For training the network, only the negative class data is supplied to the network following Baek and Cho [2] and Pramodh and Ravi [1]. The idea is that the network will learn the knowledge from the negative class data only but not from the positive class. The objective function is to minimize the normalized root mean squared error (NRMSE). Once the network is sufficiently trained we use it for testing. To test the network, only positive class data is fed to the network. Since it is a single-class classifier, ideally, in the test phase, NRMSE would be larger compared to that of the training phase. In order to classify a pattern in the test phase, relative error is computed for each of the nodes present in input layer and output layer. We employed a mechanism using a threshold value for classifying the pattern as negative class or positive class. If the relative error is greater than the threshold value for all the input features of that pattern, then that pattern is classified to belong to positive class. Otherwise, it is classified to belong to negative class.

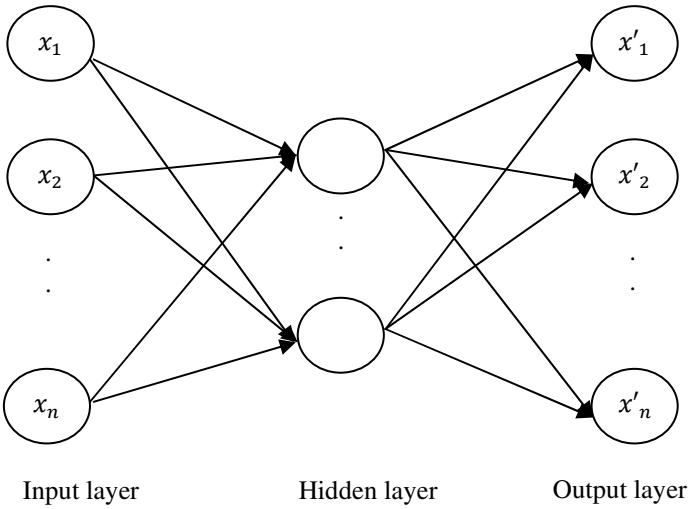


Fig. 1. PSOAANN as a single class classifier

5 Training Algorithm of PSOAANN as a Single Class Classifier

We used Particle Swarm Optimization algorithm to train the 3-layered auto associative neural network unlike 5 layered as Pramodh and Ravi [1]. The training algorithm for PSOAANN as a single class classifier is as follows:

1) Select the number of hidden nodes required in the hidden layer. Initialize randomly weight values between the input and hidden layers and between the hidden and the output layers in the range of [-10 10]. The output nodes contain the input variables as the target variables thereby bringing in the auto associative concept which is taken same as the number of input nodes.

2) Compute $x_i^{\wedge}$ as follows: x_i is the actual input and $x_i^{\wedge}$ is the predicted input. Let nin be the number of input nodes. The predicted output is calculated as follows:

$$x_i^{\wedge} = \frac{1}{1 + e^{-\left[\sum_{j=1}^{nhn} v_{ji} \frac{1}{1+e^{-\left(\sum_{i=1}^{nin} w_{ij} x_{ij}\right)}}\right]}}$$

Where i = 1 to *nin*; *i, j, o* represent the input, hidden and output nodes respectively; w_{ij} represents the weights between the input and hidden layers; v_{ji} represents the weights between the hidden and output layers; x_{ij} represents the input data.

3) Compute error measure (NRMSE) E as follows

$$E = \sqrt{\frac{\sum_{i=1}^{nin}(x_i - x_i^{\wedge})^2}{\sum_{i=1}^{nin} x_i^2}}$$

4) Test whether convergence is achieved by checking if $|E_{old} - E_{new}| < \epsilon$, ϵ is a predefined small value.

5) Update all weight values by PSO algorithm.

6) Repeat step 2 – 5 until convergence is achieved.

7) Threshold value is a user defined parameter. In order to classify a pattern in the test phase, relative error is computed for each of the nodes present in input layer and output layer. If the relative error is greater than the threshold value for all the input features of that pattern, then that pattern is classified to belong to positive class. Otherwise, it is classified to belong to negative class. The classification rate is computed as:

$$classification\ rate = \frac{no.of\ patterns\ classified\ as\ insovlent}{no.of\ total\ patterns\ in\ test\ data} * 100$$

6 Results and Discussions

The effectiveness of the proposed hybrid is analyzed on classification datasets namely Spanish banks [23], Turkish banks [24], UK banks [25], US banks [26], Credit UK [27] and WBC [28] datasets.

Datasets are divided into two sets namely training and test datasets. Training dataset consists of negative class data used to train the network. Test dataset consists of positive class data to test the network.

Population size of PSO is defined as 30, c_1 and c_2 are varied among 2, 3 or 4, and the number of hidden nodes is varied between 2 and 8. The value of the threshold used in test phase is taken as 0.05 uniformly for all datasets.

Accuracies of the datasets are tabulated in Table 1. The proposed single class classifier results are compared to that of MGDAAANN [1] across Spanish Banks, Turkish Banks and US Banks datasets. The proposed method yielded much better results when compared to that of MGDAAANN [1]. Also, in the case of UK Banks, Credit UK Banks and WBC datasets we achieved high classification rates.

Table 1. Accuracies of Positive class

Datasets	PSOAANN (proposed architecture)	Pramodh and Ravi [1]
SPANISH	93.10	72.97
TURKISH	83.33	76.92
UK	86.67	NA
US	98.46	87.69
Credit UK	91.02	NA
WBC	91.91	NA

The proposed method is much simpler than that of the MGDAAANN [1] in that MGDAAANN [1] has five layered architecture and also they used the modified great deluge algorithm (MGDA) in updating the weights. However, in our study, we reduced the architecture size to three layers and also employed PSO for updating the weights during the training of the network. Also, MGDA is a point-based search method, whereas, PSO is a population-based search method. Consequently, the proposed methodology including architecture is far simpler and robust compared to that of MGDAAANN [1].

From the results it is concluded that the proposed single class classified achieved better results across all the datasets analyzed in the study.

7 Conclusions

In the present study, we proposed PSOAANN as a single class classifier. The effectiveness of the proposed model is tested against five binary classification datasets namely Spanish Banks, Turkish Banks, UK Banks, US Banks, Credit UK and WBC. We noticed that the results yielded by the PSOAANN are better when compared with that of the results obtained by MGDAAANN [1]. Of particular significance is that we reduced the architecture size to three layers and also employed PSO for updating the weights during the training of the network. Also, MGDA is a point-based search method, whereas, PSO is a population-based search method.

From the results, it is concluded that the proposed PSOAANN as a single class classifier is not only a simpler architecture but also very robust in achieving high accuracies. Finally, it is also concluded that PSOAANN as a single class classifier can be used as an effective tool in classifying datasets, where the class of interest (usually the positive class) is totally missing in the training data, which is the case in many real life problems for e.g. financial fraud detection.

References

1. Pramod, C., Ravi, V.: Modified Great Deluge Algorithm based Auto-Associative Neural Network for bankruptcy Prediction. International Journal of Computational Intelligence Research 3(4), 363–370 (2007)
2. Baek, J., Cho, S.: Bankruptcy Prediction for credit Risk Using an Auto-Associative Neural Network in Korean Firms. In: The Proceedings of the CIFEr, Hong Kong (2003)
3. Altman, E.: Financial Ratios, Discriminant Analysis and the Prediction of Corporate Bankruptcy. Journal of Finance 23, 589–609 (1968)
4. Wilson, R.L., Sharda, R.: Bankruptcy prediction using neural networks. Decision Support Systems 11, 545–557 (1994)
5. Cole, R., Gunther, J.: A CAMEL Rating's Shelf Life. Federal Reserve Bank of Dallas Review, 13–20 (December 1995)
6. Fraser, D.: The Determinants of Bank Profits: An Analysis of Extremes. Financial Review 11, 69–87 (1976)
7. Karels, G.V., Prakash, A.J.: Multivariate normality and forecasting for business bankruptcy. Journal of Business Finance & Accounting 14, 573–593 (1987)

8. Odom, M., Sharda, R.: A Neural Network for Bankruptcy Prediction. In: Proceedings of the IJCNN International Conference on Neural Networks, San Diego, CA (1990)
9. Ohlson, J.A.: Financial Rations and the Probabilistic Prediction of Bankruptcy. Journal of Accounting Research 18, 109–131 (1980)
10. Tam, K.Y.: Neural Network Models and the Prediction of Bank Bankruptcy. OMEGA 19, 429–445 (1991)
11. Salchenberger, L., Mine, C., Lash, N.: Neural Networks: A Tool for Predicting Thrift Failures. Decision Sciences 23, 899–916 (1992)
12. Tam, K.Y., Kiang, M.: Predicting Bank Failures: A Neural Network Approach. Decision Sciences 23, 926–947 (1992)
13. Atiya, A.F.: Bankruptcy prediction for credit risk using neural networks: A survey and new results. The IEEE Transactions on Neural Networks 12, 929–935 (2001)
14. Shin, K.S., Lee, T.S., Kim, H.J.: An application of support vector machines in bankruptcy prediction model. Expert Systems with Applications (28), 127–135 (2005)
15. Canbas, S., Caubak, Kilic, S.B.: Prediction of commercial bank failure via multivariate statistical analysis of financial structures: The Turkish case. European Journal of Operational Research 166, 528–546 (2005)
16. Ravikumar, P., Ravi, V.: Bankruptcy prediction in banks by Fuzzy Rule based classifier. In: The Proceedings of 1st IEEE International Conference on Digital and Information Management, Bangalore (2006)
17. Ravi, V., Ravi Kumar, P., Ravi Srinivas, E., Kasabov, N.K.: A Semi-Online training algorithm for the Radial Basis Function Neural Networks: Applications to Bankruptcy Prediction in Banks. In: Ravi, V. (ed.) Advances in Banking Technology and Management: Impact of ICT and CRM. Idea Group Inc., USA (2007)
18. Ravikumar, P., Ravi, V.: Bankruptcy prediction in Banks by an Ensemble classifier. In: The Proceedings of IEEE International Conference on Industrial Technology, Mumbai (2006)
19. Ravi Kumar, P., Ravi, V.: Bankruptcy prediction in banks and firms via statistical and intelligent techniques - A Review. Eur. J. Oper. Res. (2006), doi:10.1016/j.ejor.2006.08.043
20. Ravi, V., Kurniawan, H., Thai, P.N.K., Ravikumar, P.: Soft Computing System for Bank Performance prediction. Accepted in Applied Soft Computing Journal (2007)
21. Farquad, M.A.H., Ravi, V., Sriramjee, Praveen, G.: Credit Scoring Using PCA-SVM Hybrid Model. In: Das, V.V., Stephen, J., Chaba, Y. (eds.) CNC 2011. CCIS, vol. 142, pp. 249–253. Springer, Heidelberg (2011)
22. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proceeding of IEEE International Conference on Neural Networks, Piscataway, NJ, USA, pp. 1942–1948 (1995)
23. Olmeda, I., Fernandez, E.: Hybrid classifiers for financial multicriteria decision making: The case of bankruptcy prediction. Comp. Economics 10, 317–335 (1997)
24. Canbas, S., Caubak, B., Kilic, S.B.: Prediction of commercial bank failure via multivariate statistical analysis of financial structures: The Turkish case. European Journal of Operational Research 166, 528–546 (2005)
25. Beynon, M.J., Peel, M.J.: Variable Precision Refought Set Theory and Data Discretisation: An Application to Corporate Failure Prediction. Omega 29, 561–576 (2001)
26. Rahimian, E., Singh, S., Thammachote, T., Virmani, R.: Bankruptcy prediction by neural network. In: Trippi, R.R., Turban, E. (eds.) Neural Networks in Finance and Investing, Burr Ridge, Irwin Professional Publishing, USA (1996)
27. Thomas, L.C., Edelman, D.B., Crook, J.N.: Credit scoring and its applications. SIAM, Philadelphia (2002)
28. Asuncion, A., Newman, D.J.: UCI machine learning repository. University of California, School of Information and Computer Science, Irvine, CA (2007)

A Network Theoretic Analysis of Evolutionary Algorithms

Karthik Kuber, Stuart W. Card, Kishan G. Mehrotra, and Chilukuri K. Mohan

Dept. of Electrical Engineering and Computer Science,
Syracuse University, Syracuse, NY 13244
{kkuber,swcard,mehrotra,mohan}@syr.edu

Abstract. Network theoretic analyses have been shown to be extremely useful in multiple fields and applications. We propose this approach to study the dynamic behavior of evolutionary algorithms, the first such analysis to the best of our knowledge. Evolving populations are represented as dynamic networks, and we show that changes in population characteristics can be recognized at the level of the networks representing successive generations, with implications for possible improvements in the evolutionary algorithm, e.g., in deciding when a population is prematurely converging, and when a reinitialization of the population may be beneficial to reduce computational effort. In this paper, we show that network-theoretic analyses of evolutionary algorithms help in: (i) studying community-level behaviors, and (ii) using graph properties and metrics to analyze evolutionary algorithms.

Keywords: Evolutionary Algorithms, Genetic Algorithms, Dynamic Networks, Early Detection of Convergence.

1 Introduction

Evolutionary algorithms (EAs) operate on sets or multi-sets of individuals (representations of candidate solutions), in which the relationships between individuals are usually ignored. Even when exceptions to this have been explored (e.g., niching [11], crowding [5], and multi-deme algorithms [15]), attention has been focused on relationships between subsets of the population. A few articles have been written focusing on clusters in an evolutionary system [13], adaptive crossover and mutation rates based on clusters [19], formation of stable clusters particularly in swarms [2], and communities in evolving networks [14] etc. By contrast, our focus is on examining relationships between individuals, asking what happens over time to the structural properties of networks representing evolving populations. Our primary focus is on networks for reasons such as: (i) Networks take into account relationships between individuals (via the existence of edges and their lengths), (ii) Networks are better suited to depict various kinds of relationships such as ancestral connections, and (iii) Networks are better representatives of the kinds of relationships that biological species share. This paper presents a preliminary study using a network representation of real-coded

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 585–593, 2012.

Genetic Algorithms (GAs) [9], using nodes in the network to represent individuals in the population, applied to benchmark optimization problems studied by many researchers.

In most other applications where networks change over time, the nodes in a network are mostly the same from one instant to the next, although some edges (and a small number of nodes) appear or disappear. However, in evolutionary algorithms, we observe that individuals in an evolving population may not persist over time. Nodes vanish in one generation, so that there is no obvious "continuity" between networks representing successive generations. Hence the key questions to be posed are at the level of the entire network or its subgraphs.

What kinds of relationships between individuals could be interesting in analyzing evolutionary algorithms? In this paper, we explore Euclidean distance between individuals (in the problem-specific data space). Thus, an edge exists between two nodes (representing two individuals in the population) if the distance between them is less than a threshold that may be chosen based on problem-specific characteristics or the properties of the current network.

In Section 2, we define the network and review some of the network parameters under study, and discuss parallels between a GA and a dynamic network. We also trace the evolution of these parameters over multiple generations. In Section 3, we discuss inherent connections between the changing structure of the convex hull of the largest component, evolution of triangles amongst individuals, the identification of the space where the optimum lies in a GA, and subsequently, its convergence. In Section 4, we demonstrate another utility of such metrics by "intelligently" identifying additional points in the search space to evaluate. Finally, in Section 5, we summarize our observations.

2 Network Creation

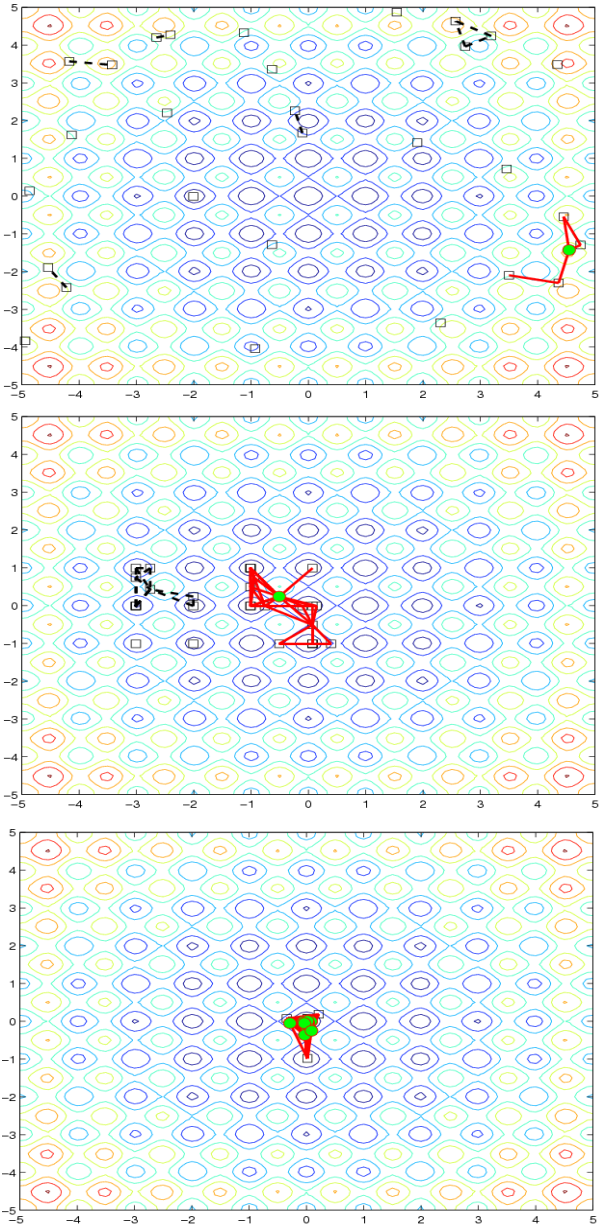
Each network $G =< V, E >$ consists of:

- V: All n individuals in the population are vertices or nodes.
- E: Two individuals have an edge between them if the Euclidean distance between them is less than some value δ which we refer to as the "Edge Threshold". In our simulations, this value is chosen to be proportional to the length of the body-diagonal of the search space.

Snapshots of network and component formation along with emergence of a large component are shown in Figure 1 (a)-(c), for a real-coded GA, population size = 30, two-point crossover rate = 0.8, adaptive feasible mutation rate = 0.2, with roulette selection, generated using MATLAB's GA Toolbox. The background shows the contour plot of Rastrigin's function [16], and networks are superimposed on it.

2.1 Network Metrics under Study

We plot the following network measures over time (Figure 2). This serves as a basis for a comparative exploratory study.



(a) Exploratory phase - We see a few components, one larger than the rest, and a few unconnected individuals.

(b) Midway - We see that a large component has formed. Smaller components and unconnected individuals can also be seen.

(c) Late in evolution - Only one component is seen (and the convex hull has become much smaller), but the algorithm has still not converged in the classical sense.

Fig. 1. Various stages of evolution and network formation. The background is the contour plot of the Rastrigin function - global minimum at (0,0). Solid (red) lines indicate connections within the largest component, dotted (black) lines connect within smaller components. Shaded (green) circles are the individuals with highest centrality. Squares denote other individuals.

(a) Number of nodes in the largest component: It steadily increases and reaches the population size while the EA approaches convergence.
(b) Degree: Two measures of interest are the average degree of all nodes in the population and the average degree of all nodes in the largest component. Both these values grow steadily, and convergence is indicated when the largest component degree ceases to grow.
(c) Hull size of the largest component: The convex hull area (or volume) of the component. For $n > 3$ dimensions, the definition extends to n-polytopes, and many algorithms have been proposed to compute the hypervolume of the hull. We see that the hull size rises initially and then shrinks (barring minor fluctuations).
(d) Number of edges: It has small values in the early stages of EA evolution, incremental jumps in the middle, and finally a large jump near EA convergence, to remain essentially constant thereafter.
(e) Number of triangles: This refers to the number of three-tuples of nodes within the edge threshold distance δ from each other. The behavior of this measure is similar to that of edge count.
(f) Objective function values of nodes with highest degree: This has a behavior similar to that of the mean or best function value.
(g) Centroidal function values: The objective function at the centroid of the largest component. We observe that there are some generations where this value is better than the best function value.

The plots in Figure 2 are for the two-variable Rastrigin's function [16], but we have applied this approach to Ackley's path function [1], DeJong's first function [5], Easom's function [6] and Schwefel's function [18] as well.

3 GA Convergence, Its Relationship to the Largest Component, and Triangle Formation

Typically, the termination of a GA is based on an a priori bound on the number of fitness evaluations; values of the best objective function value, average function value, some combination of the two; or based on testing whether the function value has changed less than a threshold. Various other stopping criteria have been proposed in [3], [10], [17] and [8].

Figure 2(b) leads us to the following observation.

Observation 1. *Initially there is no significant large component. As a large component emerges, the convex hull of this component grows. Once the hull swells, it shrinks in size to a much smaller area (barring minor size fluctuations) until it reaches a very small fraction of its previous highest peak. As the hull is shrinking in area and the number of individuals in it is increasing (or has reached the maximum), the algorithm is approaching convergence, and cannot be expected to do much better.*

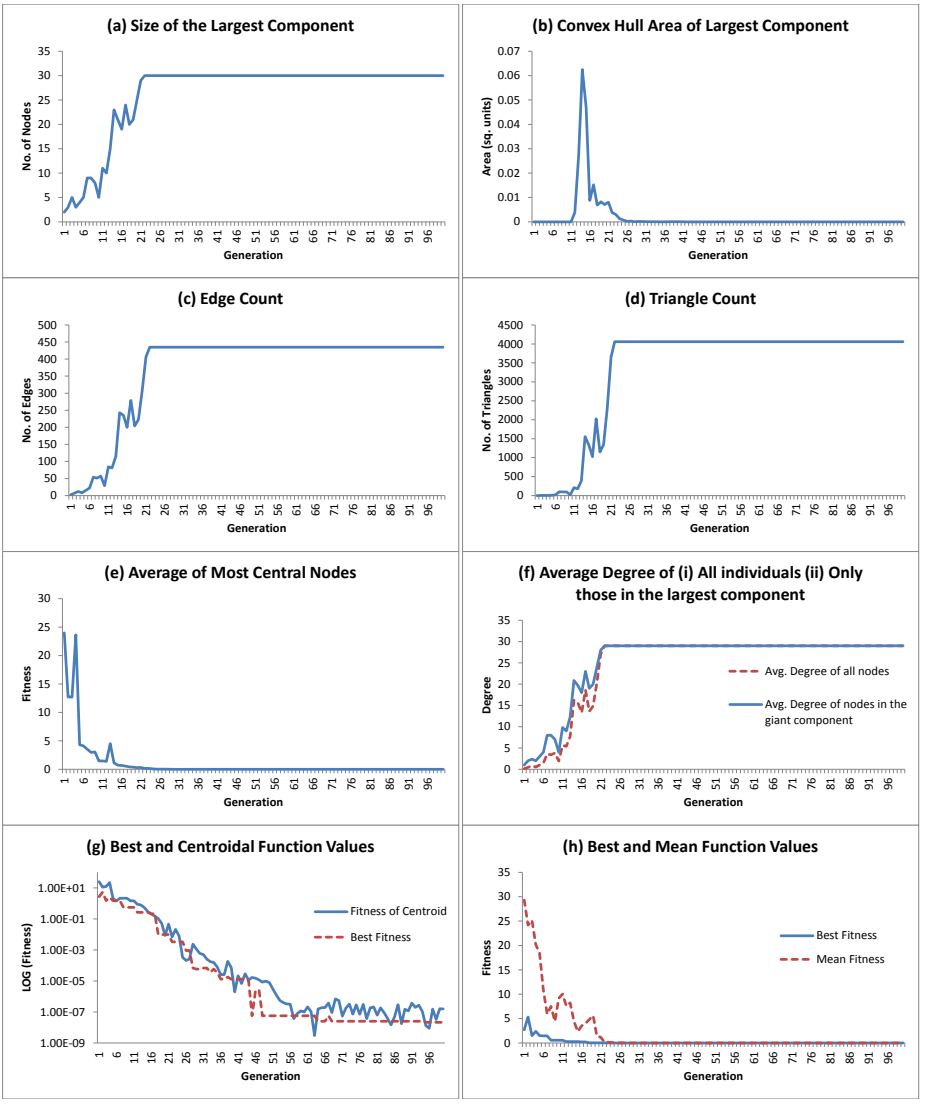


Fig. 2. Plots of network metrics over time. In (b) the area of largest component (convex hull) typically is very low in the beginning and rises slowly while the individuals are still in the exploratory phase. We then notice a steep rise and a large component forms, and then a fall when convergence occurs, and it appears to have "collapsed under its own gravity". In (g), we see that there are a few instances where the centroid has found a better point in the surface that all the other individuals combined. This is because points are converging from different parts of the space; assuming there are sufficient learning possibilities in the space, there should be some instances of a point in their center which is a more optimal one.

Most standard GA crossovers are convex operators (although mutation is not) [12]. This observation of convexity is consistent with our view that a component forms and then shrinks, and this notion has been exploited in the above algorithm. The results are in Table 1. We also observe that the count of the number of triangles increases linearly and remains well below the value of the total possible number of triangles, viz. $\binom{n}{3}$,where n is the total population size. The count then increases exponentially, and approaches a value close to $\binom{n}{3}$. This phenomenon is seen in Figure 2(d). Algorithm 1 exploits this observation in order to reduce computational effort.

Algorithm 1. Convergence based on the convex hull of the largest component

Initialize the population;
α takes values in $[0.6, 0.9]$, ϵ takes values in $[0.005, 0.03]$;
repeat Perform recombinations and mutations to obtain next generation of GA
until number of individuals in the largest component $\geq \alpha.n$
$\gamma \leftarrow$ Greatest Hull Area (observed so far) of the largest component
repeatPerform recombinations and mutations to obtain next generation of GA
until hull area $\leq \epsilon.\gamma$, and number of individuals in the largest component $\geq \alpha.n$

Table 1. Results of running Algorithm 1 (Average of 30 runs). (i) δ - Edge threshold (ii) Pop - Population Size (iii) Gen - Generation at which the algorithm is stopped according to the algorithm, (iv) Diff - The difference between the function value at this stopped generation and the value obtained if we had continued the GA until convergence occurs, (v) r - Ratio of number of triangles at the stopped generation and total number of triangles possible.

		δ=**0.001**			δ=**0.005**			δ=**0.01**		
Function	**Pop**	**Gen**	**Diff**	r	**Gen**	**Diff**	r	**Gen**	**Diff**	r
Rastrigin	30	48	0	1	36	8.00E-05	1	35	0.00016	1
	100	44	7.43E-07	0.99994	33	2.10E-05	0.99746	33	9.10E-05	0.99913
Ackley	30	39	0	1	29	0.0003	1	27	0.0008	1
	100	39	0	0.99796	31	0.0002	0.99988	24	0.001	0.99802
Easom	30	48	1.00E-05	0.97116	41	0.00017	1	40	0.0001	0.97919
	100	35	6.00E-06	0.99054	29	0.00015	0.94589	32	6.00E-05	0.98565
DeJong	30	49	4.90E-08	1	36	1.30E-07	1	34	1.83E-06	1
	100	44	3.44E-09	0.99845	32	1.43E-07	0.99951	33	4.19E-07	0.99647
Schwefel	30	129	0	1	162	0	1	131	0	1
	100	59	0	0.99685	57	3.42E-05	1	55	0.048228	1

Time Complexity: An average over 30 runs of Rastrigin's function (variables: 2, population size: 30, generations: 200) took 0.61 seconds to run, while stopping at Generation 48 by including computation of network metrics took only 0.19 seconds to run. We observe that although there is a slight overhead time per generation, stopping early clearly saves a lot of computation time without sacrificing quality (note that $Diff \approx 0$). This verifies that the generations where the GA was stopped for each problem instance were indeed acceptable. We are presently working on a theoretical time complexity analysis as presented in [4].

Experimentally, we also found that the values do not differ much for $\alpha \in [0.6, 0.9]$ or $\epsilon \in [0.005, 0.03]$, they all lie within 1 or 2 generations of each other. Smaller values of α typically yield components that are too small to indicate convergence, and large values of ϵ may stop the algorithm too early. Further, we observed that it is best to keep the edge threshold (δ) low ($\leq PopSize^{-1}$). As δ increases, we again run the risk of stopping the algorithm too early and may yield unacceptable values of "Diff".

4 GA and Centroid of the Largest Component

In Figure 2(g), we observe that the function value at the centroid of the largest component is occasionally better than the best individual in the population. This observation is exploited to improve the GA, and is summarized below.

Observation 2. *The individuals of a population are likely to converge from different directions into the optimum and hence the centroid of the largest component, which lies in the middle of this component, occasionally has the best objective function value amongst all individuals. In other words, the largest component identifies the region of interest, i.e. the points are likely to have better function values than others in the population, and the centroid may have a better value than all of them.*

Algorithm 2 is based on this observation, and helps reach better solutions more rapidly in some problems.

Algorithm 2. Centroid Elitism

```
Initialize the population;
repeatCompute function value of centroid of the largest component
    if There is more than one largest component then
        Pick any component at random;
    end if
    if Function value of centroid is better than the function value at all other points
     in the population then
        Delete worst individual in population;
        Introduce new individual at centroid;
    end if
     Perform recombinations and mutations to obtain next generation of GA
until Number of generations ≥ Maximum number of generations
```

The emphasis is on the approach of the population towards the optimum, rather than just the position of individuals. (It must be noted that this algorithm is not the same as multi-parent recombination [7].) Table 2 shows that in 22 out of 25 cases, the modification yields a better function value. Note that many of these runs show premature convergences (as a result of low population sizes), and eliminating an occasional least fit point in favor of a highly fit centroid betters the performance, i.e. the trade-off between exploration and exploitation is acceptable in this situation. The t-test compares best values obtained at the $100th$ generation for 30 runs of both the regular and modified algorithms.

Table 2. Best function values at various generations using regular GA & Algorithm 2 (Avg. of 30 runs). PopSize=30 (Reg: Regular GA, Mod: Modified using Algorithm 2).

	Rastrigin		**Ackley**		**Easom**		**DeJong**		**Schwefel**	
Gen	**Reg**	**Mod**	**Reg**	**Mod**	**Reg**	**Mod**	**Reg**	**Mod**	**Reg**	**Mod**
10	1.1229	0.97876	0.2677	0.224	0.85403	*0.93311*	0.0072289	*0.0092654*	116.94	76.869
30	0.76316	0.53083	0.1145	0.0853	0.4368	0.24146	3.99E-06	*2.42E-05*	97.03	58.508
50	0.7628	0.53064	0.1142	0.0849	0.43332	0.19999	9.05E-09	4.01E-09	88.06	46.281
70	0.7628	0.53064	0.1142	0.0849	0.43331	0.19999	2.36E-10	9.98E-11	85.289	40.273
90	0.7628	0.53064	0.1142	0.0849	0.43331	0.19999	1.68E-10	1.98E-11	84.373	37.009
t-test (p-values)	0.0004		0.2815		0.0252		0.0001		0.0218	

5 Conclusions

In this paper, we have analyzed a GA via network metrics, mainly to identify convergence faster, without sacrificing the quality of the results, and also to use these metrics to help speed up a GA. Experiments performed on five test problems demonstrate that we can use these evolving networks to identify a niche space where the optimum solution lies, as opposed to waiting for the GA to converge. We also demonstrate that the largest component is a key identifier of the region of interest and its centroid is a good representative of this space.

These simulation experiments show that there are potential benefits to utilizing the science that dynamic networks have to offer and applying them to GAs. Viewing these GA populations as evolving networks can give us a different perspective about what the GAs set out to achieve as their optimization or prediction goal.

References

1. Ackley, D.H.: An empirical study of bit vector function optimization. Genetic Algorithms and Simulated Annealing 1, 170–204 (1987)
2. Al-Rifaie, M.M., Mark, J.B.: Stochastic diffusion search review (2010)
3. Aytug, H., Koehler, G.J.: New stopping criterion for genetic algorithms. European Journal of Operational Research 126(3), 662–674 (2000)
4. Chen, T., He, J., Sun, G., Chen, G., Yao, X.: A new approach for analyzing average time complexity of population-based evolutionary algorithms on unimodal problems. IEEE Trans. on Systems, Man, and Cybernetics: Part B 39(5), 1092–1106 (2009)
5. De Jong, K.A.: Analysis of the behavior of a class of genetic adaptive systems (1975)
6. Easom, E.E.: A Survey of Global Optimization Techniques (1990)
7. Eiben, A., Raue, P., Ruttkay, Z.: Genetic Algorithms with Multi-parent Recombination. In: Davidor, Y., Männer, R., Schwefel, H.-P. (eds.) PPSN 1994. LNCS, vol. 866, pp. 78–87. Springer, Heidelberg (1994)
8. Gibbs, M.S., Maier, H.R., Dandy, G.C., Nixon, J.B.: Minimum number of generations required for convergence of genetic algorithms. In: IEEE CEC (2006)

9. Goldberg, D.E.: Genetic algorithms in search, optimization, and machine learning. Addison-Wesley (1989)
10. Greenhalgh, D., Marshall, S.: Convergence criteria for genetic algorithms. SIAM Journal on Computing 30, 269–282 (2000)
11. Mahfoud, S.W.: Niching methods for genetic algorithms. Urbana 51(95001) (1995)
12. Moraglio, A.: Towards a geometric unification of evolutionary algorithms (2007)
13. Nair, P.B., Keane, A.J.: Passive vibration suppression of flexible space structures via optimal geometric redesign. AIAA Journal 39(7), 1338–1346 (2001)
14. Pandit, S., Yang, Y., Kawadia, V., Sreenivasan, S., Chawla, N.V.: Detecting communities in time-evolving proximity networks. In: Network Sci. Workshop. IEEE (2011)
15. Pettey, C.B., Leuze, M.R., Grefenstette, J.J.: A parallel genetic algorithm. In: Proceedings of the Second ICGA (1987)
16. Rastrigin, L.A.: Extremal control systems. Theoretical Foundations of Engineering Cybernetics Series, vol. 3 (1974)
17. Safe, M., Carballido, J., Ponzoni, I., Brignole, N.: On stopping criteria for genetic algorithms. In: Bazzan, A.L.C., Labidi, S. (eds.) SBIA 2004. LNCS (LNAI), vol. 3171, pp. 405–413. Springer, Heidelberg (2004)
18. Schwefel, H.P.: Numerical optimization of computer models. John Wiley & Sons, Inc. (1981)
19. Zhang, J., Chung, H.S.H., Lo, W.L.: Clustering-based adaptive crossover and mutation probabilities for genetic algorithms. IEEE Transactions on Evolutionary Computation 11(3), 326–335 (2007)

Characterization of Trabecular Architecture in Human Femur Radiographic Images Using Directional Multiresolution Transform and AdaBoost Model

Thomas Christy Bobby[1] and Swaminathan Ramakrishnan[2]

[1] Department of Instrumentation Engineering, MIT Campus,
Anna University, Chennai, India
[2] Biomedical Engineering Group,
Department of Applied Mechanics, IIT Madras, India
christybobbysenthil@gmail.com, sramki@iitm.ac.in

Abstract. In this work, directional multiresolution curvelet transform is performed in radiographic images to characterize the trabecular structure. The trabecular regions of normal and abnormal human femur bone images are used for the study. The regions of interest such as femoral neck and head are analyzed and compared. The curvelet coefficients are calculated based on each scale and orientation for trabecular images. The mean and energy of the curvelet coefficients associated with each subband are computed. These values are used as the texture feature vector elements to evaluate changes taking place in the trabecular architecture. The three most significant mean and energy feature vector are found using principal component analysis and these values are used as an input to the Adaboost classifier. The results show that the architectural variations are more in the femoral neck when compared to femoral head. AdaBoost classifier performs better in terms of sensitivity (90%) and specificity (100%) for the chosen parameters for femoral neck region when compared to head regions.

Keywords: Curvelet transform, Human femur, Trabecular analysis, Osteoporosis, Radiographic images, AdaBoost.

1 Introduction

Osteoporosis is a systemic skeletal disease characterized by reduction in bone strength that increases the fracture risk. The bone strength reflects both bone density and quality. Bone quality depends on trabecular bone architecture, mineralization, turnover, and microfracture accumulation. Fracture risk evaluations rely on many surrogate markers, among which trabecular bone microarchitecture occupies a place of prominence. Hence the potential benefits from trabecular architecture evaluation improve the prediction of the fracture [1, 2]. Digital radiography is a widely available imaging modality that has the potential to reflect trabecular microarchitecture. The significant parts of the information that are available in three dimensional images are

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 594–601, 2012.

also available in great detail on standard radiographs. X-ray imaging remains a very cost-effective technique, with many applications in both the medical and material science fields. Hence, there has been considerable interest in using conventional radiography combined with various image and texture analysis techniques for assessing trabecular structure [3, 1].

Texture characterization of the images is an important part in medical image analysis. Texture analysis describes a wide range of techniques that enable quantification of the gray-level patterns, pixel interrelationships, and the spectral properties of an image that are imperceptible to the human visual system [4]. A variety of techniques developed for extracting texture features, is broadly classified into the spatial and spectral methods. The spectral domain methods such as Fourier, Wavelets, Gabor filters and Radon transforms are most often used to extract clinical information on images recorded with different modalities [5-7].Transform methods have the advantage of being insensitive to noise. Therefore, transforms have widely been used to represent image textures. Discrete curvelet transform is a directional multiresolution method, which is effective in representing the curve and line edges in an image. Also, discrete curvelet transform gives more directional information from high frequency components [8, 9]. Thus curvelet transform provide efficient representation for images like femur trabecular structures with line discontinuities and curve edges in different orientations.

Principal Component Analysis (PCA) is a standard feature reduction tool in data analysis. It identifies patterns in data, and expresses in such a way to highlight their similarities and differences. In the analysis of multivariate datasets, PCA is used to select the most significant parameters and the parameters are ranked based on their highest magnitudes in the principal components obtained. It has been widely employed in various signal and image processing problems [10].

Conventionally, radiologists identify image patterns and diagnose on the basis of a combination of their training, experience, and individual judgement. It follows that there will be certain degree of variability in image interpretation as long as it relies primarily on human visual perception. [11]. Hence identification of tools for automated classification of femur bone images is an important step towards clinical decision making. Classifiers such as Multi Layer Perceptron, Radial Basis Function, Support Vector Machines and AdaBoost classifier [12-14] have been employed recently in various medical image and signal applications to automate the classification procedures. AdaBoost is an ensemble learner where the joint decision rule of multiple weak classifiers forms an overall strong classifier. Recently, AdaBoost classifiers have automated the classification process for several imaging applications such as Alzheimer's disease and bone fracture detection [15].

In this work, radiographic human femur bone images are processed using image processing techniques and the sub-regions such as femoral head and neck are delineated from the processed images. The mean and energy feature vectors are calculated from the curvelet coefficients of different sub-bands for the delineated regions of interest. From the derived feature vectors, the first three most significant feature vectors are chosen using PCA and are used as inputs to the classifier. AdaBoost classifier is employed for the classification of normal and abnormal femur bone images. The performances of the classifier are estimated and compared for the subregions.

2 Methodology

Digitized pelvis images (N = 44) recorded using Siemens 500mA polyscope clinical X-ray unit are considered for this analysis. Auto threshold binarization algorithm is employed to recognize the presence of mineralization in the digitized images [16]. From the processed bone images, Regions of Interest (ROI) of image size 152 × 152 (M*N) pixels corresponding to femoral head and neck are delineated, and considered for the analysis. Femoral head and neck are structurally different regions where more mineral loss and architectural variations occurs due to different loading conditions and stress distribution [17-19]. Singh identified these groups in the proximal femur trabecular bone, which change most markedly in appearance with bone loss and are heterogeneous in nature and have varying bone strength [20]. Discrete curvelet transform [9] coefficients can be defined by

$$c^D(j,l,k_1,k_2) = \sum_{\substack{0\le m<M \\ 0\le n<N}} f[m,n]\,\varphi^D{}_{j,l,k_1,k_2}[m,n] \tag{1}$$

Here, each $\varphi^D{}_{j,l,k_1,k_2}[m,n]$ is a digital curvelet waveform. Discrete curvelet transform based on wrapping of Fourier samples with 4 decomposition levels is applied to an image to obtain its coefficients. It takes the image as input in the form of a Cartesian array *f [m, n]* such that $0 \le m < M$, $0 \le n < N$ and generates a number of curvelet coefficients indexed by a scale *j*, orientation *l* (16) and two spatial location parameters *(k1, k2)*. The curvelet coefficients are generated and stored in each sub band. Table 1 shows the details of scale, total number of sub-bands and the sub-bands considered to calculate the statistical parameters at level four decomposition. The statistical texture features such as mean and energy are derived from these curvelet coefficients.

Table 1. The details of curvelet subband Distribution at Each Scale

Curvelet transform (4 level decomposition)				
Scale	1	2	3	4
Total no. of sub-bands	1	16	32	1
Sub-band considered for statistical operations	1	8	16	1

The mean and energy features are calculated for the sub-band at the coarsest and the finest scale separately. First half of the total sub-bands at a resolution level 2 and 3 at angle θ produce the same coefficients as the curvelet at angle $(\theta+\pi)$ in the frequency domain due to symmetric property of the sub-bands. The total number of feature vectors generated is 26 (1+8+16+1) for each statistical operation [21].The Principal Component Analysis is performed on the derived feature vectors to select the significant feature vector. PCA uncovers combinations of the original variables which describe the dominant patterns and the main trends in the data. This is done through an eigenvector decomposition of the covariance matrix of the original

variables. The extracted latent variables are orthogonal and they are sorted according to their eigenvalues. The high dimensional space described by matrix X is modelled using PCA as

$$X = TP^T + E \tag{2}$$

where T is the score matrix composed by the principal components, P is the loadings composed by the eigenvectors of the covariance matrix and E is the residual matrix [22]. The first three most significant parameters derived using PCA are used as an input to the AdaBoost classifiers. The total numbers of training data considered for analysis are 44, out of which 20 are considered for testing.

AdaBoost assigns an initial uniform weight $W_1(i) = 1/n$ to each training sample x_i, where n is the number of training samples. At iteration k, AdaBoost finds the classifier h_k trained using samples x_i that minimizes the weighted error E_k according to $W_k(i)$. The weights are updated by

$$W_{k+1}(i) = \frac{W_k(i).\exp(-a_k y_i h_k(x_i))}{Z_k}, \quad a_k = \frac{1}{2}\log\left(\frac{1-E_k}{E_k}\right) \tag{3}$$

where, y_i are the ground truth labels, and Z_k is a normalization factor where α_k is set. Weights for correctly classified samples are increased and weights for incorrectly classified samples are decreased. Due to this AdaBoost focuses on informative and difficult training samples. The resulting classifier of the AdaBoost algorithm becomes

$$H(x) = Sign\left(\sum_{i=1}^{T} \alpha_t h_t(x)\right) \tag{4}$$

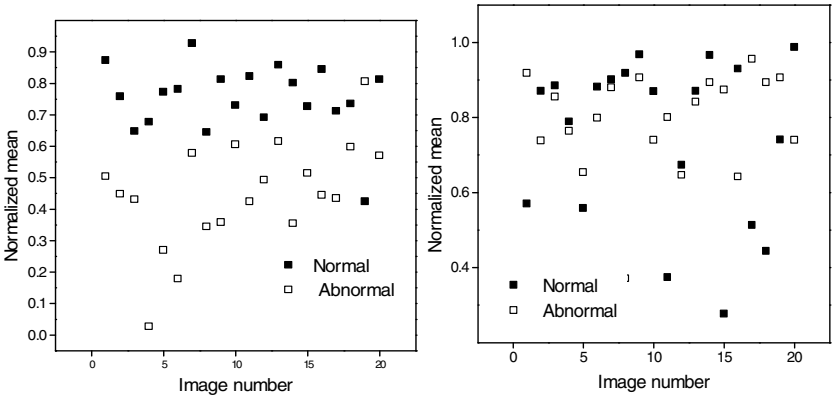
where h_t is the t^{th} weak classifier hypothesis and $H(x)$ is the strong classifier hypothesis. AdaBoost minimizes the weighted error on a single feature. At iteration k, AdaBoost selects the feature which minimizes the weighted error. The value of $W_{k+1}(i)$ is increased when x_i is classified correctly by h_k, and decreased otherwise [15].The performance of the algorithm and the classifiers are tested using estimators such as sensitivity and specificity. Specificity is a statistical measure and it is related to true negatives of all the negative classes in the dataset. Sensitivity indicates the proportion of true positive of all the positive classes in the dataset [23].

3 Results and Discussion

Typical planar radiographic images of normal and abnormal femur trabecular bone are shown in Fig. 1 (a) and (b) respectively. The trabecular patterns are closely arranged in normal images whereas the trabecular spacing is seen with discontinuities in abnormal images. The overlap between trabeculae is found to be less in abnormal when compared to normal images.

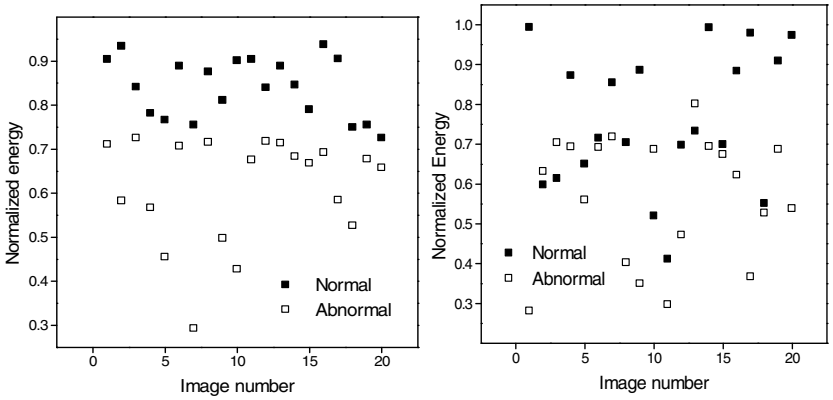
(a) (b)

Fig. 1. Typical (a) normal and (b) abnormal images



(a) (b)

Fig. 2. Scattergram of normalized mean values for (a) femoral neck (b) femoral head

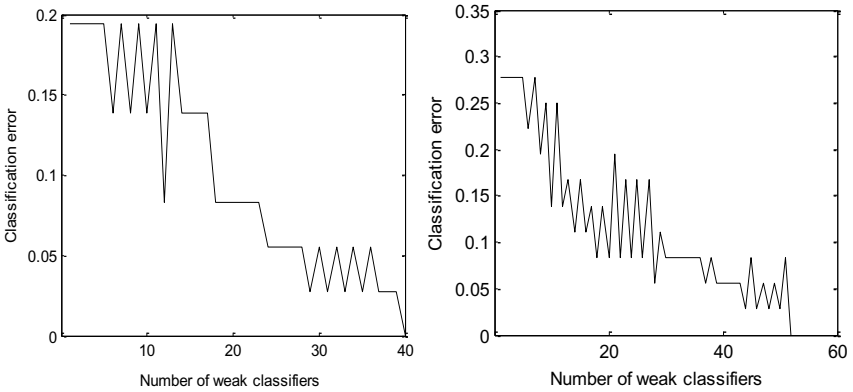


(a) (b)

Fig. 3. Scattergram of normalized energy values for (a) femoral neck (b) femoral head

Figures 2 and 3 show the normalize value of mean and energy values of first significant feature vector derived for femoral head and neck. Fig 2(a) and 3(a) show the mean and energy values respectively of femoral neck which are found to have clear demarcations between normal and abnormal images. Whereas, the mean and

energy values of femoral head, are found to be overlapping as shown in Fig 2(b) and 3(b) respectively. The other two significant features also show same kind of scattergram for mean and energy values. This is attributable to the proven fact that the deviation in architectural variations in the form of discontinuities and reduction in mineralization are more in femoral neck region compared to femoral head region [16].



(a) (b)

Fig. 4. Error analysis for AdaBoost classifier (a) femoral neck (b) femoral head

In the analysis using AdaBoost classifier, the variations in error for varying numbers of weak classifiers are shown in Fig 4. Typical error curves for femoral neck and head regions are shown in Figure 4(a) and 4(b) respectively. It is observed that there is a large reduction in error for the increased number of weak classifiers. AdaBoost classifier is found to utilize more number of weak classifiers for classification in femoral head region compared to neck region. This may be due to the similarity or difference pattern in the most significant features chosen by PCA. It is also observed that the statistical parameters are not very distinct in femoral head region due to less variation in the trabecular architecture compared to femoral neck.

Table 2. Comparison of performance measures using AdaBoost classifiers in femoral head and neck regions

Performance estimators	AdaBoost Classifiers	
	Femoral neck	Femoral head
Sensitivity (%)	90	80
Specificity (%)	100	90

The performance estimators of the Adaboost classifiers for femoral neck and head regions are shown in Table 2. It is observed that the sensitivity and specificity are high for femoral neck when compared to the femoral head region. Also, the specificity is high for both the regions when compared to the sensitivity value.

This explains that the classifier performs well for abnormal cases. The classifier performance could further be increased by increasing the number of training and testing data sets.

4 Conclusion

Analyses of medical images using transform based methods are shown to be effective for various clinical inferences. Especially, they are used to extract variations manifested in images in the form of discontinuities and irregular patterns [5-7, 24]. In biomechanics, analysis of trabecular images in terms of their architecture are to be performed for better understanding of strength of bone in addition to bone mineral density. Such trabecular analysis yields valuable surrogate information regarding fracture risk, bone strength and response to therapy [1, 2].

In this work, discrete curvelet transform is employed to analyze considerable regional variations in trabecular bone structure using radiographic femur images. To automate the analysis, Adaboost classifier is employed for classification purpose. The results show that, it is possible to differentiate normal and abnormal images using curvelet transform derived features and architectural variations are more in the femoral neck region when compared to femoral head region. The observed changes are attributed to varying degree of trabecular bone loss, homogeneity and anisotropy. AdaBoost classifier provides better classification with acceptable sensitivity and specificity. Thus it appears that curvelet transform based feature extraction together with AdaBoost classification seems to be useful for trabecular architectural analysis. As automated analysis of human bone architecture is required for mass screening of disorders such as osteoporosis, this study seems to be clinically relevant.

References

1. Lespessailles, E., Chappard, C., Bonnet, N., Benhamou, C.L.: Imaging techniques for evaluating bone microarchitecture. Joint Bone Spine 73, 254–261 (2006)
2. Donnelly, E.: Methods for Assessing Bone Quality. Clin. Orthop. Relat. Res. 469, 2128–2138 (2011)
3. Corroller, T.L., Halgrin, J., Pithioux, M., Guenoun, D., Chabrand, P., Champsaur, P.: Combination of texture analysis and bone mineral density improves the prediction of fracture load in human femurs. Osteoporos Int. 23, 163–169 (2012)
4. Xie, X.: A Review of Recent Advances in Surface Defect Detection using Texture analysis Techniques. ELCVIA 7(3), 1–22 (2008)
5. Boehm, H.F., Lutz, M., Korner, M., Mutschler, W.: Using Radon Transform of standard radiographs of the hip to differentiate between post-menopausal women with and without fracture of the proximal femur. Osteoporosis Int. 20, 323–333 (2009)
6. Gregory, J.S., Stewart, A., Undrill, P.E., Reid, D.M., Aspden, R.M.: Identification of hip fracture patients from radiographs using Fourier analysis of the trabecular structure: A cross-sectional study. BMC Medical Imaging 4 (4), 1–11 (2004)
7. Bullmore, E., Fadili, J., Maxim, V., Sendur, L., Whitcher, B., Suckling, J., Brammer, M., Breakspear, M.: Wavelets and functional magnetic resonance imaging of the human brain. Neuroimage 23, 234–249 (2004)

8. Starck, J.L., Candès, E.J., Donoho, D.L.: The Curvelet Transform for Image Denoising. IEEE T. Image Process 11, 670–684 (2002)
9. Candes, E., Demanet, L., Donoho, D., Ying, L.: Fast discrete curvelet transforms. Multiscale Model. Simul. 5, 861–899 (2006)
10. Samanwoy, G.D., Hojjat, A., Nahid, D.: Principal component analysis-enhanced cosine radial basis function neural network for robust epilepsy and seizure detection. IEEE T. Bio-Med. Eng. 55, 512–518 (2008)
11. Kassner, A., Thornhill, R.E.: Texture Analysis: A Review of Neurologic MR Imaging Applications. Am. J. Neuroradiol. 31, 809–816 (2010)
12. Oliveira, J.E.E.D., Araujo, A.D.A., Deserno, T.M.: Content-based image retrieval applied to BI-R ADS tissue classification in screening mammography. World Journal of Radiology 3(1), 24–31 (2011)
13. Jonathan, H.M., Zhuowen, T., Apostolova, L.G., Amity, E.G., Toga, A.W., Thompson, P.M.: Comparison of AdaBoost and support vector machines for detecting Alzheimer's disease through automated hippocampal segmentation. IEEE T. Med. Imaging. 29, 30–43 (2010)
14. Kavitha, A.,, Ramakrishnan, S.: Prediction of Forced Expiratory Volume (FEV6) in flow–volume spirometry using support vector machines and radial basis function neural networks. J. Mech. Med. Biol. 10, 683–693 (2010)
15. Flores, A., Linguraru, M.G., Okada, K.: Boosted-LDA for biomedical data analysis. In: MICCAI Workshop on Machine Learning in Medical Imaging, pp. 1–8 (2010)
16. Christopher, J.J., Ramakrishnan, S.: Assessment and classification of mechanical strength components of human femur trabecular bone using texture analysis and neural network. J. Med. Syst. 32, 117–122 (2008)
17. Rudman, K.E., Aspden, R.M., Meakin, J.R.: Compression or tension? The stress distribution in the proximal femur. Biomed. Eng. Online 5(12), 1–7 (2006)
18. Blain, H., Chavassieux, P., Portero-Muzy, N., Bonnel, F., Canovas, F., Chammas, M., Maury, P., Delmas, P.D.: Cortical and trabecular bone distribution in the femoral neck in osteoporosis and osteoarthritis. Bone 43, 862–868 (2008)
19. Alan, H.: The structure of the femoral neck: A physical dissection with emphasis on the internal trabecular system. Ann. Anat. 192, 168–177 (2010)
20. Singh, M., Nagarath, A.R., Maini, P.S.: Changes in trabecular pattern of the upper end of the femur as an index of osteoporosis. J. Bone Joint Surg. Am. 52, 457–467 (1970)
21. Sumana, I., Islam, M., Zhang, D.S., Lu, G.: Content Based Image Retrieval Using Curvelet Transform. In: IEEE International Workshop on Multimedia Signal Processing, Australia, pp. 11–16 (2008)
22. Aguado, D., Montoy, T., Borras, L., Seco, A., Ferrer, J.: Using SOM and PCA for analyzing and interpreting data from a P-removal SBR. Eng. Appl. Artif. Intel. 21, 919–930 (2008)
23. Fawcett, T.: An introduction to ROC analysis. Pattern Recognit. Lett. 27, 861–874 (2006)
24. Ratnaparkhe, V.R., Manthalkar, R.R., Joshi, Y.V.: Texture characterization of CT images based on ridgelet transform. ICGST-GVIP, 8(5), 43–50 (2009)

Missing Value Estimation of Microarray Data Using Similarity Measurement

Soumen Kumar Pati[1] and Asit Kumar Das[2]

[1] Department of Computer Science/Information Technology,
St. Thomas' College of Engineering and Technology, 4, D.H. Road, Kolkata-23
[2] Department of Computer Science and Technology,
Bengal Engineering and Science University, Shibpur, Howrah-03
Soumen_pati@rediffmail.com, asitdas72@rediffmail.com

Abstract. DNA gene expression profiling plays an important role in a wide range of areas in biological science for handling cancer diseases. Data generated in microarray related experiments have many missing expression values which lose valuable information from the dataset. The proposed method first partitions the genes without missing values using clustering algorithm and then measures the similarity between a gene with missing values and the centroid of the clusters and finally, the missing values are estimated by the corresponding expression values of the centroid giving maximum similarity factor. The method explicitly depends on expression values to imputes missing values, completed the input dataset with low errors for data analysis and knowledge discovery. The method is compared with prominent approaches, such as zero-impute, row-average-impute and KNN-impute in terms of "Normalized Root Mean Square Error" to claim its novelty.

Keywords: DNA Microarray data, Gene expression value, Clustering algorithm, Similarity measurement, Missing value imputation.

1 Introduction

DNA microarray technology gives a global view of gene expression monitoring the mRNA levels of thousands of genes in particular cells or tissues. Microarray datasets [1] are usually in the form of large tables of expression levels of genes (rows) under different experimental samples (columns). The datasets frequently contain missing values due to insufficient resolution, spotting or scratches on the slide, image corruption, dust or hybridization failures and so on [2]. Unfortunately, most of algorithms for gene data analysis require a complete matrix as input. So the proper and more accurate prediction of Missing values remains an important preprocessing step to analyze microarray dataset. Several approaches [3-8] are proposed by the researchers to deal with missing values. The approach [3] repeats the original experiment to get microarray dataset without missing values, which is expensive and more time consuming. The approach [4] ignores genes containing missing values, that usually loses many useful information and may bias the results if the remaining genes

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 602–610, 2012.

unable to represent the entire dataset. Some approaches [4, 5] estimate the missing values by a global constant such as zero (0), or by the average of the available sample values for that gene, which distort relationships among expression values for that gene. And others [7] consider the correlation structure among expression values for a gene. The estimating procedure consists of two steps: in the first step similar expression values related genes to the gene with missing value, are selected and in the second step the missing values are predicted using observed values of selected genes, for example the widely used weighted K-nearest neighbor (KNN) imputation, estimate the missing values using a weighted average of K most similar genes [6]. These methods have better performance than previous one, but the drawback is that their estimation ability depends on parameter K (number of K neighbor genes used to estimate missing value) for which no theoretical way exist to determine them appropriately and thus need to be specified by the user. Whereas, in [2, 8], cluster-based algorithms have been proposed to deal with missing values which don't need user to determine K parameters [7] but microarray dataset is very high dimensional and there exist large number of genes with large number of samples which may degrades the clustering performance. Also this method depends on number of clusters whose selection becomes very crucial. So this approach is also inefficient to deal with missing values.

In the article, a novel missing value estimation technique has been proposed on microarray dataset for imputing missing values that not only overcomes the constraints of the existing methods [2-8] but also gives significantly less Normalized Root Mean Square Error (NRMSE). The method of missing value estimation consists of the following steps:

i. The dataset is standardized to Z-score using Transitional State Discrimination method (TSD) [9] and the samples are characterized by N (here, N = 5) discrete sample values. As the samples are collected from both normal and cancerous patients, they are divided into two disjoint classes. For each gene, frequencies of sample values are computed in each class (i.e., normal and cancerous).
ii. Based on the frequencies of discrete sample values, the genes without missing value are partitioned into $3 \times N$ (here, 15) different groups, explained in the following section. N out of $3 \times N$ groups contains whole portion (i.e., normal and cancerous samples) of the genes while each N of remaining $2 \times N$ groups contains only one portion (i.e., either normal or cancerous samples) of the genes.
iii. Either a gene with missing values is associated to one of the N groups containing whole portion or each of its two portions (i.e., normal and cancerous) is associated to one of the $2 \times N$ respective groups containing only one portion.
iv. Now the determined group(s) is partitioned into optimal set of clusters using clustering algorithm [10] and similarity factors are measured between centroid of each partition and associated portion of the gene. The missing values of the associated portion of the gene are imputed by the respective values of the centroid with most similar partition. Thus, missing values of each gene are imputed by repeating step (iii) and step (iv).

The article is organized into four sections. Section 2 describes the proposed missing value estimation technique. The experimental results and performance of the proposed method for various benchmark gene expression datasets is evaluated in Section 3. Finally, conclusions are drawn in Section 4.

2 Missing Value Estimation Method

Microarray technology [1] is a very high throughput technology that evaluates the expression of immense number of genes simultaneously under different experimental conditions. These conditions may be a time series during a biological process or a collection of different tissue samples (e.g. normal versus cancerous samples). Usually data from microarray experiments contains missing values due to different reasons including dust or scratches on the slide, error in experiments, image corruption, insufficient resolution for which (5 – 50)% genes are affected. Therefore missing value estimation is an important data preprocessing step to impute proper expression values with less error.

2.1 Discretization of Gene Expression Values

Initially, experimental gene dataset (***U, C***) are discretized using Transitional State Discrimination method (*TSD*) [9], where *U*, the universe of discourse contains *n* genes and *C*, the condition attribute set contains *m* samples. In *TSD* [9], discretization factor f_{ij}, based on which the dataset is discretized, is computed for sample $C_j \in C$ of gene $g_i \in U$, using (1), for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$.

$$f_{ij} = \frac{M_i[C_j] - \mu_i}{\delta_i} \tag{1}$$

Where, μ_i and δ_i, the mean and standard deviation respectively of gene g_i and $M_i[C_j]$ is the value of sample C_j in gene g_i. Then mean (N_i) of negative values and mean (P_i) of positive values of each gene g_i are computed and discretized to one of *N* (here, *N* = 5) fuzzy linguistic terms using (2).

$$f_{ij} = \begin{cases} 'VL' & if\ f_{ij} \le N_i \\ 'L' & if\ N_i < f_{ij} < 0 \\ 'Z' & if\ f_{ij} = 0 \\ 'H' & if\ 0 < f_{ij} < P_i \\ 'VH' & if\ f_{ij} \ge P_i \end{cases} \tag{2}$$

After discretization, the dataset is divided into two sets, one set (*MISS*) contains genes with missing value and other set (*NOMISS*) contains genes without missing value.

2.2 Formation of Correlated Gene Subsets

Let, the samples of genes are collected from d_1 normal and d_2 cancerous patients; so each gene contains d_1 normal and d_2 cancerous samples. Let, each gene $g_i \in NOMISS$

is represented as $g_i = \{g^n_{i1}, g^n_{i2}, \ldots, g^n_{id_1}, g^c_{i1}, g^c_{i2}, \ldots, g^c_{id_2}\}$, where g^n_{ij} for $j = 1, 2, \ldots, d_1$ are normal samples and g^c_{ik} for $k = 1, 2, \ldots, d_2$ are cancerous samples. Frequencies of discrete expression values for samples $\{g^n_{i1}, g^n_{i2}, \ldots, g^n_{id_1}\}$ and $\{g^c_{i1}, g^c_{i2}, \ldots, g^c_{id_2}\}$ of gene g_i are computed as $\{f^{ni}_{VL}, f^{ni}_{L}, f^{ni}_{Z}, f^{ni}_{H}, f^{ni}_{VH}\}$ and $\{f^{ci}_{VL}, f^{ci}_{L}, f^{ci}_{Z}, f^{ci}_{H}, f^{ci}_{VH}\}$ respectively, where f^{ni}_{VL} is the frequency of expression value '*VL*' in normal samples of gene g_i, similar meaning of other terms. Let $f^{ni}_{max} = max\{f^{ni}_{VL}, f^{ni}_{L}, f^{ni}_{Z}, f^{ni}_{H}, f^{ni}_{VH}\}$ and $f^{ci}_{max} = max\{f^{ci}_{VL}, f^{ci}_{L}, f^{ci}_{Z}, f^{ci}_{H}, f^{ci}_{VH}\}$. The gene subsets are formed as follows:

If f^{ni}_{max} and f^{ci}_{max} are computed from
(i) Same discrete expression value, say '*VL*' then the gene $g_i = \{g^n_{i1}, g^n_{i2}, \ldots, g^n_{id_1}, g^c_{i1}, g^c_{i2}, \ldots, g^c_{id_2}\}$ is placed in subset $GENE\_WHOLE_{VL}$ (abbreviated as GW_{VL}, used synonymously in the paper), same situation for other discrete values. Thus, five subsets GW_{VL}, GW_L, GW_Z, GW_H and GW_{VH} are formed. Each of these five subsets contains genes of *NOMISS,* where maximum frequency of discrete value occurs for same discrete value in both normal and cancerous samples.
(ii) Different discrete expression value, say f^{ni}_{max} occurs for '*VL*' and f^{ci}_{max} occurs for '*VH*'. In this case, the normal part $\{g^n_{i1}, g^n_{i2}, \ldots, g^n_{id_1}\}$ of g_i is placed in subset $GENE\_NORMAL_{VL}$ (abbreviated as GN_{VL}), same situation for other discrete values. And cancerous part $\{g^c_{i1}, g^c_{i2}, \ldots, g^c_{id_2}\}$ of g_i is placed in subset $GENE\_CANCER_{VH}$ (abbreviated as GC_{VH}), same situation for other discrete values. Thus, GN_{VL}, GN_L, GN_Z, GN_H and GN_{VH} are formed, each of which contains normal samples of genes whose maximum frequency discrete value differs from that of cancerous samples. Similarly, gene subsets containing only cancerous samples are formed which are GC_{VL}, GC_L, GC_Z, GC_H and GC_{VH}.

Thus, genes without missing value (i.e., set *NOMISS*) are partitioned into fifteen subsets. These subsets are formed according to the gene expression values of the dataset and each subset contains similar type of genes.

2.3 Similarity Measurement

Fifteen gene subsets are formed from the set *NOMISS* of genes without missing values. Each set contains the genes of similar type. On the other hand, the set *MISS* contains genes with missing values which need to be estimated as data preprocessing step of knowledge discovery. Each gene $g_j \in MISS$ can also be thought of as $g_j = \{g^n_{j1}, g^n_{j2}, \ldots, g^n_{jd_1}, g^c_{j1}, g^c_{j2}, \ldots, g^c_{jd_2}\}$ with some g^n_{jk} and g^c_{jl} may be missed, for $k = 1, 2, \ldots, d_1$ and $l = 1, 2, \ldots, d_2$ which are estimated by the proposed method.

The same process is applied to compute the maximum frequency of discrete expression values in both normal and cancerous samples of gene $g_j \in MISS$. If maximum frequency occurs in both types of samples for same expression value, say *'VL'*, then g_j is associated with gene subset GW_{VL}. But if maximum frequency occurs for different expression values, say '*VL*' and '*VH*' for normal type and cancerous type respectively, then normal samples $\{g^n_{j1}, g^n_{j2}, \ldots, g^n_{jd_1}\}$ of g_j is associated with GN_{VL} and cancerous samples $\{g^c_{j1}, g^c_{j2}, \ldots, g^c_{jd_2}\}$ of gene g_j is associated with GC_{VH}. Thus

each gene $g_j \in MISS$ is either (*a*) associated with any one subset of {GW_{VL}, GW_L, GW_Z, GW_H, GW_{VH}} or (*b*) normal portion of it is associated with any one of {GN_{VL}, GN_L, GN_Z, GN_H, GN_{VH}} and cancerous portion of it is associated with any one of {GC_{VL}, GC_L, GC_Z, GC_H, GC_{VH}}. Similarity of gene gj in case of (*a*) is discussed below; whereas same logic is applied in case of (*b*), which is not described redundantly.

(*a*) Now, associated gene subset with real values is partitioned using clustering algorithm [10] which provides optimal set of K-clusters. Centroids of all K-clusters are computed and discretized using (2). Thus, K-centroids of (d_1+d_2)-tuples, one for each cluster is obtained. Let, the centroids of cluster T is $CENTRE_T = \{C^n_{t1}, C^n_{t2}, \ldots, C^n_{td_1}, C^c_{t1}, C^c_{t2}, \ldots, C^c_{td_2}\}$, for $t = 1, 2, \ldots, k$, where, C^n_{tj} is the mean (centroid) of j-th normal samples in cluster T, for $j = 1,2,\ldots,d_1$ and C^c_{tj} is the mean (centroid) of j-th cancerous samples of cluster T, for $j = 1, 2, \ldots, d_2$. Now the similarity S_{jT} of gene $g_j \in MISS$ with cluster T is the number of samples having discrete value equals to that of centroid of T, define by following function:

```
Function: Similarity (gene g_j, cluster T {
  /* gene g_j = {g^n_j1, g^n_j2, ..., g^n_jd1, g^c_j1, g^c_j2, ..., g^c_jd2} and centroid of
     Cluster T is CENTRE_T = {C^n_t1, C^n_t2, ...., C^n_td1, C^c_t1, C^c_t2, ...., C^c_td2} */
  S_jT =0; //similarity between gene g_j and cluster T
  For i = 1 to d_1
     If (g^n_ji = C^n_ti )
        S_jT = S_jT + 1;
  For i = 1 to d_2
     If (g^c_ji = C^c_ti )
        S_jT = S_jT +1;
  Return (S_jT);
}
```

Thus, similarity of g_j with all K clusters are obtained and if S_{jP} is maximum for $1 \le P \le K$ and the missing g^n_{jq} will be estimated by C^n_{pq}, $1 \le q \le d_1$ and missing g^c_{jr} will be estimated by C^c_{pr}, $1 \le r \le d_2$. Thus, all gene g_j with missing values are estimated. The overall algorithm of missing value is described below:

```
Algorithm. MISSING_VALUE_IMPUTATION (U, C)
Input:  U is the gene dataset containing n genes, C is
        the sample set containing m samples.
Output: Gene dataset with estimated missing values.

Step1: Discretized dataset U with N number of discrete
       values, using (1) and (2).
Step2: Create gene set MISS with missing values and
       NOMISS without missing values.
```

Step3: Find maximum frequency f_1 and f_2 of discrete values of a gene of *NOMISS* for normal and cancerous samples.
Step4: If (f_1 and f_2 occurs for same discrete value) then
Put whole gene into one of N gene subsets associated with respective discrete value.
Else, Put normal and cancerous part of gene separately into two subsets of 2×N gene subsets
Step5: Repeat Step3 and step4 for all genes of *NOMISS*.
Step6: Take a gene from *MISS* and select its associated set among 3×N gene subsets based on samples behavior.
Step7: Perform clustering algorithm [10] on selected gene subset and find optimal number of clusters.
Step8: Select cluster to which considering gene has maximum similarity.
Step9: Impute missing value of the sample of the gene by the corresponding value in centroid of the selected cluster.
Step10: Repeat Step6 to Step9 for all genes of *MISS*.
Step11: Stop.

3 Experimental Results and Performance Evaluation

Experimental studies presented here provide an evidence of effectiveness of the proposed missing values imputation method on experimental microarray dataset. Experiments were carried out on large number of different kinds of microarray data, few of which are summarized below:

(i) Leukemia dataset: Training dataset consists 27 ALL and 11 AML samples, over 7129 human genes. The raw data is available at http://www-genome.wi.mit.edu/cgi bin/cancer/datasets.cgi.
(ii) Diffuse Large B-cell Lymphoma (DLBCL) dataset: The dataset contains 58 DLBCL and 19 Follicular Lymphoma (FL) samples, over 7129 genes. Raw data are available at http://www-genome.wi.mit.edu/cgibin/cancer/datasets.cgi.
(iii) Lung Cancer dataset: Training dataset contains 16 samples labeled as "MPM" and 16 samples labeled as "ADCA" with around 12533 genes. The raw data are available at http://www. chestsurg.org/microarray.htm.
(iv) Prostate Cancer dataset: Training dataset consists 52 "relapse" and 50 "non-relapse" samples, over 12600 genes. The raw data are available at http://www-genome. wi. mit.edu/mpr/prostate.

The microarray gene expression dataset is divided into two subsets where one contains without missing value related genes and other contains randomly created

missing values related genes with randomly created missing positions where predicted values are imputed by the proposed method. The performance of the proposed method with compare to some traditional missing value estimation methods (i.e., Zero Imputation, Row Average and KNN) are measured by Normalize Root Mean Square Error (*NRMSE*). The *NRMSE* is computed for different methods using (3).

$$NRMSE = \frac{1}{std\_dev(X_{known})} \sqrt{\frac{\sum_{i=1}^{n} \left(X_{predict} - X_{known}\right)^2}{n}} \tag{3}$$

Where, X_{known} is the original gene expression value and $X_{predict}$ is the estimated value of the proposed algorithm, $std\_dev(X_{known})$ is the standard deviation of original expression values and n is the total number of missing values. The number n is computed randomly according to 5%, 10%, 15%, 20%, 25% and 30% of missing values and *NRMSE* are computed for all methods. The result shows that *NRMSE* produced by the proposed algorithm are significantly less than the other methods for different dataset, which confirms the potentiality and superiority of the proposed method. The KNN technique is applied for different values of *K* and taking the best results among them. The outstanding estimation ability of proposed missing value imputation method is important due to the use of correlation structure of gene expression values, novel clustering algorithm and similarity factor measurement. The other methods depends how far sample values of number of genes are closed with the missing value ignoring the characteristic of expression values of genes, which might be different. But the proposed method depends on the characteristics of gene expression values. The Fig. 1 to Fig. 4 shows the visual proof of several dataset by computing NRMSE for several methods.

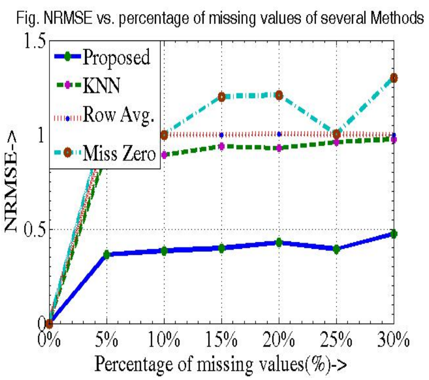


Fig. 1. Comparison of NRMSE value with different methods for Leukemia dataset

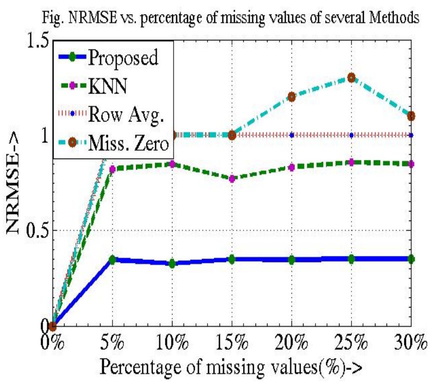


Fig. 2. Comparison of NRMSE value with different methods for DLBCL dataset

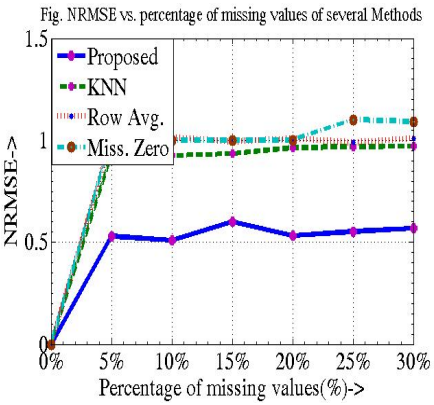


Fig. 3. Comparison of NRMSE value with different methods for Lung cancer dataset

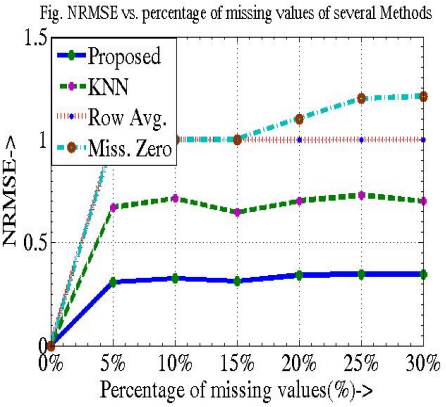


Fig. 4. Comparison of NRMSE value with different methods for Prostate cancer dataset

All the algorithms are implemented using Mat lab 7.8.1 version. Also all comparison figures are drawn using Mat lab 7.8.1 version. The comparisons are performed on PC (Intel(R) Core(TM) 2 Duo T5750 2.0 GHz, 2.0 GHz with 2.0 GB of Ram).

4 Discussions and Conclusion

Systematic Missing data can bring lots of difficulties in microarray data analysis simply because most existing methods were not designed for them and without imputing these values properly, the result will be erroneous. So this is most important preprocessing step to deal with missing values in the context of the integration of post-genomic experimental dataset. The existing statistical techniques incorporate with the context estimate missing values without measuring the correlation between normal and cancerous samples, which may give some valuable information about the nature of the gene. To this circumstance, the proposed method is conceptual and computational challenge totally depends on expression values and independent on number of genes. To measure the correlation between the normal and cancerous samples, the dataset is split into small subsets which help to estimate the missing values effectively. The performance of proposed method is analyzed on four common publicly available microarray dataset and compared the accuracy with Zero-impute, Row-average and KNN in terms of the NRMSE which shows the goodness of proposed method.

References

1. DeRisi, J., et al.: Use of a cDNA microarray to analyse gene expression patterns in human cancer. Nat. Genet. 14(4), 457–460 (1996)

2. Luo, J., Yang, T., Wang, Y.: Missing Value Estimation for Microarray Data Based On Fuzzy C-means Clustering. In: Proceedings of the Eighth International Conference on High-Performance Computing in Asia-Pacific Region (2005)
3. Butte, A.J., Ye, J.: Determining Significant Fold Differences in Gene Expression Analysis. In: Pac. Symp. Biocomput., vol. 6, pp. 6–17 (2001)
4. Alizadeh, A.A., et al.: Distinct Types of Diffuse Large B-Cell Lymphoma Identified by Gene Expression Profiling. Nature 403, 503–511 (2000)
5. Schafer, J.L., Graham, J.W.: Missing data: our view of the state of the art. Psychol. Methods 7, 144–177 (2002)
6. Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., Altman, R.B.: Missing value estimation methods for DNA microarrays. Bioinformatics 17, 520–525 (2001)
7. Huynen, M., Snel, B., Lathe, W., Bork, P.: Genome Res. 10, 1204–1210 (2000)
8. Zhang, S., Zhang, J., Zhu, X., Qin, Y., Zhang, C.: Missing Value Imputation Based on Data Clustering. Transactions on Computational Science (TCOS) 1, 128–138 (2008)
9. Velarde Cristina, C., Escudero, R., Zaliz, R.R.: Boolean Networks: A Study on Microarray Data Discretization. In: ESTYLF 2008, Cuencas Mineras, Mieres, Langreo, pp. 17–19 (2008)
10. Pati, S.K., Das, A.K.: Cluster Analysis of Microarray Data Based on Similarity Measurement. International Journal of Bioinformatics Research 3(2), 207–213 (2011) ISSN: 0975-3087

A Strategy Pool Adaptive Artificial Bee Colony Algorithm for Dynamic Environment through Multi-population Approach

Digbalay Bose[1], Subhodip Biswas[1], Souvik Kundu[1], and Swagatam Das[2]

[1] Dept. of Electronics and Communication Engineering,
Jadavpur University, Kolkata 700 032, India
[2] Indian Statistical Institute, Kolkata- 700108, India
{digbose92,subho.opto.placis91,sk210892}@gmail.com,
swagatam.das@isical.ac.in

Abstract. Swarm Intelligence is based on developing metaheuristics that are modeled on certain life-sustaining principles exhibited by the biotic components of the ecosystem. There has been a surge in interest for nature inspired computing for devising more efficient models that can find solution to real-world problems using minimal resources at disposal. In this paper, an enhanced version of Artificial Bee Colony algorithm have been proposed that takes on the task of finding the optimal solution in a continuously changing (dynamic) solution space by incorporating a pool of varied perturbation strategies that operate on a multi-population group and synergizing the strategy pool with a set of diversity-inclusion techniques that help to maintain population diversity.

1 Introduction

A plethora of real-world problems encountered in our daily life are in the state of continuous change. This ongoing changing phenomenon led to the advent of dynamic optimization problems (DOPs). Instead of treating them as an entirely separate domain of problems, they can be thought of as a collection of problems that may either change their functional behavior or the problem itself after a pre-defined interval of time has elapsed. In the recent past, researchers have devised standard numerical benchmarks to evaluate the performance of an algorithm when subjected to change. Literature presents us instances of the gradual improvement in the change detection strategies like hyper-mutation, clustering, memory scheme, multi-population approach and so on. The difficulty faced by algorithms in a dynamic environment attributes to change-detection and the sub-sequent control mechanism that it uses to nullify the effect of this change so as to maintain its performance in the new environment. Diversity-maintaining techniques like anti-convergence [9], introduction of random individuals [9], exclusion principle [9] etc. are the need-of-the-hour for diversification in dynamic problems.

Swarm literature speaks volumes of many algorithms that draw their functionality from the foraging behavior of insect colonies like honeybees [6], etc. Artificial Bee

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 611–619, 2012.

Colony (ABC) algorithm proposed by D. Karaboga [7-8], is based on the minimal foraging model of honey bees used in nectar collection from the adjoining environment of their honeycomb. The main advantage of algorithms modeled on insect colony is the underlying principles of swarm behavior like self-organization, task allocation, task force decentralization and communication among the individuals..

Due to the advantages perceived, the ABC algorithm is the optimizer of our choice. But even with the foreseen efficiencies ABC could not find its crux when presented with the dynamic set of problems which led the authors to integrate multiple diversity schemes within the ABC framework including the sub-population maintained. From henceforth the proposed ABC variant shall be referred to as PSpABsC (Perturbation Strategy Pool Artificial Bee sub-Colony) algorithm for convenience.

The rest of the paper is organized in the following way: Section 2 details the basic framework of our proposed algorithm followed by the experimental result and analysis in Section 3. In Section 4 the conclusions drawn by the authors from this adaptive approach to ABC algorithm have been discussed followed by directions for future research that may be undertaken.

2 The ABC Framework

The framework of the basic ABC algorithm is explained in self-explanatory steps:

- Set initial trial vectors as food sources to begin the search
- Employ each employed forager to each Food source to prevent crowding

Repeat

- Perform exploitation of the food source and memorize the location of food source if it is better than already existing source.
- Discard the previous source's location if newly found one is better
- Convey the information to the waiting onlooker bees and recruit them
- Based on Roulette-wheel selection allow the onlookers to select food sites
- Perform similar exploitation followed by Greedy Selection until all the onlookers have been garrisoned at the food sources at their disposal.

Stop if the termination criterion is met

3 The PSpABsC Algorithm

The modifications imbibed within the PSpAsBC framework are discussed in this section with reference to the classical ABC algorithm elucidating the justification for doing so, which led to a novel hybridized approach to dynamic optimization.

3.1 Perturbation Strategy Pool (PSp)

Originally ABC algorithm was designed to operate on static conditions for continuous real parameter problems. But as the parameter space grew more complex the need

was felt for modified perturbation strategy that could generate controlled exploitation-exploration in the search. Studying the movement undertaken by bees to search for better food source, the authors have proposed replacing the basic strategy of perturbation, that displaces the bee's location by randomly sampling a difference between its present and a randomly selected location and multiplying it by a scale factor that is uniformly generated in the range of [-1, 1], we propose a strategy pool that uses the spatial information of the neighborhood of its source. A three-strategy pool is used that incorporated the following perturbation techniques

1. **Pert/source-rand/1 :** Basic strategy used that can be formulated as

$$\theta_i^j = \theta_i^j + \left(\theta_i^j - \theta_k^j\right) \times rand\ (0,1) \tag{1}$$

2. **Pert/rand/1:** This strategy perturbs the position of the forager by imparting a direction guided along the difference vector between two random food sources

$$\theta_i^j = \theta_{rand,1}^j + \left(\theta_{rand,2}^j - \theta_{rand,3}^j\right) \times rand\ (0,1) \tag{2}$$

3. **Pert/rand-*f*ittest/1:** This is a greedy variant that guides the foragers towards the fittest food source found and helps to perform exploitation. It is denoted as

$$\theta_i^j = \theta_i^j + \left(\theta_{fittest}^j - \theta_{rand}^j\right) \times rand\ (0,1) \tag{3}$$

In formulae (1), (2) and (3) the θ_i, θ_k refer to the i^{th} and k^{th} food source, and i, k takes integral values in the range [1, FN] where FN denotes the Food Number and j is the parameter to be perturbed. The randomly sampled vectors must be different.

A perturbation probability is sampled from Gaussian distribution.

$$Prob_i^{Pert} = Gaussian\ (0.5, 0.1) \tag{4}$$

The perturbation probability is rounded off so as to lie in the range [0, 1] and is multiplied by the no. of perturbation strategies (3 in this case) and a value obtained as

$$Strategy = \lceil Prob_i^{Pert} \times 3 \rceil \tag{5}$$

The Perturbation Strategy is Pert/source-rand/1, Pert/rand/1 and Pert/rand-*f*ittest/1 if the value of *Strategy* is 1, 2 and 3 respectively. The underlying philosophy behind the use of the perturbation strategy pool is to guide the forager into spatially unexplored regions that may hold better reward in store for them.

3.2 Multi-population Approach via Sub-colony

The multi-population approach was initially proposed for niching techniques where a population is divided into different sub-species each occupying different niche aided by geophysical factors. Use of species based techniques proved beneficial since they successfully maintain the genetic diversity over succeeding generations. Hence we use the multi-population approach whereby the bee colony is divided into multiple sub-colonies each comprising of foragers. Parallel execution of the sub-population takes place in each cycle such that they track different optima and the problem of local trap is successfully overcome. Ideal assumption is that one of them finds the

global optima before the termination criteria is met. Since diversity is a key issue in case of dynamic environments use of subpopulations can alleviate the problem of rapid convergence.

3.3 Diversity Scheme to Prevent Premature Convergence

To ensure that sufficient diversity is maintained in each subpopulation few individuals are not subjected to the modified ABC scheme but they are modified by certain probabilistic schemes. Two such approaches involve introduction of quantum individuals and a modified variant of Brownian individuals. For each subpopulation a particle generating parameter i.e. C_i is maintained, which is generated in every iteration; i denotes the i^{th} subpopulation. If $C_i=0$, only Brownian individuals are generated for that particular subpopulation by selecting two members of the subpopulation in a random manner and if C_i=1, quantum individuals are generated. This ensures every subpopulation an equal chance of being subjected to either of the two diversity generating schemes illustrated below:-

(a) Quantum Individuals: Quantum individuals derive their ideology from inherent uncertainty associated with their positions due to probabilistic determination method. This idea is thus utilized for generating individuals in the vicinity of the local best positions of each subpopulation. For generation of quantum individuals a cloud of radius $R_{quantum}$ is considered around the local best of each subpopulation.

(i) Obtain a radius value ***r_rand*** uniform generated in the range **[0, $R_{quantum}$].**

(ii) Generate a vector $\vec{V}$ having all elements derived from a normal distribution of mean 0 and standard deviation1.

(iii) Distance of the vector $\vec{V}$ is obtained from the origin as ***dist_origin,*** where distance is calculated as :-

$$dist\_origin = \sqrt{\sum_{j=1}^{D} V_j^2} \tag{6}$$

(iv) The quantum individual's updated position is given by:-

$$\overrightarrow{Loc_{best}} + \left(\frac{r\_rand}{dist\_origin}\right) * \vec{V} \tag{7}$$

(b) Modified Brownian individuals: Original Brownian individuals, inspired by the randomized motion of suspended particles, are generated around the local best position

$$\overrightarrow{Loc_{best}} + \overrightarrow{Norm(0, sigma)} \tag{8}$$

Here Loc_{best} is the position of the local best and *Norm* is a vector whose elements are obtained from a normal distribution having standard deviation *sigma*.

We propose a novel way of generating Brownian individuals in which the vector *Norm* Is replaced by a vector *Levy_mut* in which the elements are generated using levy distribution [10].Since Levy distribution has an infinite second moment, and a

much longer tail as compared to Cauchy and Gaussian distribution, it imparts greater diversity when applied in case of brownian individuals as compared to standard normal distribution. Thus the newly modified Brownian individual will have a position

$$\overrightarrow{Loc_{best}} + \overrightarrow{Levy\_Mut} \tag{9}$$

3.4 Exclusion Principle

The segmentation of the entire population into multiple sub-populations is based on the principle that different subpopulations must explore multiple regions of the search space preventing a situation of convergence of multiple subpopulations around local optima, resulting in an inefficient tracking of the changing environment. To eliminate this problem of convergence, the Euclidean distance between the best members of two subpopulations is calculated and if it is less than a minimum threshold called exclusion radius ***(R_{excl})***, then the subpopulation whose best member has worse functional value is reinitialized randomly in the search space.

3.5 Improvement Based Re-initialization Scheme

During the course of optimization process, an individual may get trapped in a local minimum which adversely affects the entire search process. Here we have proposed an improvement based re-initialization scheme, similar to the commonly used aging mechanism which takes into account the improvements in the functional value of each member of every subpopulation. For each member of every subpopulation a trial counter is maintained, which is denoted by ***improve_set$_{ij}$*** where *i* stands for the i^{th} subpopulation and *j* stands for the j^{th} member of the i^{th} subpopulation. Each entry improve_***set$_{ij}$*** is incremented whenever there is an improvement in the functional value of the jth member of the i^{th} subpopulation. If the counter value is less than a minimum threshold called ***min_improve***, then the corresponding individual is reinitialized. The entire subpopulation is reinitialized only when the maximum improvement corresponding to the subpopulation members is less than ***min_improve***. This re-initialization scheme is applied to all the subpopulations except the subpopulation containing the global best individual.

3.6 Change Detection and Response to Change

An efficient algorithm for dynamic environments must be able to detect the occurrence of change in the functional landscape. For the purpose of change detection a randomly generated vector $\overrightarrow{test\_change}$ is generated initially and is evaluated in every generation. The vector $\overrightarrow{test\_change}$ is not subjected to any optimization process and the change in its functional values in the current and the previous generation is considered as a change in the environment. Whenever the change is detected in the environment, all parameters associated with the algorithm are reinitialized again.

4 Experimental Set-ups and Analysis

4.1 Numerical Benchmark

There is an abundance of test problems in case of dynamic optimization problems on which algorithms can be tested to assess their efficiency and robustness. In our paper, our algorithm along with the contender algorithms have been tested on the Generalized Dynamic Benchmark Generator (GDBG) obtained from the technical report under IEEE Congress on Evolutionary Computation, 2009. Overall there are 6 functions included under GDBG, with each 7 change instances associated with each function.

4.2 Parameter Settings and Contestant Algorithms

The parametric setting for our proposed PSpABsC algorithm is given below

- Number of subpopulations:- *Npop=5*
- Number of members of each subpopulation:- *mem=12*
- Quantum radius:- $r_{quantum}=1$
- Exclusion Radius:- $R_{excl}=0.03$
- Minimum improvement rate:- *min_improve=8*
- Dimension of each member:- *10*

The contestant algorithms selected to compete against our proposed PSpABsC algorithm are DASA [14], dopt-aiNet [15], CPSO [16] and DynDE [17]. Parametric settings for these algorithms have been kept according to their standard literature that produces the best possible results. Each of the competing algorithms are made to run for 25 independent test runs and the offline error recorded in each case is the mean of the errors procured in three different environments. The standard deviation is calculated according to the formula specified in the technical report and is specified as

$$\text{STD}=\sqrt{\frac{1}{runs*num\_change-1}\sum_{i=1}^{runs}\sum_{j=1}^{num\_change}(E_{i,j}^{last}(t)-Avg\_mean)^2} \quad (10)$$

where *runs* is set to 25, *num_change* is 3 for different change instances and *Avg_mean* is the average mean error and is calculated as

$$\text{Average mean (Avg\_mean)}=\sum_{i=1}^{runs}\sum_{j=1}^{num\_change}E_{i,j}^{last}(t)/(runs*num\_change) \quad (11)$$

4.3 Experimental Results

The data obtained from experimental set-up is tabulated in Table 1 given below.

4.4 Analysis of Results

From Table 1 we can infer that PSpABsC algorithm has successfully managed to outperform its competitors on 34 out of 49 possible cases which accounts to an

overwhelming majority of 69.4 %. Thus the improvements suggested in Section 3 have managed to create an efficient optimizer that performs significantly well on dynamic test bed. The presence of diversity maintaining individuals coupled with the niche based subpopulation technique helps to guide the optimizer in an ill-scaled, rugged environment preventing local trap. Even the modified levy-based Brownian individuals create a wide spread repulsion mechanism when change is detected.

Table 1. Mean offline error and standard deviation values achieved by the PSp-ABsC and other algorithms tested on gdbg functions for 25 test runs

Test Functions	Algorithm	Error	T1	T2	T3	T4	T5	T6	T7
F1 (Number of peaks = 10)	DASA	Average	0.1854	4.1802	6.3706	0.4873	2.5485	2.3442	4.8452
		(Std)	(1.2501)	(9.0716)	(10.7128)	(1.9504)	(4.8002)	(8.6685)	(8.9662)
	DynDE	Average	0.0732	**2.5567**	5.4245	0.1263	1.5651	1.3115	4.1137
		(Std)	(2.9567)	**(8.4313)**	(9.2485)	(0.9426)	(4.6461)	(6.2511)	(8.5249)
	dopt-aiNET	Average	0.1353	5.8667	4.2545	5.3563	4.4356	9.9407	4.2110
		(Std)	(1.0061)	(10.2772)	(8.1828)	(8.9414)	(5.5545)	(15.8214)	(8.6873)
	CPSO	Average	0.0351	2.7185	4.1315	**0.0944**	1.8698	1.1569	4.5401
		(Std)	(0.4262)	(6.5230)	(8.9947)	**(0.7855)**	(4.4910)	(4.8054)	(9.1194)
	PSpABsC	Average	**0.0436**	3.0173	**3.8764**	5.7194	**1.3341**	**1.2069**	**3.9127**
		(Std)	**(0.3425)**	(4.4738)	**(5.2314)**	(4.4123)	**(2.4123)**	**(3.4785)**	**(5.1175)**
F1 (Number of peaks= 50)	DASA	Average	0.4425	4.8661	8.4247	0.5853	1.1832	2.0728	7.8412
		(Std)	(1.3911)	(7.0052)	(9.5682)	(1.0901)	(2.1818)	(5.9719)	(9.0558)
	DynDE	Average	0.3286	4.6547	6.4641	0.1412	1.0162	**0.9859**	6.2513
		(Std)	(1.5224)	(6.3453)	(9.3523)	(0.5914)	(2.6489)	**(4.8631)**	(9.0651)
	dopt-aiNET	Average	0.3644	4.7485	**5.2531**	2.6565	2.8641	6.8330	4.8172
		(Std)	(0.9725)	(6.7580)	**(6.6830)**	(5.9773)	(4.1579)	(11.8790)	(6.4528)
	CPSO	Average	0.2624	3.2792	6.3198	0.1255	0.8481	1.4821	6.6467
		(Std)	(0.9362)	(5.3034)	(7.4420)	(0.3859)	(1.7790)	(4.3932)	(7.9411)
	PSpABsC	Average	**0.1375**	**4.3342**	7.8394	**0.0778**	**0.7612**	1.7524	**4.1752**
		(Std)	**(0.3345)**	**(6.3753)**	(9.7813)	**(0.1632)**	**(1.4321)**	(5.8765)	**(6.5432)**
F2	DASA	Average	3.3017	25.6105	18.9904	1.4512	49.6022	2.1182	3.8752
		(Std)	(8.7885)	(83.2124)	(67.8204)	(3.8311)	(112.4132)	(5.2912)	(8.1285)
	DynDE	Average	1.3627	13.0179	11.9214	0.7842	**20.7842**	2.1845	**2.4235**
		(Std)	(5.0315)	(48.2532)	(45.7054)	(2.2248)	**(64.5341)**	(3.9643)	**(7.1031)**
	dopt-aiNET	Average	**0.0984**	8.1209	17.9979	1.0652	101.384	6.5192	3.7385
		(Std)	**(0.0291)**	(14.3832)	(62.2259)	(2.8269)	(134.5180)	(13.8172)	(7.9542)
	CPSO	Average	1.2475	10.1055	10.2725	0.5664	25.1424	1.9871	3.6510
		(Std)	(4.1780)	(35.0601)	(33.4527)	(2.1371)	(64.2500)	(5.2175)	(6.9274)
	PSpABsC	Average	0.2314	**7.1327**	**10.1311**	**0.4412**	52.7156	**1.7152**	7.7143
		(Std)	(0.7745)	**(9.4315)**	**(30.4462)**	**(1.7754)**	(134.3215)	**(4.1713)**	(12.2347)
F3	DASA	Average	**15.7025**	824.389	688.358	435.488	697.210	626.1120	433.252
		(Std)	**(67.1131)**	(204.0035)	(298.0124)	(441.2120)	(315.4223)	(460.6211)	(380.2234)
	DynDE	Average	21.2512	792.457	**635.614**	341.701	749.265	519.550	415.324
		(Std)	(73.6549)	(255.6163)	**(342.7753)**	(419.8116)	(280.9181)	(438.2467)	(390.3450)
	dopt-aiNET	Average	810.830	1078.70	1073.41	1031.54	1023.90	1186.94	1061.33
		(Std)	(66.1085)	(64.1245)	(64.9950)	(274.7490)	(57.8713)	(292.2960)	(110.0980)
	CPSO	Average	137.527	855.139	765.966	430.620	859.704	753.039	653.703
		(Std)	(221.0011)	(161.0024)	(235.8834)	(432.2391)	(121.5581)	(361.7855)	(334.4892)
	PSpABsC	Average	**13.3734**	847.153	896.314	**157.667**	**523.714**	**428.114**	**227.445**
		(Std)	**(43.1034)**	(244.0025)	(404.357)	**(202.613)**	**(49.8842)**	**(334.8822)**	**(334.7456)**

Table 1. (*continued*)

F4	**DASA**	Average (Std)	5.6001 (26.5331)	65.6105 (160.0281)	53.6158 (140.0155)	3.8512 (4.2258)	118.212 (178.2506)	2.9812 (7.5942)	27.4432 (90.2513)
	DynDE	Average (Std)	1.8616 (5.7531)	39.5923 (98.6312)	23.4921 (94.5314)	0.9691 (3.1723)	44.6713 (121.7162)	1.5624 (6.2149)	6.5213 (26.5951)
	dopt-aiNET	Average (Std)	1.4227 (4.5459)	122.440 (201.627)	98.6688 (196.6950)	4.2632 (9.7255)	304.566 (203.2430)	12.6491 (55.8367)	52.9010 (130.593)
	CPSO	Average (Std)	2.6771 (7.0552)	37.1512 (99.4352)	36.6711 (97.1805)	**0.7926 (2.775)**	67.1702 (130.3059)	4.8814 (15.3965)	12.7924 (19.2105)
	PSpABsC	Average (Std)	**1.3936 (5.0523)**	**20.9394 (32.8612)**	**2.0520 (4.1624)**	2.0523 (4.7752)	**33.2880 (94.6789)**	**1.4231 (5.4465)**	**5.0413 (8.7944)**
F5	**DASA**	Average (Std)	0.9551 (3.4310)	2.0199 (4.0504)	0.9494 (3.3135)	**0.3928 (1.6184)**	2.3052 (6.3610)	0.4671 (1.7346)	1.1128 (3.7620)
	DynDE	Average (Std)	2.9929 (6.8831)	2.9481 (4.7179)	2.9125 (5.3886)	1.3796 (2.4199)	8.4378 (12.1132)	2.3049 (3.6182)	**0.5214 (0.7135)**
	dopt-aiNET	Average (Std)	40.8943 (221.212)	34.4531 (119.896)	34.9420 (115.025)	120.637 (293.542)	943.223 (633.318)	480.305 (610.801)	219.461 (427.817)
	CPSO	Average (Std)	1.8559 (5.1812)	2.8791 (6.7875)	3.403 (6.4480)	1.0954 (4.8651)	7.9869 (13.8170)	4.0535 (8.3719)	6.5278 (22.8129)
	PSpABsC	Average (Std)	**0.7784 (2.4762)**	**2.0023 (3.8745)**	**0.4487 (2.1425)**	44.7321 (87.3456)	**1.9742 (5.3417)**	**0.3145 (0.8843)**	35.6672 (53.1142)
F6	**DASA**	Average (Std)	8.8752 (13.319)	37.128 (122.0147)	26.7341 (98.4018)	9.7442 (22.0541)	37.9102 (118.0146)	13.3481 (57.4802)	17.7422 (36.7159)
	DynDE	Average (Std)	6.0471 (11.0458)	**30.2205 (62.2093)**	**19.3782 (67.3585)**	8.8731 (26.6683)	43.3514 (136.9062)	**12.1784 (25.2617)**	15.3644 (21.0744)
	dopt-aiNET	Average (Std)	20.4434 (79.3230)	391.195 (395.4350)	456.443 (405.0380)	83.9698 (220.1770)	845.862 (251.208)	482.201 (434.421)	372.470 (394.6628)
	CPSO	Average (Std)	6.7254 (9.9747)	31.5738 (63.5115)	27.1358 (83.9873)	9.2742 (24.2344)	71.5704 (160.3211)	23.6757 (51.5521)	32.5842 (76.9105)
	PSpABsC	Average (Std)	**5.9973 (10.3342)**	78.2158 (118.7734)	84.1145 (112.1304)	**7.1392 (17.9745)**	**23.5174 (97.3466)**	26.7756 (58.9945)	**13.4914 (47.8219)**

5 Conclusion

It was inferred from the outline of our research paper that for complex landscapes the multi-population approach has its own unique advantages over existing techniques and when this is combined with diversity inclusion techniques through instantiation of random Brownian, quantum individuals or techniques like exclusion principle or anti-convergence. The proposed PSpABsC algorithm is based on this perspective taken to dynamic optimization problems. To the best of the authors' knowledge this paper provides the first instance of testing ABC algorithm on GDBG benchmark and opens up an avenue for future works on incorporation of swarm-based techniques on dynamic problems to test the strength of adaptation and decentralization of swarms.

References

1. Talbi, E.G.: Metaheuristics-From Design to implementation. John Wiley and Sons (2009)
2. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, Piscataway, NJ, pp. 1942–1948 (1995)

3. Engelbrecht, A.: Fundamentals of Computational Swarm Intelligence. John Wiley and Sons, UK (2005)
4. Clerc, M., Kennedy, J.: The particle swarm-explosion, stability and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation, 58–73 (2002)
5. Kennedy, J., Eberhart, R.C.: Swarm Intelligence. Morgan Kauffman, San Francisco (2001)
6. Pham, D.T., Ghanbarzadeh, A., Koc, E., Otri, S., Rahim, S., Zaldi, M.: The Bees Algortihm – A Novel Tool for Complex Optimization Problems. In: Proceedings of IPROMS 2006 Conference, pp. 454–461 (2006)
7. Karaboga, D.: An idea based on honey bee swarm for numerical optimization, Technical Report TR 06, Erciyes University, Engg. Faculty, Computer Engineering Department (2005)
8. Karaboga, D., Basturk, B.: A powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm. Journal of Global Optimization 39(3) (2007)
9. Blackwell, T., Branke, J.: Multi swarms, Exclusion and Anti convergence in dynamic environments. IEEE Transactions on Evolutionary Computation, 459–472 (2004)
10. Lee, C.Y., Yao, X.: Evolutionary Programming using Mutation based on the Levy probability distribution. IEEE Transactions on Evolutionary Computation, 1–13 (2004)
11. Branke, J.: Memory enhanced evolutionary algorithms for changing optimization problems. In: IEEE Congress on Evolutionary Computation (1999)
12. Yang, S., Ong, Y.S., Jin, Y.: Evolutionary Computation in Dynamic and Uncertain Environment. Springer, Berlin (2007)
13. Li, C., Yang, S., Nguyen, T.T., Yu, E.L., Yao, X., Jin, Y., Beyer, H.G., Suganthan, P.N.: Benchmark Generator for CEC 2009 Competition on Dynamic Optimization, Univ. of Leicester, Univ. of Birmingham, Nanyang Technological University, Tech. Rep. (2008)
14. Korosec, P., Silc, J.: The differential ant-stigmergy algorithm applied to dynamic optimization problems. In: IEEE Congress on Evolutionary Computation, pp. 407–410 (2009)
15. de Franca, F.O., Von Zuben, F.J.: A dynamic artificial immune algorithm applied to challenging benchmarking problems. In: IEEE Cong. on Evo. Computation, pp. 423–430 (2009)
16. Yang, S., Li, C.: A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments. IEEE Trans. on Evo. Comp. 14(6) (2010)
17. Mendes, R., Mohais, A.S.: DynDE: a differential evolution for dynamic optimization problems. In: IEEE Congress on Evolutionary Computation, pp. 2808–2815 (2005)

Multistage Covariance Matrix Adaptation with Differential Evolution for Constrained Optimization

Shantanab Debchoudhury, Rohan Mukherjee, and Rupam Kundu

Department of Electronics and Telecommunication Engineering,
Jadavpur University,Kolkata-700032
{sdch10,rohan.mukherjii,rupam2422}@gmail.com

Abstract. Single Objective minimizations often involve simultaneous satisfaction of a number of conditions, known as constraints. MCMADE proposes a two-stage algorithm having an initial CMA or Covariance Matrix Adaptation phase and a subsequent Differential Evolution strategy in the second phase. The two phases are synchronized using a stagnate parameter. To handle the constraints, a simple penalty function, without any penalty parameter has been employed which adds the margin of violations to the fitness value of each particle in the landscape. MCMADE has been tested on the problem set specified by the CEC 2010 benchmark.

1 Introduction

A number of single objective problems in real life involve constraints that need to be satisfied along with the optimization process. MCMADE or Multistage Covariance Matrix Adaptation with Differential Evolution is a two-stage strategy that uses a population to initially adapt on the lines of Covariance Matrix Adaptation technique (CMA) and then subsequently evolve with the help of a basic differential evolution model. The Constraint handling technique is a parameter free penalty function which just adds up the violations obtained from the equality and inequality constraints associated with the function landscape. The results have been tested on CEC 2010 benchmark on Constrained Optimization. We proceed to our discussion on the finer details only after we mention briefly the intricacies involved with a Constrained Optimization problem and the popular methods to deal with these constraints.

1.1 Constrained Optimization and Optimization Techniques

A Constrained Optimization problem is defined in the search space *S* as:

Minimize $f(\vec{X}), \vec{X} = \{X_1, X_2, ..., X_n\}, \quad \vec{X} \in S$

Subject to:

$$g_i(\vec{X}) <= 0 \qquad i = 1, 2, ..., n_1$$

$$h_j(\vec{X}) = 0 \qquad j = 1, 2, ..., n_2$$

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 620–627, 2012.

The former are known as inequality constraints and the latter equality constraints. These conditions have to be met along the process of optimization.

Different procedures have been adapted to handle the constraints. Superiority of Feasible Solutions [1] classifies the feasible solutions according to the minimum fitness value, and the infeasible solutions according to the lesser amount of violations. Infeasible solutions are given lesser priority in selection compared to feasible ones. Other techniques include the $\mathcal{E}$-Constraint (EC) process [2], the stochastic ranking process [3] and use of Penalty functions. The latter method is widely popular and penalty functions, thus, have varied forms, static as well as adaptive [4], [5], [6]. ECHT, proposed by R. Mallipeddi and P.N. Suganthan, uses an ensemble of all these four techniques to efficiently handle constraints [7]. With this background on the dynamics of a Constrained Optimization problem, we proceed to discuss the fundamental elements of our proposed algorithm, MCMADE.

2 MCMADE Algorithm

MCMADE basically is a controlled conglomeration of two evolutionary schemes, - the Covariance Matrix Adaptation Evolution Strategy (CMA-ES), and the Differential Evolution. Before we go into the finer details of the MCMADE Algorithm we briefly look at the features of these two techniques.

2.1 CMA-ES Variant Used in Stage I of MCMADE

The Covariance Matrix Adaptation and Evolution Strategy has been built on the basis of important features like maintenance of easy generalizability, property of invariance towards linear changes, rotation and transformations. The λ individuals chosen as population size are updated from the g-th generation to the (g+1)th generation according to the following equation:

$$x_k^{(g+1)} \sim N(x_W^{(g)}, \sigma^{(g)^2} C^{(g)}). \tag{1}$$

where $N(\mu, C)$ implies a normally distributed random vector having mean μ with the covariance matrix being C. The update equation for the individuals can be approximately given as:

$$N(x_W^{(g)}, \sigma^{(g)^2} C^{(g)}) \sim x_W^{(g)} + \sigma^{(g)}\ B^{(g)} D^{(g)} N(0, I). \tag{2}$$

The term $x_W^{(g)}$ is called the recombination point of the individuals at g-th generation and is given as the weighted mean of the individuals

Now the adaptation of the mutation parameters is done by two processes which we describe now:

A. Adaptation of Covariance Matrix. The adaptation of the covariance matrix $C^{(g)}$ depends on three factors; $p_c^{(g+1)}$ or the evolution path, $x_W^{(g)}$ the recombination point, and on the separation of the present parents with the recombination point

B. Adaptation of Global Step Size. While $p_c^{(g+1)}$ is a evolution path, $p_\sigma^{(g+1)}$ provides a parallel path in order to perturb the global step size $\sigma^{(g+1)}$

The two adaptation procedures are key to the functioning of CMA-ES algorithm. Now initial conditions include setting of $p_c^{(g+1)}$ and $p_\sigma^{(g+1)}$ to 0. Covariance matrix is set so that $C^{(0)}$ is set to *I*. $x^{(0)}$ and $\sigma^{(0)}$ are problem dependent though $\sigma^{(0)}$ is set to 0.6 in our case. Population size λ is set as $\lambda = 50 + \lfloor 3.\ln(n) \rfloor$, n being the dimension, while number of parents for recombination is given by $\mu = \lfloor \lambda / 2 \rfloor$. Due to constraints on paper size, we do not delve further into the intricacies of the algorithm and the steps involved can easily be referred to from [8].

Thus we conclude the beginning stage of MCMADE algorithm which is a CMA-ES variant.

2.2 Classical DE Algorithm Used in Stage II of MCMADE

Differential Evolution (DE) first proposed by Storn and Price [9] remains a pillar of evolutionary algorithm. DE is a three staged process with mutation or forming of a donor from the target population according to certain schemes, crossover or mixing or donor and target individuals to form a trial individual, and finally selection which involves choosing the functionally fitter of the trial and target individuals for incorporation into the main population.

For the mutation stage, we have used "DE/rand/1" scheme with scaling factor set to a random number between 0 and 1. The crossover probability is set to 0.9.

DE is an extremely popular minimization scheme and we restrict ourselves from discussing further about the algorithm, which has been extensively covered in [10].

2.3 The Multistage Synchronization and Its Utility

The two widely important minimization schemes of CMA-ES and DE have been discussed in Sections 2.1 and 2.2 respectively. Now our objective is unfilled if we do not synchronize the two stages properly. Over employment of CMA strategy would lead to high rate of convergence up to a stage when the vectors are unable to adapt, while high scope for DE perturbation leads to insufficient exploitation as compared to CMA strategy. Thus proper balancing of the two methods yield a successful outcome.

In MCMADE, CMA technique is used in Stage I. This leads to, as expected high exploitation leading to a fast path of convergence with appreciable avoiding of local traps. However, at a certain point of time, exploitation is saturated and the mean around which new particles are generated, fails to produce any significant exploration. Herein lies the utility of a Stage II where we have led the particles to a DE perturbation scheme. Having failed to adapt any more, the particles are now given sufficient time to evolve with whatever past information they have acquired. A DE environment is suitable for such purpose. The multistage strategy also helps to

efficiently utilize the time allotted for the task, as CMA update schemes have low complexity leading to extremely fast initial performance.

Stagnate Parameter and Justification for the Value Set. MCMADE makes use of a stagnate parameter to determine when Covariance Matrix adaptation becomes ineffective. It is of utmost importance as it prepares the population for onset of Differential Evolution. Higher value of Stagnate Parameter would mean unnecessary wastage of allotted time, while a lower value of Stagnate Parameter would limit the scope of Covariance matrix adaptation. In MCMADE, the maximum allowable value for the Stagnate Parameter has been set to 20.

Working mechanism of Stagnate Factor:

```
if (fbest_g==fbest_{g+1})
     Stagnate_Parameter=Stagnate_Parameter+1
 else
      Stagnate_Parameter=0
end if
if(Stagnate_Parameter==20)
       Pass population to DE
end if
```

Here $fbest_g$ denotes the best fitness value in the g-th generation. Thus we see that once best fitness value fails to update for 20 consecutive iterations, the entire population is subjected to differential evolution.

2.4 Constraint Handling Technique Used in MCMADE

MCMADE makes use of a parameter-free simple penalty function to update the fitness value at each function evaluation. Let us assume that the fitness function $f(\vec{X})$ be subjected to n_1 number of inequality constraints g_i where i=1 to n_1, and n_2 number of equality constraints h_j, where j=1 to n_2

$$g_i(\vec{X}) <= 0 \text{ for } i=1,2,\ldots\ldots,n_1$$
$$h_j(\vec{X}) = 0 \text{ for } j=1,2,\ldots\ldots,n_2$$

In such a case the total violation for an inequality constraint would be represented by

$$Ineq\_viol = \sum_{i=1}^{n_1} \max(0, g_i(\vec{X})), \tag{3}$$

while the total violation for the equality constraints would be given as

$$Eq\_viol = \sum_{j=1}^{n_2} \max(\delta, \left|h_j(\vec{X})\right|). \tag{4}$$

Here δ is a tolerance parameter whose value is set to 0.0001.The penalty applied is thus given by:

$$Total\_penalty = Ineq\_viol + Eq\_viol\ . \tag{5}$$

Thus the final value of a vector-individual is given by:

$$f(\vec{X}) = f(\vec{X}) + Total\_penalty\ . \tag{6}$$

The penalty function is thus a straightforward sum and does not contain any penalty parameter. In this manner constraints can be efficiently handled for the problem in question.

3 Experimental Settings, Results and Discussions

All runs have been taken on a machine with Intel® Core™ i3-350M 2.26GHz processor, RAM 3 GB and using MATLAB 2010a. MCMADE has been tested on the 18 test functions of CEC 2010 Competition and Special Session on Single Objective Constrained Real-Parameter Optimization [11]. The results obtained have been compared with those tested by algorithms like Takahama and Sakai's εDEag algorithm, Mallipeddi and Suganthan's ECHT scheme. Other papers employing the same benchmark include [12], [13]. Results of both 10 dimension and 30 dimension problems have been tabulated under Table 1. It is to be noted that 10D problems have been allotted maximum of 2e5 functional evaluations while 30D problems have been allotted maximum of 6e5 functional evaluations. Also the fitness values tabulated include addition of violations, as obtained through equation (6).

Table 1. Results of 10D and 30D problems compared with other papers

		10D PROBLEMS			30D PROBLEMS		
Functions		ε DEag	ECHT	MCMADE	ε DEag	ECHT	MCMADE
C01	Best	**-7.47e-01**	**-7.47e-01**	-7.41e-01	**-8.21e-01**	**-8.21e-01**	-8.04e-01
	Mean	-7.43e-01	-7.39e-01	-6.46e-01	-8.19e-01	-7.890e-01	-6.79e-01
	STD	8.43e-03	8.76e-03	1.11e-01	6.33e-02	2.8e-02	2.04e-01
C02	Best	**-2.28e00**	**-2.28e00**	**-2.28e00**	-2.16e00	-2.17e00	**-2.28e00**
	Mean	-2.25e00	-2.27e00	-2.27e00	-2.14e00	-1.97e00	-2.264e00
	STD	9.66e-03	8.90e-03	1.79e-02	1.87e-03	5.99e-01	2.38e-03
C03	Best	**0.00e00**	**0.00e00**	**0.00e00**	2.87e01	4.33e-21	**0.00e00**
	Mean	0.00e00	0.00e00	0.00e00	2.90e01	1.05e02	0.00e00
	STD	0.00e00	0.00e00	0.00e00	6.44e00	8.77e01	0.00e00
C04	Best	-9.98e-06	**-1.00e-05**	**-1.00e-05**	5.77e-03	-3.30e-06	**-3.33e-06**
	Mean	-9.95e-06	-1.00e-05	-1.00e-05	8.90e-03	-1.01e-06	-3.33e-06
	STD	2.20e-07	0.00e00	0.00e00	9.92e-03	2.11e-01	0.00e00

Table 1. (*continued*)

		10D PROBLEMS			30D PROBLEMS		
Functions		$\mathcal{E}$ DEag	ECHT	**MCMADE**	$\mathcal{E}$ DEag	ECHT	**MCMADE**
C05	Best	**-4.84e02**	**-4.84e02**	**-4.84e02**	-4.50e02	-2.13e02	**-4.84e02**
	Mean	-4.83e02	-4.14e02	4.84e02	-4.48e02	-1.03e02	-4.84e02
	STD	1.64e-13	9.07e01	0.00e00	3.15e00	1.88e02	0.00e00
C06	Best	**-5.79e02**	**-5.79e02**	**-5.79e02**	-5.27e02	-2.92e02	**-5.32e02**
	Mean	-5.79e02	-5.57e02	-5.79e02	-5.26e02	-1.14e02	-5.32e02
	STD	7.06e-02	5.11e01	0.00e00	9.88e-01	1.02e02	0.00e00
C07	Best	**0.00e00**	**0.00e00**	**0.00e00**	3.23e-15	**0.00e00**	**0.00e00**
	Mean	0.00e00	1.62e-01	0.00e00	4.45e-15	2.48e-01	0.00e00
	STD	0.00e00	8.91e-01	0.00e00	2.24e-15	7.7e-01	0.00e00
C08	Best	**0.00e00**	**0.00e00**	**0.00e00**	3.40e-15	**0.00e00**	**0.00e00**
	Mean	1.33e01	2.78e01	0.00e00	9.06e-15	3.55e02	0.00e00
	STD	5.59e00	7.83e00	0.00e00	1.85e-13	1.02e02	0.00e00
C09	Best	**0.00e00**	**0.00e00**	**0.00e00**	7.12e-16	**0.00e00**	**0.00e00**
	Mean	0.00e00	2.38e-01	0.00e00	1.83e01	4.8e01	0.00e00
	STD	0.00e00	1.02e00	0.00e00	6.02e01	2.98e02	0.00e00
C10	Best	**0.00e00**	**0.00e00**	**0.00e00**	3.33e01	**0.00e00**	**0.00e00**
	Mean	0.00e00	2.11e00	0.00e00	3.33e01	6.34e01	0.00e00
	STD	0.00e00	8.35e00	0.00e00	5.49e-01	9.97e01	0.00e00
C11	Best	**-1.5e-03**	**-1.5e-03**	**-1.5e-03**	-3.14e-04	-3.96e-04	**-4.83e-04**
	Mean	-1.5e-03	-4.57e-03	-1.5e-03	-2.84e-04	2.9e-03	-4.83e-04
	STD	0.00e00	1.86e-02	0.00e00	3.11e-05	7.06e-03	0.00e00
C12	Best	-5.70e02	**-1.99e-01**	**-1.99e-01**	-1.98e-01	**-1.99e-01**	**-1.99e-01**
	Mean	-2.86e02	-1.65e02	-2.75e02	3.77e02	-2.4e01	-2.90e01
	STD	2.99e02	3.04e02	1.45e02	4.31e02	1.43e02	3.44e02
C13	Best	**-6.84e01**	**-6.84e01**	-6.47e01	-6.63e01	**-6.79e01**	-6.18e01
	Mean	-6.68e01	-6.45e01	-6.40e01	-6.51e01	-6.42e01	-5.84e01
	STD	3.89e-5	3.77e00	6.99e-01	8.78e-01	2.77e00	2.05e00
C14	Best	**0.00e00**	**0.00e00**	**0.00e00**	7.61e-14	**0.00e00**	**0.00e00**
	Mean	0.00e00	7.14e05	0.00e00	8.01e-13	3.21e05	0.00e00
	STD	0.00e00	5.04e06	0.00e00	8.88e-13	8.00e05	0.00e00
C15	Best	**0.00e00**	**0.00e00**	3.37e00	**2.16e01**	2.01e09	**2.16e01**
	Mean	4.57e-01	7.66e13	3.59e-01	2.16e01	3.11e11	7.57e01
	STD	9.92e-01	8.00e13	1.50e-01	5.22e00	6.12e11	1.10e02
C16	Best	**0.00e00**	**0.00e00**	1.97e-01	**0.00e00**	**0.00e00**	1.09e00
	Mean	7.85e-01	9.17e-02	6.64e-01	3.89e-21	0.00e00	1.11e00
	STD	8.68e-01	6.23e-02	4.55e-01	9.09e-19	0.00e00	4.43e-02
C17	Best	1.01e-17	**0.00e00**	4.81e-46	2.56e-01	**0.00e00**	1.38e-33
	Mean	1.88e-01	1.20e-01	4.16e-34	8.33e00	4.47e-01	5.38e-30
	STD	2.01e-01	4.13e-01	7.59e-34	5.51e00	4.01e-01	6.71e-30
C18	Best	6.39e-20	**0.00e00**	**0.00e00**	1.32e00	**0.00e00**	**0.00e00**
	Mean	1.05e-17	0.00e00	0.00e00	8.81e01	0.00e00	5.38e-32
	STD	3.23e-18	0.00e00	0.00e00	1.97e02	0.00e00	9.52e-32

The above results show that MCMADE efficiently tackles constrained optimization problems. The figures below show the convergence graphs in case of some 10D and 30D problems. The graphs show that MCMADE is a reliable as well as fast procedure resulting in quick convergence in most cases.

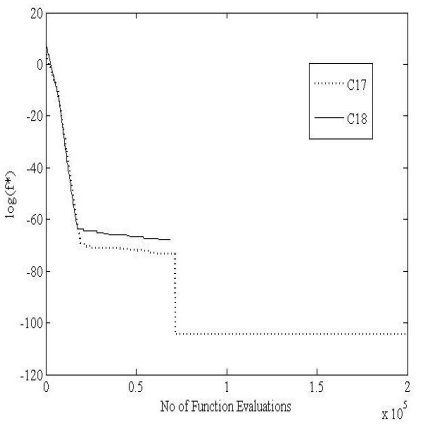


Fig. 1. Convergence graph for 10D Problems C17 and C18. (best solution)

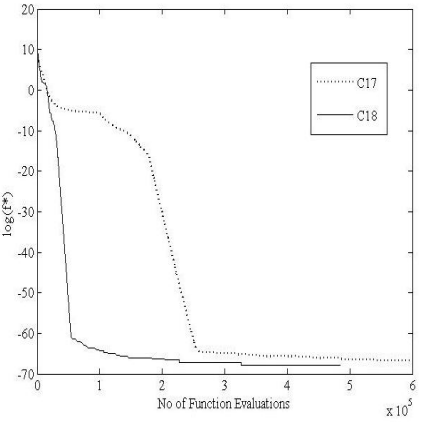


Fig. 2. Convergence graph for 30D Problems C17 and C18. (best solution)

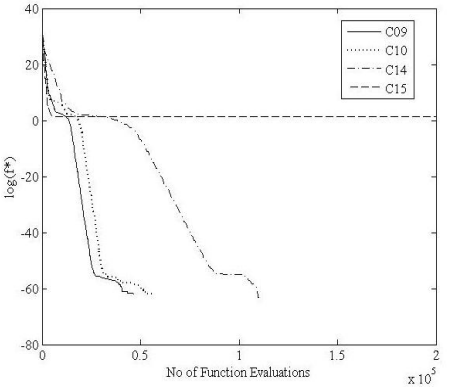


Fig. 3. Convergence graph for 10D Problems C09, C10, C14 and C15. (best solution)

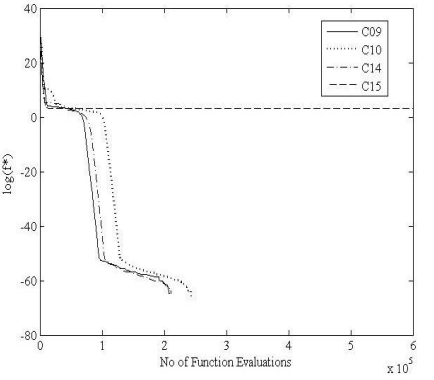


Fig. 4. Convergence graph for 30D Problems C09, C10, C14 and C15. (best solution)

4 Conclusions

Before concluding our discussion on MCMADE we recapitulate in a nutshell the positive aspects of this algorithm.

First and foremost, irrespective of the involvement of a dual-stage procedure, MCMADE remains true in terms of performance in case of majority of problems. The second important aspect is the speed of operation. CMA-ES or Stage I of

MCMADE is an especially fast process, and due to involvement of only single population 50-member population the DE or Stage II is also appreciably fast and reliable. The third aspect is the fact that, the penalty function used as constraint handling technique is extraordinarily simply and does not involve any penalty parameters to scale the violations involved. Moreover, the synchronization of the two stages is also much simple yet efficient, involving only a stagnate parameter whose value increases every time an ineffective adaptation is detected.

Thus it is evident that MCMADE acts a powerful tool in the field of Constrained Optimization, not only in terms of simplicity but also in terms of efficiency and reliability that helps to tackle the manifold challenges posed through the varied nature of the problem environment.

References

[1] Deb, K.: An efficient constraint handling method for genetic algorithms. Computer Methods in Applied Mechanics and Engineering 186, 311–338 (2000)

[2] Takahama, Sakai: Constrained Optimization by the Constrained Differential Evolution with Gradient-Based Mutation and Feasible Elites. In: IEEE Congress on Evolutionary Computation, Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada, pp. 1–8 (2006)

[3] Runarsson, T.A., Yao, X.: Stochastic ranking for constrained evolutionary optimization. IEEE Transactions on Evolutionary Computation 4, 284–294 (2000)

[4] Farmani, R., Wright, J.A.: Self-Adaptive Fitness Formulation for Constrained Optimization. IEEE Transactions on Evolutionary Computation 7, 445–455 (2003)

[5] Tessema, Yen: A self adaptive penalty function based algorithm for constrained optimization. In: Yen, Lucas, Fogel, Kendall, Salomon, Zhang, Coello, Runarsson (eds.) Proceedings of the 2006 IEEE Congress on Evolutionary Computation, July 16-21, pp. 246–253. IEEE Press, Vancouver (2006)

[6] Coello, C.A.C.: Use of a self-adaptive penalty approach for engineering optimization problems. Computers in Industry 41(2), 113–127 (2000)

[7] Mallipeddi, Suganthan: Ensemble of Constraint Handling Techniques. IEEE Transactions on Evolutionary Computation, available online

[8] Hansen, N., Kern, S.: Evaluating the CMA Evolution Strategy on Multimodal Test Functions. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN VIII. LNCS, vol. 3242, pp. 282–291. Springer, Heidelberg (2004)

[9] Price, Storn, Lampinen: Differential Evolution – A Practical Approach to Global Optimization. Springer, Berlin (2005)

[10] Das, S., Suganthan, P.N.: Differential Evolution – A Survey of the state-of-the-art. IEEE Transactions on Evolutionary Computation 15(1), 4–31 (2011)

[11] Mallipeddi, Suganthan: Problem Definitions and Evaluation Criteria for the CEC 2010 Competition on Constrained Real-Parameter Optimization. Nanyang Technological University, Singapore (2010)

[12] Polakova, Tvrdik: Various Mutation Strategies in Enhanced Competitive Differential Evolution for Constrained Optimization. In: Differential Evolution (SDE), IEEE Symposium Series on Computational Intelligence (2011)

[13] Sardar, S., Maity, S., Das, S., Suganthan, P.N.: Constrained Real Parameter Optimization with a Gradient Repair based Differential Evolution Algorithm. In: Differential Evolution (SDE), IEEE Symposium Series on Computational Intelligence (2011)

Evolutionary and Immune Algorithms Applied to Association Rule Mining

Danilo Souza da Cunha and Leandro Nunes de Castro

Mackenzie Presbyterian University, Natural Computing Laboratory (LCoN),
R. Piauí, 130, Consolação, São Paulo, Brazil
danilocunha85@gmail.com, lnunes@mackenzie.br

Abstract. Association rule mining is one of the most important data mining tasks. It corresponds to the determination of rules that associate items to other items in a data set, where the items are attributes in transactional databases. Although evolutionary algorithms have been used in this task for some time, there are few applications of immune algorithms to such problem. This paper presents one typical genetic algorithm plus two clonal selection algorithms applied to association rule mining under the perspective of several measures of interest.

Keywords: Association Rule Mining, Evolutionary Algorithms, Artificial Immune Systems, Data Mining.

1 Introduction

It is surprising the increase in the amount of information, products and services available nowadays. With increased bandwidth, the growth in the number of Internet access and new websites for small, medium and large companies has become an everyday reality. This results in a very competitive and more attentive market to customer needs, without forgetting to offer convenience and ease when it comes to closing a deal. As examples, one can mention a huge number and variety of travel packages (cruises, tours, ecotourism, etc.), cell phones, books, videos, DVDs, CDs, lab products, service providers, clothing, home products, computers, shoes, etc.

This paper provides an investigation into the use of a genetic algorithm, two types of Artificial Immune Systems (the traditional CLONALG, called CLONALG1 and CLONALG2), and the well-known Apriori rule mining algorithm to the problem of association rule mining.

2 Association Rule Mining

Association Rules (ARs) were originally proposed to assist the strategic decision making of market managers [1], [2]. Their initial objectives were: to help deciding which items should be sold; to design sales strategies; and to organize advertisements on the shelves, always aiming at maximizing profit.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 628–635, 2012.

Association rule mining, originally known as market-basket analysis, is one of the main data mining tasks [3]. It is a descriptive task that uses unsupervised learning and focuses on the identification of associations of items that occur together in a given data set [4].

2.1 A Formal Model

A transaction is defined as a set of items that occur together. In the scenario described in the original market-basket analysis, items in a transaction are those that were purchased together by a user. An association rule is a rule of the form $A \rightarrow C$, where A and C are itemsets. A is called the body (or antecedent) and C is called the head (or consequent) of the rule: $A \cap C = \emptyset$. The rule means that the presence of (all items) A in a transaction implies the presence of (all items) in the same transaction C with some associated probability [1]. The original algorithm introduced in [1] is limited to extracting association rules with only one item in the antecedent. Given a set of transactions T, it is interesting to generate all rules that satisfy certain additional constraints in two ways:

1. *Syntactic constraints*: the number of items that appear in a rule is limited; and
2. *Support constraints*: involve limitations on the number of transactions in T that support the rule, with support being defined as the number of transactions in T that contain A and C simultaneously.

2.2 Measures of Interest

The measures of interest, *confidence* and *support*, proposed in [1] and [2], are the most studied in the literature and were designed to assess the quality of a rule [4]. The support of an AR is a measure of its frequency in the set of transactions:

$$\textit{Support}\ (A \rightarrow C) = P\ (A \cup C) = (\text{Frequency of } A \text{ and } C/\text{Total transactions}). \quad (1)$$

The confidence of a rule, on the other hand, is a measure of the satisfiability of a rule when its antecedent part is found in T; that is, from all the occurrences of A, how many times C also occurs:

$$\textit{Confidence}\ (A \rightarrow C) = P\ (C \mid A) = (\text{Frequency of } A \text{ and } C/\text{Frequency of A}). \quad (2)$$

While confidence is a measure of the strength of a rule, the support corresponds to its statistical significance. One motivation for defining a threshold for the support comes from the fact that, for practical reasons, only rules above a minimum support threshold value are sufficiently interesting for a user [1].

Some researchers have been working to quantify the value or interest of a given rule. What it means to be useful or interesting is strongly dependent upon the application domain and linked to the user needs. To apply an algorithm to a database is easy; the real challenge lies in finding relevance in the knowledge extracted by this algorithm [5]. The *Apriori Algorithm* and those that derive from it became standard for association rule mining in large databases. To discover the occurrence of frequent itemsets, the algorithm performs multiple iterations through the data.

3 Evolutionary Design of Association Rule Mining

This section describes the basic evolutionary design principles for mining association rules. It provides the general concepts, representation used for the association rules, measures of interest, and the main genetic operators (*selection, crossover, mutation* and *hypermutation*).

3.1 General Concepts

The intersection of A and C in a rule $A \rightarrow C$ is always empty [4]. The use of evolutionary and immune algorithms for mining association rules follows their original steps [6], [7]:

- *Definition of a representation for the candidate solutions* (*individuals* or *cells* of the population): define the type of representation that will be used to encode each individual or each cell of the population;
- *Definition of an evaluation function*: define the evaluation or fitness function that will be used to assess the quality of the rules proposed;
- *Definition of genetic operators*: choose the genetic operators that will allow the variation of individuals or cells in the population.

In the context of evolutionary algorithms for association rule mining, there are two basic approaches to represent the rules: 1) *Pittsburgh* [8], and 2) *Michigan* [9]. The Pittsburgh approach considers each individual in the population as a set of rules. This is suitable for classification problems, but the size of the chromosome may limit the number of rules generated [10]. In the Michigan approach each chromosome represents a single rule and the entire population corresponds to the set of rules. The Michigan approach has been most commonly used in problems of association rule mining by means of evolutionary and immune algorithms [7], [10], [11].

3.2 Representation

The Michigan approach proposed in [12] was adapted and used by [5] and [13] to generate association rules. In this proposal, each item is represented by two bits, as follows (Figure 1): 00 means items belonging to the antecedent part; 11 means items belonging to the consequent part; and 01 or 10 means items not belonging to the rule. The resulting rule in this case is B ^ E ^ H → A ^ F. Note that, in this representation, items are always connected using the AND, (^), operator.

A	B	C	D	E	F	G	H
11	00	01	10	00	11	10	00

Fig. 1. Individual representation that considers a chromosome as an association rule

An advantage of this type of representation is that rules of various sizes can be obtained. Its main disadvantage is the length of the chromosome, which is always twice the total number of items.

3.3 Fitness Function

The fitness functions often used in evolutionary algorithms to assess association rules are similar to those employed for classification rule mining. For instance, in [5] the authors measure the degree of *comprehensibility* and *interestingness* of a rule. The comprehensibility measure, or comprehensibility of Type 1, of the association rule, $C_1(R)$, is defined as:

$$C_1(R) = log(1 + |C|)/log(1 + |A \cup C|). \tag{3}$$

where $|C|$ and $|A \cup C|$ are the numbers of items in the consequent part and the entire rule, respectively. The interestingness measure of a rule, $I(R)$, is calculated as:

$$I(R) = (|A \cup C|/|A|) * (|A \cup C|/|C|) * (1-(|A \cup C|/|D|)). \tag{4}$$

where $|D|$ is the number of transactions in the database. In [12] to measure comprehensibility of a rule, or comprehensibility of Type 2, $C_2(R)$ is given by:

$$C_2(R) = log(1 + |C|) + log(1 + |A \cup C|). \tag{5}$$

A detailed description of the various measures of interest usually used in association rules can be found in [14].

3.4 Genetic Operators

Evolutionary algorithms usually use one of the following encoding schemes [6]: Boolean, categorical, integer or real-valued. For Boolean and categorical representations, standard crossover and mutation operators, such as single-point crossover and mutation, can be used [15], [16]. For learning integer and real-valued association rules, although standard genetic operators are also the most commonly used, there are special cases that depend on the representation (see [17] for examples of different operators).

4 Proposed Algorithms: Evolutionary and Immune

The genetic algorithm proposed here was based on the works of [5] and [13], with the difference that our algorithm applies elitism to preserve the current best solutions. The two versions of the artificial immune system (CLONALG1 and CLONALG2) follow the same immune principles as the original CLONALG [18], but introduce some modifications while building the association rules.

Pseudocode 1 provides a brief description of the elitist genetic algorithm, eGA, proposed to mining association rules, where P is a population of candidate rules, pc the crossover probability, pm the mutation probability, pe the elitism rate and **D** the database to be mined.

```
procedure [P] = eGA(pc,pm,pe,D)
  initialize P
  f := evaluate(P,D);
  P := select(P,f,pe);
  while not_stopping_criterion do,
  P := reproduce(P,f,pc);
  P := variate(P,pm);
       f := evaluate(P,D);
       P := select(P,f,pe);
       t := t+1;
  end while
end procedure
```

Pseudocode 1. Evolutionary algorithm for association rule mining.

Pseudocode 2 presents CLONALG1, where **D** is the database to build association rules, *max_it* the number of iterations, n_1 the number of high affinity cells to be selected for cloning, and n_2 the number of low affinity cells to be replaced.

The *deterministicSelection* method works like in the original CLONALG selecting n_1 best cells from the population based on their fitness values. The number of clones is proportional to their fitness: the higher the fitness, the higher the number of clones generated, and vice-versa.

```
procedure [P] = CLONALG1(D,max_it,n1,n2)
  initialize P
  t := 1;
  while t ≤ max_it do,
  f := evaluate(P);
  P1 := deterministicSelect(P,n1,f);
  C := clone(P1,f);
  C1 := mutate(C,f);
  f1 := evaluate(C1);
       P1 := select(C1,n1,f1);
       P := replace(P,n2);
       t ← t + 1;
  end while
end procedure
```

Pseudocode 2. Immune algorithm for association rule mining: version 1.

The second version of the immune algorithm used here, CLONALG2, proposes a probabilistic, instead of deterministic, selection mechanism. The selection method works differently from the original CLONALG by selecting probabilistically n_1 cells from the population according to their fitness. The number of clones is proportional to the fitness of the cells, like in CLONALG1. Such probabilistic selection is expected to generate and maintain a larger diversity in the population during the search. Pseudocode 3 summarizes CLONALG 2.

```
procedure [P] = CLONALG2(D,max_it,n1,n2)
  initialize P
  t := 1;
  while t ≤ max_it do,
  f := evaluate(P);
  P1 := ProbabiliticSelect(P,n1,f);
  C := clone(P1,f);
  C1 := mutate(C,f);
  f1 := evaluate(C1);
       P1 := select(C1,n1,f1);
       P := replace(P,n2);
       t ← t + 1;
  end while
end procedure
```

Pseudocode 3. Immune algorithm for association rule mining: version 2

5 Experimental Results

The Apriori, eGA, CLONALG1 and CLONALG2 algorithms were applied to the Boolean attributes of the SPECT Heart database of the University of California Machine Learning Repository, available in the following website: http://archive.ics.uci.edu/ml/datasets/SPECT+Heart.

The parametric configuration of the algorithms were as follows:

- Apriori: 30% minimum support;
- eGA: 100 individuals, 100 generations, $pc = 60\%$, $pe = 50\%$, and $pm = 1\%$;
- CLONALG1 and CLONALG2 have the same configuration: 100 cells, $n_1 = 20$, and $n_2 = 5$.

Table 1 shows the average and standard deviation of all measures of interest of the algorithms when applied to this problem. The results shown are the average and standard deviation over ten runs, of the five metrics (support, confidence, comprehensibility of type 1, comprehensibility of type 2, and interestingness), plus the number of unique rules and the processing time of each algorithm.

Table 1. Results for the Apriori, eGA, CLONALG1 and CLONALG2. It has averaged values of measure of interest builded by each algorithm.

	Apriori	eGA	CLONALG1	CLONALG2
Support	0.35 ± 0.04	0.37 ± 0.03	0.46 ± 0.02	0.37 ± 0.02
Confidence	0.65 ± 0.16	0.86 ± 0.05	0.94 ± 0.01	0.92 ± 0.01
Compreheensibility 1	0.54 ± 0.06	0.50 ± 0.05	0.50 ± 0.01	0.46 ± 0.02
Compreheensibility 2	0.14 ± 0.03	0.14 ± 0.01	0.13 ± 0.00	0.14 ± 0.01
Interestingness	0.35 ± 0.08	0.35 ± 0.08	0.30 ± 0.00	0.26 ± 0.03
Unique Rule	17 ± 0.00	1.60 ± 0.60	1.50 ± 1.50	6.40 ± 2.30
Processing Time	6.5s ± 0.00	4.5s ± 1.01	9.3s ± 1.13	9.3s ± 1.16

In absolute terms, the immune algorithms demonstrated to be superior to the other algorithms in relation to the standard measures of interest confidence and support. In relation to the other measures it was competitive, with very similar performances. To assess the relative performance of the algorithms, a paired t-test was made between each pair of algorithms, assuming a normal distribution of the data and a confidence level of 5%. For all cases, the null hypothesis was rejected, thus suggesting that these algorithms are superior to the others in terms of confidence and support. On the other hand, they are more computationally intensive than the Apriori and eGA.

6 Discussion

This paper proposed the use of evolutionary and immune algorithms for association rule mining. One genetic algorithm and two versions of a clonal selection algorithm were presented and evaluated in a single binary dataset, but taking into account several measures of interest. The comparison between Apriori, eGA, CLONALG1 and CLONALG2 suggested that the immune approaches infer rules with higher support and confidence. Apriori, by contrast, inferred more comprehensible (in terms of C1) and interesting rules.

Acknowledgments. The authors thank Capes, CNPq, Fapesp and MackPesquisa for the financial support.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules Between Sets of Items in Large. In: Proceedings of the 1993 International Conference on Management of Data (SIGMOD 1993), pp. 207–216 (1993)
2. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules. In: Proc. 20th Int. Conf. Very Large Data Bases, VLDB, pp. 487–499 (1994)
3. Han, J., Kamber, M.: Data Mining - Concepts and Techniques, 2nd edn. Morgan Kaufmann, New York (2006)
4. Cios, K.J., Pedrycz, W., Swiniarski, R.W.: Data Mining: A Knowledge Discovery Approach, vol. XV. Springer (2007)
5. Dehuri, S., Jagadev, A.K., Ghosh, A., Mall, R.: Multiobjective genetic algorithm for association rule mining using a homogeneous dedicated cluster of workstations. American Journal of Applied Sciences 3, 2086–2095 (2006)
6. Tanomaru, J.: Motivação, Fundamentos e Aplicações de Algoritmos Genéticos. In: II Congresso Brasileiro de Redes Neurais - III Escola de Redes Neurais, Curitiba (1995)
7. Mo, H., Xu, L.: Immune Clone Algorithm for Mining Association Rules on Dynamic Databases. In: Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence, pp. 202–206. IEEE Computer Society (2005)
8. Smith, S.F.: A learning system based on genetic adaptive algorithms. Pittsburgh University, Pittsburgh (1980)

9. Booker, L.B., Goldberg, D.E., Holland, J.H.: Classifier systems and genetic algorithms. Artificial Intelligence 40, 235–282 (1989)
10. Liu, B., Hsu, W., Mun, L.F., Lee, H.Y.: Finding interesting patterns using user expectations. IEEE Transactions on Knowledge and Data Engineering XI, 817–832 (1999)
11. Dong, X., Li, X.: An Immune Based Relational Database Intrusion Detection Algorithm. In: Proceedings of the 2009 Ninth International Conference on Hybrid Intelligent Systems – HIS 2009, pp. 295–300 (2009)
12. Ghosh, A., Nath, B.: Multi-objective rule mining using genetic algorithms. Information Sciences 163, 123–133 (2004)
13. Wakabi-Waiswa, P.P., Baryamureeba, V.: Extraction of interesting association rules using genetic algorithms. International Journal of Computing and ICT Research 2, 26–33 (2008)
14. Geng, L., Hamilton, H.J.: Interestingness measures for data mining: A survey. ACM Computing Surveys 38 (2006)
15. Deepa Shenoy, P., Srinivasa, K.G., Venugopal, K.R., Patnaik, L.M.: Evolutionary Approach for Mining Association Rules on Dynamic Databases. In: Whang, K.-Y., Jeon, J., Shim, K., Srivastava, J. (eds.) PAKDD 2003. LNCS (LNAI), vol. 2637, pp. 325–336. Springer, Heidelberg (2003)
16. Shenoy, P.D., Srinivasa, K.G., Venugopal, K.R., Patnaik, L.M.: Dynamic association rule mining using genetic algorithms. Intelligent Data Analysis 9(5), 439–453 (2005)
17. Mata, J., Alvarez, J.-L., Riquelme, J.-C.: Discovering Numeric Association Rules via Evolutionary Algorithm. In: Chen, M.-S., Yu, P.S., Liu, B. (eds.) PAKDD 2002. LNCS (LNAI), vol. 2336, pp. 40–51. Springer, Heidelberg (2002)
18. de Castro, L.N., Von Zuben, F.J.: Learning and Optimization Using the Clonal Selection Principle. IEEE Transactions on Evolutionary Computation, Special Issue on Artificial Immune Systems (IEEE) 6(3), 239–251 (2002)

Improving Adaptive Differential Evolution with Controlled Mutation Strategy

Sayan Basu Roy, Mainak Dan, and Pallavi Mitra

Department of Electronics and Telecommunication Engineering,
Jadavpur University, Kolkata, India
{sayanetce,mainak.dan,pallavi.mitra1992}@gmail.com

Abstract. In this paper we have proposed a DE variant, abbreviated by ADE_CM, to improve optimization performance of DE by imposing controlled mutation strategy. We also incorporated the concept of selective pressure to choose the random vectors in the selection of donor vector for each population. Basically we used DE/rand/1 and DE/target-to-best/1 schemes (with modifications using selective pressure) in the selection of donor using controlled mutation. The control parameter for mutation, linearly decreasing with generation, is the complement of the probability of selecting DE/target-to-best/1 in each generation. The algorithm is basically a trade-off between diversity and greediness. To improve diversity scaling factor is made adaptive and also a worst p% scheme is used in the difference vector of donor. ADE_CM is tested on 25 benchmark functions of CEC 2005 in 50 and 100 dimensions. Experimental results show that this algorithm outperforms many popular DE variants on most of the functions.

Keywords: Differential Evolution, Parameter Adaptation, Controlled Mutation, Selective Pressure.

1 Introduction

Global numerical optimization is a very important and developing field of research because many real world problems can be specified as optimization. In order to solve these problems, many Evolutionary Algorithms (EAs) have been developed throughout the last two decades. EAs are stochastic search methods that have been developed on basis of the evolutionary process realized in nature. The notion behind EAs is to modify gradually population candidate solutions by means of natural selection and biological evolution. Differential Evolution (DE) [1] developed by Storn and Price, emerged as a very simple but competitive form of EAs. DE is a population-based stochastic-direct-search method. The DE family of algorithms has found many applications in real-life optimization problems. Though during last decade, the research on DE has reached a formidable state, DE faces difficulties in large number of modern applications due to its search performance. The performance of DE is quite parameter-dependent as realized from both experimental [2] and theoretical studies [3]. The search performance of DE needs improvement to avoid difficulties in certain

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 636–643, 2012.

kind of function as described in [4]. To minimize parameter tuning problem, researcher tried to develop some adaptive algorithms which would be capable of self-tuning the parameters in every generation on basis of successful parameters who were able to generate more optimized population vectors in last generation. Adaptive DE algorithms are most recent trend and these algorithms have secured top ranks in various competitions held under the IEEE congress on Evolutionary computation (CEC) conference series. For detail Study, readers may go through Self-Adaptive DE (SaDE) [6], Adaptive DE with optional external Archive (JADE) [7], jDE [8] etc. In this paper we have introduced another DE variant with an intention to increase its search ability. In this proposed DE, the mutation has been controlled with a control parameter and scaling factor is adapted with help of previous successful scaling factors. This algorithm is named as improving Adaptive DE with controlled mutation strategy (ADE_CM) due to its own algorithmic components and characteristics.

2 ADE_CM Algorithm: Strategy and Parameter Adaptation

In this section we propose a new DE algorithm. These newly proposed components have been integrated with the classical DE family of algorithms. For detail study of Classical DE family, readers may consult [1].

2.1 Donor Selection through Controlled Mutation

The oldest and most successful DE mutation strategy is DE/rand/1/bin. Mezura-Montes *et al.* [5] have opined that incorporation of best solution information has some benefits over rand/1/bin strategy and used DE/target-to-best/1 in their algorithm. DE/target-to-best/*k* have faster convergence rate than DE/rand/*k* because incorporation of best solution, so far discovered guide the evolutionary search while the later has higher exploration capability than the former . However, best solution information may harm the exploration of sub search space, with a premature convergence. So initially exploring of the search subspace is more required. We have introduced a control parameter C which controls the donor selection for every target vector in each generation. C is linearly decreasing with increase in generation between 0.8 and 0.2. If rand(0,1) is greater than C, DE/target-to-best/1 policy is taken rather than DE/rand/1. This will enhance the chance of fast convergence near the optimum point.

2.2 Selective Pressure

In this scheme, r_1^i, r_2^i, r_3^i are chosen according to the linear distribution function proposed by Whitely [10]:

$$r_x := \left\lfloor \frac{\lambda}{2(\beta-1)} . \left(\sqrt{\beta^2 - 4.(\beta-1).U(0,1)}\right) \right\rfloor. \tag{8}$$

In this formula, λ is the total number of population, among those r_x is to be selected. β is the bias term which controls the selective pressure. In our algorithm, we have

selected $\beta = 1.5$. Selective Pressure is defined as the ratio of probabilities between selecting the fittest individual and selecting the individual with median fitness. Classical DE struggles with non-separable functions due to insufficient exploitation during selection of donor vector. To get rid of this difficulty, selective pressure is introduced here.

2.3 Worst $p\%$ Vector

Another modification is introduced for generating difference vector used in perturbation. We have created a group with worst $p\%$ vectors according to their functional value. The second vector in the difference term, i.e. the vector with index r_3^i is chosen from this group. Selective pressure is much more biased to the vectors with top-rank. This worst $p\%$ vector will diversify the population and explore the search subspace. When the algorithm reaches exploitation stage, the population vectors are in close proximity. Thus this policy does not affect the convergence procedure when the solutions are about to reach global optimum.

2.4 Parameter Adaptation

DE is more sensitive to the choice of scaling factor rather than crossover probability. For detail study, readers may go through [12]. So we have only used adaptation scheme only for F.

Scaling Factor. At each generation, the scaling factor (F_i) for every individual target vector is randomly produced from the Cauchy distribution with location parameter F_l and scaling parameter F_s according to the formula

$$F_i = randc(F_l, F_s). \tag{9}$$

The value of F_i's is truncated to 1 if distribution samples a value greater than 1. The value of Scaling Factor is re-sampled if a non-positive value is generated. S_F indicates the set of successful scale factors, so far generated in current generation, which produces a better trial vectors than the target vectors. Initially F_l and F_s are set to 0.5 and 0.15 respectively and then their values are updated in each generation as

$$\begin{aligned} F_l &= c_f.F_l + \left(1 - c_f\right).mean_p(S_F), \\ F_s &= c_f.F_s + \left(1 - c_f\right).mean_p(S_F), \end{aligned} \tag{10}$$

Where, the weight factor c_f varies randomly between 0.9 and 1. $mean_p(.)$ stands for the power mean defined as

$$mean_p(S_F) = \left(\sum_{x \,\epsilon\, S_F} x^n / |S_F|\right)^{1/n}, \tag{11}$$

where, $|S_F|$ denotes the cardinality of set S_F. In ADE_CM, we use $n = 1.5$ as it gives the best results on various test problems.

Crossover Probability. *CR* is a tuning element. High value of *CR* adopts more components from donor than the target vector. A controlled parameter is already used

to generate best donor vectors. We keep *CR* at a high value (at upper bound of *CR* i.e. 0.9), to incorporate more components of donor vector into the trial vector.

2.5 Pseudo Code of ADE_CM

```
Begin
Set F_l = 0.5; F_s = 0.15; CR = 0.9;
Create a random initial set of population {X⃗_{i,0} | i = 1,2,3,…,NP} of dimension D.
Calculate the fitness value of each individual population vector.
For G = 1 to gen_max
    S_F = φ; C = 0.8 – 0.6 · (G/gen_max);
    Sort the population vectors according to their fitness values.
    For i = 1 to NP
        Generate F_i = randc(F_l, F_s);
        Randomly choose r_1^i, r_2^i from current population using selective pressure such that r_1^i ≠ r_2^i ≠ i.
        Randomly choose r_3^i from the worst 20% population vectors such that r_1^i ≠ r_2^i ≠ i ≠ r_3^i.
        If C ≤ rand(0,1)
            V⃗_{i,G} = X⃗_{r_1,G} + F.(X⃗_{r_2,G} − X⃗_{r_3,G});
        Else
            V⃗_{i,G} = X⃗_{r_1,G} + F.(X⃗_{best,G} − X⃗_{i,G}) + F.(X⃗_{r_2,G} − X⃗_{r_3,G});
        End if
        Generate j_rand randomly from {1,2,….,D}.
        For j = 1 to D
            If  j = j_rand or rand(0,1) ≤ CR
                u_{j,i,G} = v_{j,i,G};
            Else
                u_{j,i,G} = x_{j,i,G};
            End if
        End for
        If f(U⃗_{i,G}) ≤ f(X⃗_{i,G})
            X⃗_{i,G+1} = U⃗_{i,G}
            S_F ← F_i;
        Else
            X⃗_{i,G+1} = X⃗_{i,G};
        End if
    End for
```

```
32.      c_f = 0.9 + rand(0,1) · 0.1;
33.      F_L = c_f · F_L + (1 − c_f) · mean_p(S_F);
34.      F_S = c_f · F_S + (1 − c_f) · mean_p(S_F);
35. End for
36. End
```

3 Experiment and Results

The ADE_CM algorithm is tested on 25 standard numerical benchmark functions from the special session and competition on real parameter optimization under the IEEE CEC 2005. Functions 1-5 are unimodal, functions 6-12 are basic multimodal, functions 13-14 are expanded multimodal and functions 15-25 are hybrid composite functions. For detail study on these functions, reader may see in [11].

Table 1. Mean and Standard Deviation of the Error Values for f_1–f_{10}(50D). The best entries are marked in boldface.

Func / Algorithm	f_1	f_2	f_3	f_4	f_5
	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
DE/rand/1/bin	8.9605e-005 (1.3478e-005)	4.0516e+003 (8.7997e+002)	5.6567e+07 (1.4526e+007)	1.0923e+004 (3.332e+003)	9.6015e+003 (7.0345e+002)
DE/target-to-best/1/bin	4.4563e-006 (8.7963e-007)	2.0365e+003 (1.2366e+003)	1.0030e+007 (6.4509e+006)	8.5655e+003 (2.7865e+003)	7.3479e+003 (1.2981e+003)
JADE	7.5613e-014 (2.8336e-014)	5.8645e-004 (5.3273e-006)	**8.6632e+004 (3.5567e+004)**	3.1760e+003 (3.9895e-001)	3.0499e+003 (6.4429e+002)
jDE	1.2546e-009 (6.1254e-010)	5.1088e+003 (2.4682e+003)	2.8845e+007 (4.4672e+006)	1.1104e+004 (3.3309e-001)	3.9856e+003 (4.7823e+02)
SaDE	6.3256e-011 (2.4565e-012)	2.0362e-003 (7.9685e-003)	8.7640e+005 (6.7825e+004)	8.9067e+004 (6.8719e+003)	6.0451e+003 (3.8971e+002)
DEGL	5.5612e-010 (8.5675e-011)	1.3206e-005 (9.4613e-06)	2.2958e+005 (1.1104e+005)	2.5547e+003 (1.8890e+002)	5.9984e+003 (8.0976e+002)
ADE_CM	**5.6843e-014 (0.0000e+000)**	**3.6815e-008 (3.8240e-008)**	9.4416e+005 (2.7796e+005)	**2.4608e+000 (1.3786e+000)**	**1.9463e+003 (9.9583e+001)**
Func / Algorithm	f_6	f_7	f_8	f_9	f_{10}
	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)	Mean (Std)
DE/rand/1/bin	4.8869e+001 (1.0095e+001)	8.6659e+003 (8.8857e-002)	2.1167e+001 (6.8895e-002)	3.4519e+002 (5.8671e+001)	3.8820e+002 (1.8834e+001)
DE/target-to-best/1/bin	4.6672e+001 (1.2984e+001)	8.4915e+003 (4.8838e+002)	2.1144e+001 (8.5504e-002)	2.5629e+002 (2.6516e+001)	3.0957e+002 (3.0056e+001)
JADE	1.5570e+001 (5.8873e+000)	6.3398e+003 (2.8974e+001)	2.1130e+001 (5.5539e-002)	1.4377e+002) (9.9918e+000)	2.0083e+002 (2.9938e+001)
jDE	4.0089e+001 (7.0089e+000)	6.1186e+003 (1.0085e+002)	2.1133e+001 (2.1937e-002)	1.8183e+002 (1.5208e+001)	1.1077e+002 (3.9978e+001)
SaDE	**1.1198e+001 (9.7539e+000)**	6.0958e+003 (5.9976e-009)	2.1132e+001 (3.7782e-002)	1.0944e+002 (1.0975e+001)	9.8864e+001 (1.4575e+001)
DEGL	1.4072e+001 (1.0928e+001)	6.1094e+003 (6.8840e-011)	2.1131e+001 (1.6609e-002)	1.5439e+002 (1.9862e+001)	1.5619e+002 (2.4409e+001)
ADE_CM	2.2437e+001 (1.2226e+001)	**5.2295e-009 (6.3359e-009)**	**2.0648e+001 (6.2600e-002)**	**4.9748e+001 (7.9907e+000)**	**7.3627e+001 (1.1667e+001)**

Better on 8 functions, Bitter on 2 functions, Equality on 0 functions

Table 2. Mean and Standard Deviation of the Error Values for f_{11}–f_{25}(50D). The best entries are marked in boldface.

Func / Algorithm	f_{11} Mean (Std)	f_{12} Mean (Std)	f_{13} Mean (Std)	f_{14} Mean (Std)	f_{15} Mean (Std)
DE/rand/1/bin	7.3651e+001 (2.1073e+000)	2.0497e+006 (6.0276e+005)	4.0647e+001 (2.4622e+000)	2.3849e+001 (1.4982e-001)	4.9167e+002 (7.9545e+001)
DE/target-to-best/1/bin	5.0677e+001 (3.4437e+000)	2.6642e+006 (2.0975e+005)	3.8947e+001 (9.1973e+000)	2.285e+001 (3.9728e-001)	5.0944e+002 (7.0649e+001)
JADE	6.4072e+001 (2.0957e+000)	1.3847e+005 (9.2289e+003)	2.6579e+001 (2.9271e+000)	2.2265e+001 (5.0349e-001)	3.7549e+002 (3.0097e+001)
jDE	7.3315e+001 (1.2415e+000)	1.0277e+005 (1.2364e+004)	2.8627e+001 (5.0629e+000)	2.2296e+001 (5.0674e-001)	4.0002e+002 (2.0977e-008)
SaDE	6.6672e+001 (1.1694e+000)	**1.0085e+04 (2.0865e+03)**	3.0975e+001 (6.4087e+000)	2.2498e+001 (4.0329e-001)	3.9014e+002 (5.0497e+001)
DEGL	6.2097e+001 (1.2657e+000)	6.0389e+004 (3.4977e+004)	3.29746e+001 (3.0975e+000)	2.2267e+001 (1.0659e-001)	3.8875e+002 (3.0164e+001)
ADE_CM	**4.9014e+001 (6.8108e+000)**	1.0643e+006 (3.8026e+005)	**4.7727e+000 (8.2710e-001)**	**2.1897e+001 (3.7870e-001)**	**2.7719e+002 (3.6051e+001)**
Func / Algorithm	f_{16} Mean (Std)	f_{17} Mean (Std)	f_{18} Mean (Std)	f_{19} Mean (Std)	f_{20} Mean (Std)
DE/rand/1/bin	2.8743e+002 (1.0097e+001)	3.7948e+002 (1.0974e+002)	9.9157e+002 (5.8197e+001)	9.4700e+002 (5.6195e+001)	9.8713e+002 (5.4943e+001)
DE/target-to-best/1/bin	2.6719e+002 (2.0972e+001)	2.6719e+002 (4.09754e+001)	9.4974e+002 (4.0977e+001)	9.3072e+002 (2.7315e+001)	9.4061e+002 (6.0443e+001)
JADE	1.6458e+002 (1.9461e+001)	1.7911e+002 (2.0613e+001)	9.2027e+002 (3.7948e+000)	9.5591e+002 (4.9253e+001)	9.9034e+002 (6.0977e+001)
jDE	2.6487e+002 (9.0067e+000)	3.1974e+002 (3.7919e+001)	9.1861e+002 (2.3716e+000)	9.1627e+002 (1.4329e+001)	9.9816e+002 (1.9134e+001)
SaDE	1.4215e+002 (2.0974e+001)	1.8974e+002 (4.5619e+000)	9.0452e+002 (4.2919e+001)	9.3195e+002 (1.7492e+001)	9.3143e+002 (5.7254e+001)
DEGL	1.4038e+002 (1.9716e+001)	1.7309e+002 (5.9785e+000)	9.5673e+002 (1.6715e+001)	9.1575e+002 (3.0247e+001)	9.2196e+002 (4.6285e+001)
ADE_CM	**4.7949e+001 (2.0325e+000)**	**8.1306e+001 (6.4336e+001)**	**8.3667e+002 (5.7629e-003)**	**8.3713e+002 (1.453e-001)**	**8.3589e+002 (3.659e-002)**
Func / Algorithm	f_{21} Mean (Std)	f_{22} Mean (Std)	f_{23} Mean (Std)	f_{24} Mean (Std)	f_{25} Mean (Std)
DE/rand/1/bin	9.1108e+002 (6.4409e+002)	1.0039e+003 (7.7729e+001)	9.2184e+002 (3.1037e+002)	7.9327e+002 (1.9818e+001)	1.7618e+003 (7.1628e+000)
DE/target-to-best/1/bin	8.9937e+002 (4.9917e+002)	9.8815e+002 (6.3309e+001)	9.3487e+002 (2.9915e+002)	7.4841e+002 (1.7518e+001)	1.7834e+003 (8.9917e+000)
JADE	8.6619e+002 (4.7710e+002)	9.1347e+002 (1.0023e+001)	8.1067e+002 (1.9038e+002)	**2.0000e+002 (0.0000e+000)**	1.6728e+003 (5.9917e+000)
jDE	8.5584e+002 (1.5539e+002)	9.3416e+002 (1.9329e+001)	8.2969e+002 (1.3094e+002)	**2.0000e+002 (0.0000e+000)**	1.5927e+003 (3.9948e+000)
SaDE	8.4419e+002 (2.7747e+002)	9.5626e+002 (1.3548e+001)	8.7715e+002 (1.3382e+002)	**2.0000e+002 (0.0000e+000)**	1.7387e+003 (3.0185e+000)
DEGL	8.3612e+002 (1.0082e+002)	9.2857e+002 (1.2917e+001)	8.3491e+002 (1.5073e+002)	7.2276e+002 (8.619e+001)	1.6194e+003 (4.8749e+000)
ADE_CM	**7.2413e+002 (9.6580e-001)**	**5.0007e+002 (3.2000e-003)**	**7.2826e+002 (1.3640e-001)**	2.1583e+002 (6.3602e-002)	**2.1425e+002 (1.6429e+000)**

Better on 13 functions, Bitter on 2 functions, Equality on 0 functions

The error values achieved by ADE_CM is compared with several well-known DE variants including two Classical DE variants and four state-of-the-art ADE variants such as SaDE[6], JADE[7], jDE[8], DEGL[9] on 50*D* problems as shown in Table 1 and 2. Their parametric setups have been chosen according to the guidelines from their respective literature. We also simulate our algorithm in 100 dimensions and provide comparative results of all the algorithms in Table 3. We exclude the results of last eleven hybrid composite functions as they have excessive computational time in 100 dimensions. The population size for any DE variants has been kept at 100 irrespective of the dimension *D*. For each function, each of these algorithms is run 50 runs. The function error values are listed at maximum FES which was set to 5×10^5 for 50 dimensions and 1×10^6 for 100 dimensions.

All algorithms are implemented in MATLAB. Simulations have been done on an Intel core-2-duo PC with 3-GB RAM and 2.1 GHz speed.

Table 3. Mean and Standard Deviation of the Error Values for f_1–f_{14}(100D). The best entries are marked in boldface.

Algo / Func	DE/rand/1/bin	DE/target-to-best/1/bin	JADE	jDE	SaDE	DEGL	ADE_CM
f_1	3.0726e05 (4.097e-06)	8.4465e-06 (9.422e-08)	6.3185e-10 (4.425e-09)	8.483e-07 (4.985e-09)	8.3945e-08 (5.387e-09)	9.6844e-07 (4.094e-08)	**1.1937e-13 (1.798e-14)**
f_2	8.946e+04 (6.197e+4)	9.0218e+03 (7.955e-05)	3.3923e+03 (8.488e-06)	6.3871e+03 (7.493e-07)	8.0371e+04 (8.535e-03)	8.926e+04 (8.456e-06)	**8.49e-01 (6.109e-01)**
f_3	9.742e+07 (7.941e+5)	9.9016e+06 (4.485e+3)	**2.7371e+06 (6.473e+4)**	9.0632e+06 (4.893e+4)	7.9171e+06 (7.998e+2)	4.8326e+07 (5.095e+4)	6.271e+06 (2.772e+5)
f_4	2.9744e+05 (4.926e+3)	7.7854e+04 (8.198e+3)	4.935e+04 (7.977e-03)	7.9261e+04 (2.746e+3)	6.043e+04 (7.557e+3)	1.937e+05 (4.946e+4)	**3.4455e+03 (2.057e+2)**
f_5	1.045e+06 (4.782e+3)	5.099e+05 (6.038e+3)	7.5251e+05 (3.946e+3)	3.0641e+05 (7.435e+3)	8.9364e+05 (2.191e+3)	9.094e+05 (6.938e+3)	**7.451e+03 (7.133e+2)**
f_6	1.7584e+05 (4.039e+1)	9.4029e+04 (7.562e+1)	7.533e+04 (7.373e+1)	8.5734e+04 (7.365e+0)	2.1972e+04 (7.491e-1)	4.2684e+04 (5.835e+1)	**1.538e+02 (5.543e+1)**
f_7	1.8467e+05 (3.815e+2)	1.6781e+05 (8.844e+2)	9.0417e+04 (8.474e+2)	1.0325e+05 (6.594e+2)	9.8736e+04 (7.459e+2)	1.2383e+05 (9.457e+2)	**9.9000e-03 (1.18e-3)**
f_8	2.2645e+01 (3.2295e-1)	2.2297e+01 (8.1945e-1)	2.1964e+01 (9.467e-1)	2.1697e+01 (8.033e-1)	2.2695e+01 (9.079e-1)	2.221e+01 (4.825e-1)	**2.0943e+01 (4.290e-2)**
f_9	9.3384e+02 (4.648e+1)	9.4137e+02 (7.783e+1)	8.9272e+02 (4.504e+1)	9.4064e+02 (7.405e+1)	8.4337e+02 (4.135e+1)	9.5591e+03 (3.927e+1)	**1.8128e+02 (7.878e+0)**
f_{10}	9.7649e+02 (5.010e+1)	7.8905e+02 (6.452e+1)	5.857e+02 (8.876e+1)	6.9046e+02 (7.859e+1)	8.3596e+02 (1.656e+1)	8.2017e+02 (3.559e+1)	**2.4038e+02 (1.300e+1)**
f_{11}	9.2248e+01 (2.096e+0)	8.9057e+01 (6.194e+0)	9.2214e001 (4.872e+0)	**7.3325e+01 (9.445e+0)**	8.3915e+01 (4.091e+0)	8.1089e+01 (1.443e+1)	1.4533e+02 (4.265e-1)
f_{12}	9.0046e+06 (8.094e+5)	6.2285e+05 (4.113e+5)	6.768e+05 (2.106e+4)	7.4547e+05 (6.562e+5)	**9.0778e+04 (5.893e+3)**	8.7781e+05 (7.762e+4)	4.0002e+06 (5.226e+5)
f_{13}	5.1309e+01 (4.605e+0)	4.4482e+01 (3.015e+0)	3.3414e+00 (4.108e+0)	3.8890e+01 (3.583e+1)	3.5583e+00 (5.109e+0)	3.9914e+01 (2.082e+1)	**1.4151e+01 (1.686e+0)**
f_{14}	4.4315e+01 (1.6457e-1)	4.5126e+01 (4.1943e-1)	4.6611e+01 (3.9078e-1)	4.3309e+01 (2.7712e-1)	4.2891e+01 (2.9320e-1)	**4.2276e+01 (7.0286e-1)**	4.3444e+01 (5.713e-1)
Better on 10 functions, Bitter on 4 functions, Equality on 0 functions							

4 Conclusion

From the Experimental results we conclude that ADE_CM algorithm has defeated all the six algorithms in 21 numbers of functions of 50*D* where as in 100*D* this number is 10 among the first 14 functions. Therefore, it can be concluded that this algorithm has the ability to optimize functions in high-dimensional search space. The idea to compare the result in 100*D* fitness problem is to evaluate the search performance of ADE_CM with scaling of search space. Thus, this algorithm has the high probability of being used in large scale optimization problems in future research.

References

1. Storn, R., Price, K.V.: Differential Evolution—A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces, ICSI, Berkeley, CA, Tech. Rep. TR-95-012, http://http.icsi.berkeley.edu/~storn/litera.html
2. Gamperle, R., Muller, S.D., Koumoutsakos, P.: A parameter study for differential evolution. In: Proc. Advances Intell. Syst., Fuzzy Syst., Evol. Comput., Crete, Greece, pp. 293–298 (2002)
3. Zhang, J., Sanderson, A.C.: An approximate Gaussian Model of Differential Evolution with Spherical Fitness Functions. In: Proc. IEEE Congr. Evol. Comput., Singapore, pp. 2220–2228 (2007)
4. Ronkkonen, J., Kukkonen, S., Price, K.V.: Real Parameter Optimization with Differential Evolution. In: Proc. IEEE CEC, vol. 1, pp. 506–513 (2005)
5. Mezura-Montes, E., Vlázquez-Reyes, J., Coello Coello, C.A.: A Comparative Study of Differential Evolution Variants for Global Optimization. In: Proc. GECCO, pp. 485–492 (2006)
6. Qin, A.K., Huang, V.L., Suganthan, P.N.: Differential Evolution Algorithm with Strategy Adaptation for Global Numerical Optimization. IEEE Trans. Evol. Comput. 13(2), 398–417 (2009)
7. Zhang, J., Sanderson, A.C.: JADE: Adaptive Differential Evolution with Optional External Archive. IEEE Trans. Evol. Comput. 13(5), 945–958 (2009)
8. Brest, J., Greiner, S., Bošković, B., Mernik, M., Žumer, V.: Self-adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems. IEEE Trans. Evol. Comput. 10(6), 646–657 (2006)
9. Das, S., Abraham, A., Chakraborty, U.K., Konar, A.: Differential Evolution Using A Neighborhood Based Mutation Operator. IEEE Trans. Evol. Comput. 13(3), 526–553 (2009)
10. Whitely, L.D.: The GENITOR Algorithm and Selection Pressure: Why Rank-Based Allocation of Reproductive Trials is Best. In: Third International Conference on Genetic Algorithms, San Mateo, CA (1989)
11. Suganthan, P.N., Hansen, N., Liang, J.J., Deb, K., Chen, Y.P., Auger, A., Tiwari, S.: Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization. Nanyang Technol. Univ., Singapore (2005)
12. Brest, J., Greiner, V., Bošković, B., Mernik, M., Žumer, V.: Self-adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems. IEEE Trans. on Evol. Comput. 10(6), 646–657 (2006)

Modified Particle Swarm Optimization with Switching Update Strategy

Rupam Kundu[1], Rohan Mukherjee[1], and Swagatam Das[2]

[1] Department of Electronics and Telecommunication Engineering, Jadavpur University
[2] Electronics and Communications Sciences Unit, Indian Statistical Institute,
Kolkata – 700 108, India
{rupam2422,rohan.mukherjii}@gmail.com, swagatam.das@isical.ac.in

Abstract. This article aims at improving the Particle Swarm Optimization, by uniquely reshaping its update strategy for generating new solutions with a switching strategy that transits between exploration and convergence, a time-varying inertia weight to control particles' movement and an aging mechanism to avoid stagnation in local basins of attraction. The algorithm addressed as MPSO-SUS has been compared with eight other state-of-artEAs on a standard benchmark of sixteen functions. The results of such comparison indicate that MPSO-SUS clearly and statistically outperform the other well-known approaches, justifying its distinctive feature which makes it a successful optimizer.

1 Introduction

Particle Swarm Optimization originally introduced for simulating social behaviour and swarm intelligence, that involves betterment of a particle's position with a linear dependency of knowledge on neighboring and personal best, is basically a metaheuristic which does not guarantee an optimal solution is ever found. Modifications on PSO to amplify its efficiency have been a real challenge since its formulation. MPSO-SUS is also a modification on the basic framework of PSO, where the particles' velocity have been updated with a linear dependency of knowledge either on neighboring or on personal best thereby switching between the two update strategies with equal probability. An aging scheme avoids unwanted stagnation and enhances search efficiency. Moreover the formula of inertia weight proposed in this paper is very effective in controlling the particles' step size. The results thus obtained have been shown to be remarkably better those obtained with most of the state-of-the-art algorithms which successfully assert our claim that the added features improve the search-ability and exploitative nature of the PSO framework.

2 Particle Swarm Optimization

PSO or Particle Swarm Optimizer [1], as the name suggests, uses a swarm of particles as coordinates in a D-dimensional landscape. Each particle is characterised by its

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 644–652, 2012.

position vector $\vec{x}_i = \{x_i^1, x_i^2, ..., x_i^D\}$,basically a co-ordinate in the D-dimensional frame, and its velocity vector $\vec{v}_i = \{v_i^1, v_i^2, ..., v_i^D\}$. The velocity and position of any particle in the Particle Swarm is updated based on the following:

$$v_i^d \leftarrow v_i^d + c_1 * rand1_i^d * \left(p_{best,i}{}^d - x_i^d \right) + c_2 * rand2_i^d * \left(g_{best,i}{}^d - x_i^d \right) \quad (1)$$

$$x_i^d \leftarrow x_i^d + v_i^d \quad (2)$$

where x_i^d and v_i^d respectively represent the *d*-th component of the position and velocity of the *i*-th particle. The best position of the particle i.e. the position at which the *i*-th particle yields its best fitness value is termed $\vec{p}_{best,i}$ where $\vec{p}_{best,i} = \{ p^1{}_{best,i}, p^2{}_{best,i}, ..., p^D{}_{best,i} \}$. Using similar terminology the global best position of the *i*-th particle is given by $\vec{g}_{best,i} = \{ g^1{}_{best,i}, g^2{}_{best,i}, ..., g^D{}_{best,i} \}$.The terms c_1 and c_2 are basically constants which varies the attraction of the particle towards the particle best and global best positions, while $rand1_i^d$ and $rand2_i^d$ are random numbers bound within the range [0,1]. A particle's velocity per dimension is always constrained to lie within a limited value denoted by v_{max}^d.

3 MPSO-SUS

In this section we discuss in details the key features of MPSO-SUS on a sectional basis:

A. Switching Update Strategy

The velocity update for a particular individual involves a linear dependence with the difference with its personal best $\vec{p}_{best,i}$ and a difference between its global best $\vec{g}_{best,i}$. Both the terms individually play significant role in the optimization procedure. If the velocity update is accomplished only by the term involving the difference with $\vec{p}_{best,i}$ the particle will continuously betters itself with no attraction towards the global optima. This causes the population to remain diverged throughout the search procedure with little or no scope for convergence. On the other hand if the update scheme involves only the difference with $\vec{g}_{best,i}$ the result will be an early convergence with little or no scope for exploitation. Early convergence may be helpful for unimodal functions. But problems which includes complex multimodal shifted or rotated functions, early convergence can cause the population to get trapped in some local basins of attraction. So a proper blend of exploration and exploitation in the equivalent ratio ensures a balanced searching phenomenon which is the desired situation needed for tackling single objective problems.

Thus the sole idea behind formulating MPSO-SUS may be the inspiration to device an optimizer where the particles' new position will be governed by the probabilistic switching between two learning strategies. Therefore, the velocity update equation is modified as:

$$if \ (rand<0.5) \quad v_i^d \leftarrow \omega_i * v_i^d + c_1 * rand1_i^d * \left(p_{best,i}{}^d - x_i^d \right);$$

$$else \quad v_i^d \leftarrow \omega_i * v_i^d + c_2 * rand2_i^d * \left(g_{best,i}^d - x_i^d \right); \ end$$

where, $d = 1:D$ and $i = 1:no\_of\_particles$, c_1 *and* $c_2 = acceleration\ parameters$ (3)

B. Aging Scheme

If any particle gets trapped in some local optimum during the searching procedure there will be minimum scope for its further optimization due to its loss of exploration ability. So in that case an Aging scheme has been employed that keeps record of every particle's performance and takes action if case of stagnation. When such particle is detected the strategy is to allow a certain period to that particle for its improvement which is being noted by the change of its $\vec{p}_{best,i}$. If its $\vec{p}_{best,i}$ does not change for λ consecutive iterations, the particle is re-initialized thereby reducing unnecessary function evaluations. After the particle gets re-initialized the counter variable keeping the record for the change of its $\vec{p}_{best,i}$ is reset to 0.

The aging scheme includes the following steps:

1. For every member of a population it is noted whether the $\vec{p}_{best,i}$ of that corresponding particle is getting updated or not.
2. If the $\vec{p}_{best,i}$ of a particle is not getting changed for a particular iteration a counter variable named $stagnate_i$ initially set to 0 is increased by unity. If the $\vec{p}_{best,i}$ of the particle changes at any iteration within the period $stagnate_i$ is declared to be 0 again.
3. If the value of $stagnate_i$ exceeds the particle is re-initialized.
4. Accordingly its fitness, and $\vec{p}_{best,i}$ are updated.

C. Pseudo-code

```
1. Initialize the population (number of subpopulations
   * no. of particles.
2. Initialize initial velocity associated with each
   particle within pre-defined range.
   velocity range= 0.2* range of particles.
3. Calculate fitness values of each particle.
4. Based on these values find local subpopulation best
   i.e. g_best,i .
5. Find the best position seen by each particle i.e.
   p_best,i .
6. While FES < Total_FES
   a. Evaluate constants ω, c1 and c2 as
                ω_i = 0.9 − (0.5) * (1 − (1/iter_i))   c1 = 2   c2 = 2
   b. Update velocity of each particle by the follow-
      ing equation
      if (rand<0.5)   v_i^d ← ω_i * v_i^d + c1 * rand1_i^d * (p_best,i^d − x_i^d);
      else   v_i^d ← ω_i * v_i^d + c2 * rand2_i^d * (g_best,i^d − x_i^d); end ;
```

c. If v_i^d exceeds its boundary values restrict it within its boundary.
d. Update a particles position by
$$x_{i,j} = x_{i-1,j-1} + v_{i,j}$$
e. Restrict particles within boundary.
f. Update fitness of each particle.
g. Update $\vec{p}_{best,i}$ of each particle if i^{th} position is better than $(i-1)^{th}$
Else increase $stagnate_i$ by 1;
If $stagnate_i$ increases λ the position is reinitialised.
f. Update $\vec{p}_{best,i}$ as before if new position is better.
j. Update $\vec{g}_{best,i}$ if i^{th} subpopulation is better suited to the landscape.

7. End while.

4 Parameters

In the following table we tabulate the value of inertia weight (ω), acceleration parameters (c_1 and c_2), maximum velocity (V_{MAXd}) and other parameters considered by MPSO-SUS and by other algorithms with which it has been compared with.

Table 1. Comparison table for parameters used by different algorithms and MPSO-SUS

Name	ω	c_1	c_2	V_{MAXd}	OTHERS
FIPS	Nil	Nil	Nil	0.5*range	χ=0.729; $\sum c_i$= 4.1
HPSO-TVAC	0.9-0.4	2.5-0.5	0.5-2.5	0.5*range	Nil
DMS-PSO	0.9-0.2	2.0	2.0	0.2*range	m=3; R=5
CLPSO	0.9-0.4	Nil	Nil	0.2*range	c=1.49445; m=7
OPSO	0.9-0.4	Nil	Nil	0.5*range	Nil
OLPSO	0.9-0.4	Nil	Nil	0.2*range	c=2
MPSO-SUS	0.9-0.5*(1- (1/*iter*))	2	2	0.2*range	λ =10

- **Aging Parameter λ**

The aging parameter is nothing but the number of iteration the Aging mechanism allows a particle to improve its fitness. Proper value of the aging parameter is an absolute necessity simply because if the value is low, it means we have allowed the particle minimal scope for improvement. If, on the other hand, the value is high the result is unnecessary exploitation and sometimes even chances of getting trapped in local optima. A value of ten suits our purpose exceedingly well.

- **Time varying Inertia Weight**

The parameter ω is needed to control the impact of previous velocity on the current velocity. In the beginning of the search large inertia weight is required for global explorations as significant moves are required but later small inertia weight is required

for local explorations. So it is evident that if ω is constant exploration will be poor as there will be no sharp demarcation between exploration and exploitation as demanded by optimization. So the parameter ω is considered time varying to get rid of the above problem. In our algorithm the value of inertia weight is gradually decreased from 0.9 to 0.4 as given in Eqn. (4).

$$\omega_i = 0.9 - 0.5*(1-(1/iter_i)) \tag{4}$$

where "*iter*" denotes the number of iterations,

With an increase in number of iterations, the expression $(1-(1/iter)))$ starting from the value 0 for the first iteration slowly approaches unity.

$$0 < (1-(1/iter)) < 1 \tag{5}$$

5 Test Functions

MPSO-SUS has been tested on 16 standard benchmark functions f_1 to f_{16}, among which $f_1 - f_4$ are unimodal, $f_5 - f_{10}$ are multimodal and $f_{11} - f_{16}$ are shifted and rotated problems. In Table 2 global optimal solution (column V), global optimal value f_{min} (column VI), biased initializations (column IV) with search range(column III) are used according to the definitions in [2]. The significance of defining Accept (column VII) lies in if a solution obtained by an algorithm falls between the acceptable value and f_{min} the run is considered successful.

6 Results and Discussions

All the runs have been performed on MATLAB R2009b on a machine with Windows® 7 64-bit configuration, 4GB RAM, 500GB hard disk and 2.26 GHz Intel® Core™ i3-350M Processor.

Table 3 shows the comparison between MPSO-SUS and other PSO variant algorithms namely HPSO [8], FIPS [7], GPSO [5], LPSO [4], OPSO [6], CLPSO [2], DMSPSO [3], and OLPSO [9] with respect to the mean solution obtained and standard deviation values. All the runs were performed using a single swarm of 40 particles which is essentially a global best seeking approach and function evaluations assigned for each function is kept fixed at 2, 00,000. Each function is tested iteratively and mean solution of 25 independent trials is reported.

Analysis of Table 3 reveals that MPSO-SUS outperforms all the other state-of-arts in 14 out of 16 cases. In f_7 and f_{13} OLPSO-L outperforms MPSO-SUS by a significant margin. But the proposed approach performs exceedingly well in case of f_{11} and f_{12} where all other approaches fail.

Convergence Graphs: The convergence graphs attained by MPSO-SUS when tested on functions $f_1, f_2, f_4, f_9, f_{10}, f_{14}$ are also shown below. For all the graphs, Best Objective value in log scale vs. Function Evaluations is plotted.

Table 2. Sixteen functions used to test

Type	Test Function	Search Range	Initialization Range	Global Opt.	f_{min}	Accept	Name
Unimodal	$f_1=\sum_{i=1}^{D} x_i^2$	$[-100,100]^D$	$[-100,50]^D$	$[0]^D$	0	$1x10^{-6}$	Sphere
	$f_2=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	$[-10,10]^D$	$[-10,5]^D$	$[0]^D$	0	$1x10^{-6}$	Schwefel'sP2.22
	$f_3=\sum_{i=1}^{D-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	$[-10,10]^D$	$[-10,10]^D$	$[1]^D$	0	100	Rosenbrock
	$f_4=\sum_{i=1}^{D} ix_i^4+rand[0,1)$	$[-1.28,1.28]^D$	$[-1.28,0.64]^D$	$[0]^D$	0	0.01	Noise
Multi-modal	$f_5=418.9829\times D-\sum_{i=1}^{D} x_i\sin(\sqrt{\lvert x_i\rvert})$	$[-500,500]^D$	$[-500,500]^D$	$[420.96]^D$	0	2000	Schwefel
	$f_6=\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$	$[-5.12,5.12]^D$	$[-5.12,5.12]^D$	$[0]^D$	0	100	Rastrigin
	$f_7=-20exp(-0.2\sqrt{1/D\sum_{i=1}^{D} x_i^2})$ $-exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i))+20+e$	$[-32,32]^D$	$[-32,16]^D$	$[0]^D$	0	$1x10^{-6}$	Ackley
	$f_8=\sum_{i=1}^{D} x_i^2/4000-\prod_{i=1}^{D} cos(x_i/\sqrt{i})+1$	$[-600,600]^D$	$[-600,200]^D$	$[0]^D$	0	$1x10^{-6}$	Griewank
	$f_9=\pi/D\{10\sin^2(\pi y_1)+\sum_{i=1}^{D-1}(x_i-1)^2[1+10\sin^2(\pi y_{i+1})]+$ $(y_D-1)^2\}+\sum_{i=1}^{D} u(x_i,10,100,4)$	$[-50,50]^D$	$[-50,25]^D$	$[0]^D$	0	$1x10^{-6}$	Generalized
	$f_{10}=0.1\{\sin^2(3\pi x_1)+\sum_{i=1}^{D-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})$ $+(x_D-1)[1+\sin^2(2\pi x_D)]\}+\sum_{i=1}^{D} u(x_i,5,100,4)$	$[-50,50]^D$	$[-50,25]^D$	$[0]^D$	0	$1x10^{-6}$	Penalized
Rotated and Shifted	$f_{11}=418.9829\times D-\sum_{i=1}^{D} z_i$ where $z_i=\begin{cases} y_i\sin(\sqrt{\lvert y_i\rvert}), if\ \lvert y_i\rvert\le 500 \\ 0.001(\lvert y_i\rvert-500)^2, if\ \lvert y_i\rvert>500 \end{cases}$ $y_i=y_i'+420.96$	$[-500,500]^D$	$[-500,500]^D$	$[420.96]^D$	0	2000	Rotated Schwefel
	$f_{12}=\sum_{i=1}^{D}(y_i^2-10\cos(2\pi y_i)+10), \vec{y}=\vec{x}*M$ M is an orthogonal matrix	$[-5.12,5.12]^D$	$[-5.12,5.12]^D$	$[0]^D$	0	100	Rotated Rastrigin
	$f_{13}=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} y_i^2}\right)$ $-\exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi y_i))$	$[-32,32]^D$	$[-32,16]^D$	$[0]^D$	0	$1x10^{-6}$	Rotated Ackley
	$f_{14}=1/4000\sum_{i=1}^{D} y_i^2-\prod_{i=1}^{D} cos(y/\sqrt{i})+1$ $y=M*x$, M is an orthogonal matrix	$[-600,600]^D$	$[-600,200]^D$	$[0]^D$	0	$1x10^{-6}$	Rotated Griewank
	$f_{15}=\sum_{i=1}^{D}\left[100\left(z_{i+1}-z_i^2\right)^2+\left(z_i-1\right)^2\right]+f\_bias$ where, $z=x-o+1$ $o=[o_1,o_2,.....,o_D]$;the shifted global optimum	$[-100,100]^D$	$[-100,100]^D$	o	390	490	Shifted Rosenbrock
	$f_{16}(x)=\sum_{i=1}^{D}\left[z_i^2-10\cos\left(2\pi z_i\right)+10\right]+f\_bias$ where, $z=x-o$ $o=[o_1,o_2,.....,o_D]$;the shifted global optimum	$[-5,5]^D$	$[-100,100]^D$	o	-330	-230	Shifted Rastrigin

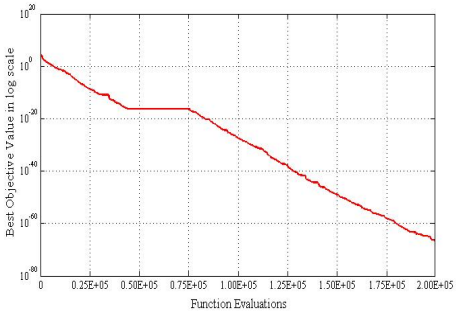


Fig. (a). Sample convergence graph for Function 1

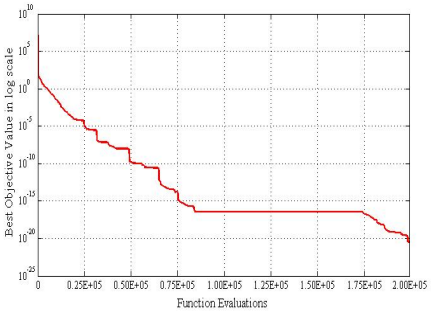


Fig. (b). Sample convergence graph for Function 2

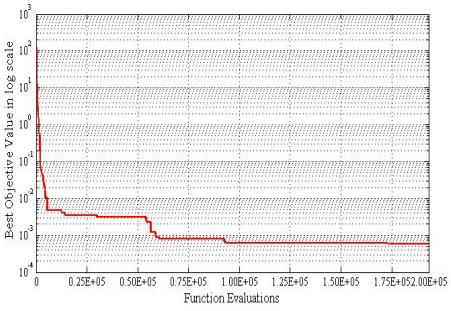


Fig. (c). Sample convergence graph for Function 4

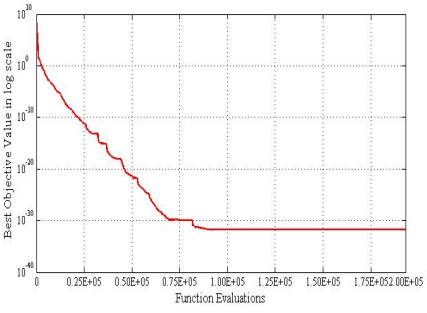


Fig. (d). Sample convergence graph for Function 9

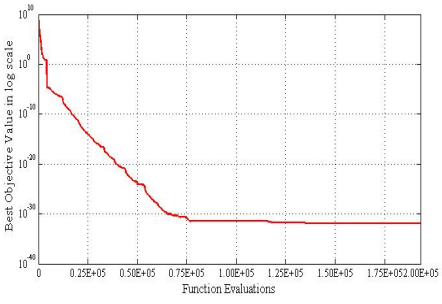


Fig. (e). Sample convergence graph for Function 10

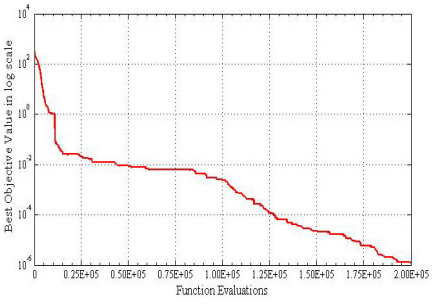


Fig. (f). Sample convergence graph for Function 14

Table 3. Mean and Standard Deviation Values Obtained by MPSO-SUS compared to those obtained through the other standard PSO-Variant algorithms

Functions		FIPS	HPSO- TVAC	DMS-PSO	CLPSO	OPSO	OLPSO-G	OLPSO-L	MPSO-SUS
f_1	Average	2.84E-13	7.13E-33	3.16E-31	4.74E-12	7.47E-18	7.19E-54	1.89E-38	**2.42E-61**
	STD	3.26E-13	2.55E-33	7.24E-31	6.73E-13	4.05E-18	9.13E-54	8.31E-58	**4.18E-61**
f_2	Average	9.98E-08	9.23E-20	3.24E-18	3.76E-08	1.50E-10	9.96E-30	8.85E-22	**3.38E-34**
	STD	5.46E-09	4.02E-20	1.11E-18	6.09E-09	7.78E-11	1.71E-29	8.00E-22	**1.80E-34**
f_3	Average	27.09	26.41	44.58	12.01	49.66	24.53	1.98	**2.11E-05**
	STD	0.34	29.99	34.25	9.81	29.92	35.34	2.55	**3.62E-05**
f_4	Average	5.87E-03	9.97E-02	4.09E-02	5.90E-03	6.09E-02	3.98E-02	4.94E-02	**0.00062**
	STD	5.94E-03	1.09E02	1.03E-03	7.79E-03	1.29E-03	8.03E-03	6.56E-03	**0.00068**
f_5	Average	9.94E02	4.98E03	1.67E03	**3.82E-04**	3.43E03	8.86E02	9.56E-04	**0.00038**
	STD	5.76E02	9.89E02	6.98E02	**2.00E-05**	8.10E02	2.19E02	3.03E-05	**0**
f_6	Average	70.32	9.44	31.52	9.53E-05	7.89	7.23	**0**	**0**
	STD	19.22	3.93	6.32	5.98E-04	2.00	1.03	**5.03E-05**	**0**
f_7	Average	4.28E-07	9.44E-14	9.38E-14	3.66E-07	7.66E-09	8.34E-15	**7.54E-15**	2.53E-14
	STD	9.11E-08	3.01E-14	5.99E-15	7.55E-08	9.52E-09	2.83E-15	**0**	3.64E-15
f_8	Average	9.33E-12	9.62E-03	7.92E-03	9.12E-09	2.37E-03	5.83E-03	**0**	**0**
	STD	5.55E-11	1.54E-03	1.09E-03	3.45E-09	7.08E-03	9.63E-03	**1.30E-06**	**0**
f_9	Average	8.11E-15	3.44E-29	3.09E-30	7.72E-14	4.09E-19	1.96E-32	1.68E-32	**1.57E-32**
	STD	8.39E-15	8.84E-29	1.34E-29	8.92E-14	7.96E-19	9.11E-33	8.99E-33	**0**
f_{10}	Average	3.42E-14	3.04E-28	4.94E-03	3.15E-12	4.87E-18	9.39E-04	1.86E-32	**1.35E-32**
	STD	8.31E-14	4.21E-28	4.99E-03	1.98E-13	2.43E-18	6.20E-03	9.29E-48	**0**
f_{11}	Average	4.58E03	5.88E03	4.46E03	4.52E03	4.67E03	4.18E03	3.87E03	**0.000221**
	STD	1.78E02	7.77E02	6.09E02	8.18E02	2.39E03	6.21E02	1.99E03	**6.19E-05**
f_{12}	Average	1.53E02	57.90	44.91	89.04	69.61	52.32	55.93	**0**
	STD	14.91	13.99	7.98	15.06	17.73	13.56	19.11	**0**
f_{13}	Average	7.12E-07	9.87	4.29E-14	8.13E-05	1.69E-08	8.56E-15	**4.35E-15**	1.58E-14
	STD	3.22E-07	1.87	3.33E-14	3.23E-05	5.42E-09	9.50E-15	**8.19E-16**	1.80E-15
f_{14}	Average	1.35E-08	9.66E-03	1.14E-02	8.12E-05	1.64E-03	1.99E-03	6.06E-08	**2.99e-09**
	STD	3.55E-08	5.77E-03	2.73E-02	4.76E-05	4.99E-03	7.10E-03	2.44E-07	**5.17e-10**
f_{15}	Average	429.93	511.13	529.17	407.17	2.63E07	429.08	422.18	**400**
	STD	22.29	94.39	97.64	42.94	7.13E07	31.14	24.12	**10.65473**
f_{16}	Average	-239.89	-312.15	-301.91	**-330**	-277.11	-325.99	**-330**	**-330**
	STD	25.98	6.22	8.13	**2.91E-05**	11.12	0.98	**2.76E-14**	**0**

7 Conclusion

In this paper we have presented a unique Switching Update Strategy which has not only enhanced the explorative capacity of the basic PSO algorithm but also added enough convergence capacity, which is evident from its significant performance in unimodal as well as multimodal functions. The variation of inertia weight has controlled greatly the particles' movement for global explorations in the beginning and local explorations in the end. The possibility of getting trapped in local optima is completely removed by the introduction of a very effective aging mechanism.

Altogether, MPSO-SUS with its unique features stands out to be a strong optimizer in terms of both fast and assured convergence.

References

1. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proc. IEEE Int. Conf. Neural Netw., vol. 4, pp. 1942–1948 (1995)
2. Liang, J.J., Qin, A.K., Suganthan, P.N., Baskar, S.: Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans. Evol. Comput. 10(3), 281–295 (2006)
3. Liang, J.J., Suganthan, P.N.: Dynamic multi-swarm particle swarm optimizer. In: Proc. Swarm Intell. Symp., pp. 124–129 (June 2005)
4. Kennedy, J., Mendes, R.: Population structure and particle swarm performance. In: Proc. IEEE Congr. Evol. Comput., pp. 1671–1676 (2002)
5. Shi, Y.H., Eberhart, R.C.: A modified particle swarm optimizer. In: Proc. IEEE World Congr. Comput. Intell., pp. 69–73 (1998)
6. Ho, S.-Y., Lin, H.-S., Liauh, W.-H., Ho, S.-J.: OPSO: Orthogonal particle swarm optimization and its application to task assignment problems. IEEE Trans. Syst., Man, Cybern. A 38(2), 288–298 (2008)
7. Mendes, R., Kennedy, J., Neves, J.: The fully informed particle swarm: Simpler, maybe better. IEEE Trans. Evol. Comput. 8(3), 204–210 (2004)
8. Ratnaweera, A., Halgamuge, S., Watson, H.: Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans. Evol. Comput. 8(3), 240–255 (2004)
9. Zhan, Z.-H., Zhang, J., Li, Y., Shi, Y.-H.: Orthogonal learning particle swarm optimization. IEEE Trans. Evol. Comput. 15(6) (December 2011)

Statistical Analysis Based Adjustment Method for Convergence Rate of Spiral Optimization

Kenichi Tamura and Keiichiro Yasuda

Tokyo Metropolitan University, 1-1 Minamiosawa, Hachioji, Tokyo 192-0397, Japan

Abstract. Recently, the authors proposed a new metaheuristics method for continuous optimization problems based on analogy of spiral phenomena in nature which is called Spiral Optimization. The Spiral Optimization has two setting parameters: the convergence rate r and the rotation angle θ. However, any adjustment methods or policies for them has not been studied so far. This paper especially focuses on the convergence rate r and develops its adjustment method through statistically analyzing its properties. The effectiveness is verified from simulation results under some conditions.

1 Introduction

Recently, the authors proposed a new metaheuristics method for continuous optimization problems based on analogy of spiral phenomena in nature which is called Spiral Optimization [1]. The focused spiral phenomena are approximated to logarithmic spirals which frequently appear in nature, such as nautilus shells, low pressure area, whirling currents, and so on.

We pointed out that the behavior of the spirals has possibility to contribute the global optimization and made the spiral model which discretely generates logarithmic spirals. Using the spiral model, we developed the spiral search model and algorithm. The effectives of Spiral Optimization was confirmed from the simulation results for many benchmark problems in comparison with some methods on Particle Swarm Optimization.

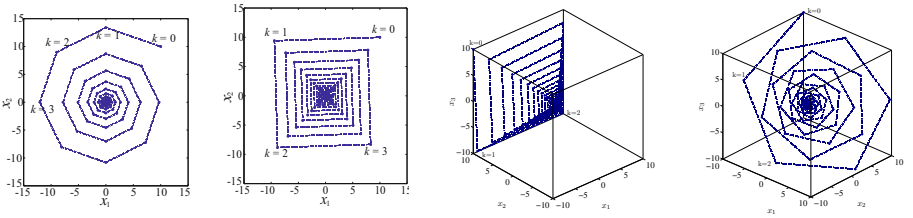
The Spiral Optimization has two setting parameters: the convergence rate r and the rotation angle θ. However, any tuning methods or policies for them have not been studied yet. This paper especially focuses on the convergence rate r and develops its adjustment method through statistically analyzing its properties. The effectiveness is verified from simulation results under some conditions.

2 Spiral Optimization[1]

The spiral model whose state $\boldsymbol{x}(k) \in \mathbb{R}^n$ at time k converges to the arbitrary center $\boldsymbol{x}^\star$ while drawing a spiral trajectory from the arbitrary initial state $\boldsymbol{x}(0) \in \mathbb{R}^n$ as k increases is formulated as follows:

$$\boldsymbol{x}(k+1) = S(r,\theta)\boldsymbol{x}(k) - (S(r,\theta) - I)\boldsymbol{x}^\star \tag{1}$$

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 653–661, 2012.



(a) $\theta = \pi/4, r = 0.95$ (b) $\theta = \pi/2, r = 0.95$ (c) $\theta = \pi/2, r = 0.95$ (d) $\theta = \pi/4, r = 0.95$

Fig. 1. Examples of Spiral Trajectories of (11)

where $S(r, \theta) = rR(\theta)$, the parameter $0 \leq \theta < 2\pi$ is the rotation angle around the center $\boldsymbol{x}^\star$ per k, the parameter $0 < r < 1$ is the convergence rate of distance between a point and the center $\boldsymbol{x}^\star$ per k, $I \in \mathbb{R}^{n \times n}$ is the identity matrix, and $R(\cdot) \in \mathbb{R}^{n \times n}$ is a n-dimensional rotation matrix defined as composition of basic rotation matrices $R_{i,j}(\cdot)$ as follows:

$$R(\theta) = \prod_{i=1}^{n-1} \left(\prod_{j=1}^{i} R_{n-i,n+1-j}(\theta) \right) \tag{2}$$

whose basic rotation matrices $R_{i,j}(\cdot)$, $i = 1, \cdots, n-1$, $j = 2, \cdots, n$ are composed of the following (k, l) elements:

$$R_{i,j}^{[k,l]}(\theta) = \begin{cases} \cos\theta \ (k = l = i \text{ or } k = l = j) \\ 1 \ (k = l \neq i \neq j) \\ -\sin\theta \ (k = i \text{ and } l = j) \\ \sin\theta \ (k = j \text{ and } l = i) \\ 0 \ (\text{otherwise}) \end{cases}, \quad k = 1, 2 \cdots, n, \quad l = 1, 2 \cdots, n. \tag{3}$$

Figure 1 shows the examples of trajectories of (11) with $\boldsymbol{x}^\star = \boldsymbol{0}$ in 2-dimensional and 3-dimensional space, respectively.

As the examples suggests, the behavior of the spirals are congenial to a well known effective strategy in metaheurisitics that is "diversification in the early phase and intensification in the latter phase during a search" [1].

The spiral search model is a multipoint search model which uses m of the spiral model (11) and has the interaction $\boldsymbol{x}^\star$ among them, which can efficiently and naturally realize the strategy from diversification to intensification, and its algorithm using the search model is shown below.

Algorithm of Spiral Optimization

Step 0: [Preparation]. Set the number of search points $m \geq 2$, the parameters $0 \leq \theta < 2\pi$, $0 < r < 1$, and the maximum iteration number $k_{\max}$.

Step 1: [Initialization]. Set the initial points $\boldsymbol{x}_i(0) \in \mathbb{R}^n, i = 1, 2, \cdots, m$ in the feasibility region at random and the center $\boldsymbol{x}^\star = \boldsymbol{x}_{i_g}(0)$, $i_g = \arg\min_i f(\boldsymbol{x}_i(0))$, $i = 1, 2, \cdots, m$. Set $k = 0$.

Step 2: [Updating $\boldsymbol{x}_i$]. $\boldsymbol{x}_i(k+1) = S(r,\theta)\boldsymbol{x}_i(k) - (S(r,\theta) - I)\boldsymbol{x}^\star$, $i = 1, 2, \cdots, m$.

Step 3: [Updating $\boldsymbol{x}^\star$]. $\boldsymbol{x}^\star = \boldsymbol{x}_{i_g}(k+1)$, $i_g = \arg\min_i f(\boldsymbol{x}_i(k+1))$, $i = 1, 2, \cdots, m$.

Step 4: [Checking Termination Criterion]. If $k = k_{\max}$ then terminate. Otherwise, set $k := k + 1$, and return to Step 2.

These search points which approach the common center $\boldsymbol{x}^\star$ always set as the best solution during a search while drawing spiral trajectories are finding better solutions.

The effectives of the algorithm was confirmed from the simulation results for many benchmark problems in comparison with some methods on Particle Swarm Optimization [1].

3 Analysis of Convergence Rate

The Spiral Optimization has two parameters that users must set: the convergence rate r and the rotation angle θ. Our previous works [1] tuned them by trial and error to get good search performance for each problem and condition. However, any parameter adjustment methods or policies for them have not studied so far.

To make the Spiral Optimization more practical and useful, it is important to develop their effective parameter adjustment methods for various problems and conditions without trial and error.

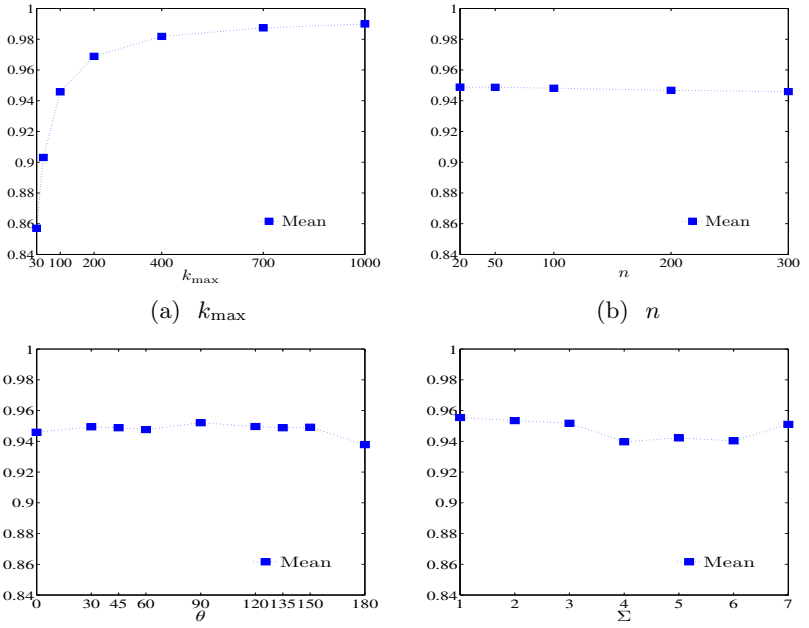
In this paper, we especially focus on the convergence rate r in the two parameters. This section analyzes properties of the convergence rate r using much simulation data under various problems and conditions in which the number of search points is fixed as $m = 20$. Namely, we examine how the convergence rate r value effects search results under various computational situations.

To analyze properties of the convergence rate r statistically, we examine many simulation data under various computational situations. The situations are composed of all combinations of the conditions shown in Table 1: 9 kinds of the rotation angle θ, 7 kinds of the maximum iteration number $k_{\max}$, 5 kinds of the dimension n and 7 kids of the representative benchmark functions Σ: 1. Parabolic, 2. Schwefel, 3. 2^nminima, 4. Rastrigin, 5. Alpine, 6. Levy, 7. Ackely. (See [2] for these definition which are omitted as the page limit here.)

The simulation at each combination is conducted by changing r from 0.800 to 0.999 by 0.001 under 100 kinds of initial conditions given in $-5 \le x_i(0) \le 5$, $i = 1, 2, \cdots, n$. Therefore, the simulation result per each r at every combination is calculated as the average of 100 results under the 100 different initial conditions.

Table 1. Condition Values Used for Analysis

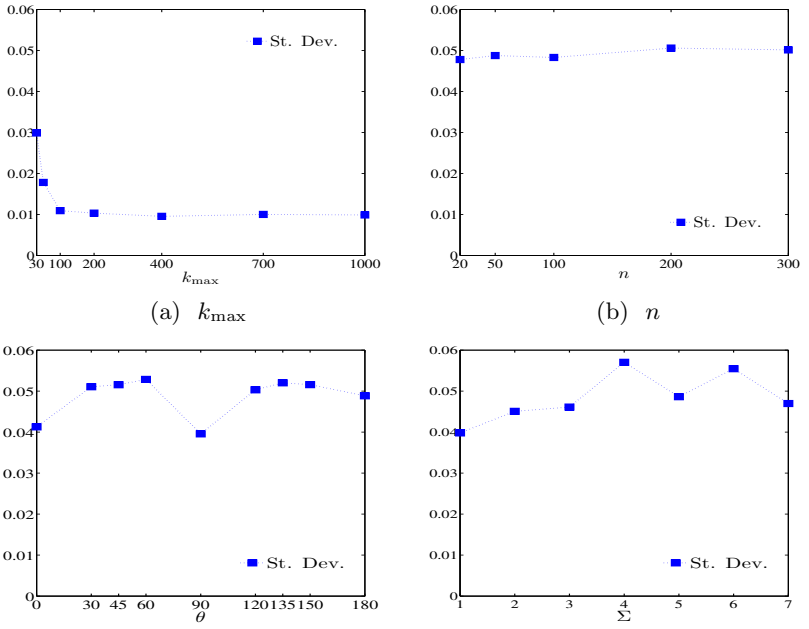
Conditions	Values
θ	0, 30, 45, 60, 90, 120, 135, 150, 180
$k_{\max}$	30, 50, 100, 200, 400, 700, 1000
n	20, 50, 100, 200, 300
Σ	1, 2, 3, 4, 5, 6, 7



(a) $k_{\max}$ (b) n

(c) θ (d) Σ

Fig. 2. Mean of the Best r Values at Each Condition



(a) $k_{\max}$ (b) n

(c) θ (d) Σ

Fig. 3. Std. Dev. of the Best r Values at Each Condition

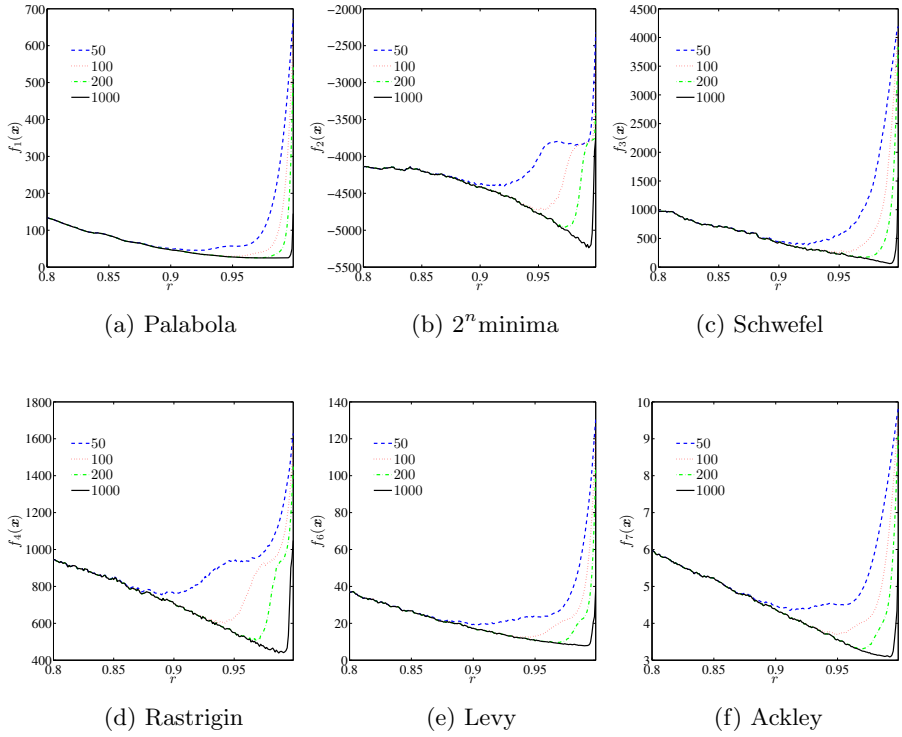


Fig. 4. Relation between r Value and Evaluation Value at Each $k_{\max}$ at Each Problem ($n = 100, \theta = 90$)

Finding the best r value in the range $0.800 - 0.999$ at every combination, Fig. 2 shows the mean of the best r values categorized according to the same value of each condition from all the best r values. Namely, (a) shows the mean of the best r values at every $k_{\max}$, in the same way, (b), (c) and (d) show the mean of the best r values at every n, θ and Σ, respectively. Fig. 3 shows similarly the standard deviation of the best r values at the same value of each condition.

These results reveal the following property of the convergence rate r:

[I] The best r values have strong correlation with $k_{\max}$ that they approach 1 as $k_{\max}$ increases.

Furthermore, Fig. 4 shows the fitness value $f(\boldsymbol{x})$ on $r = 0.800 - 0.999$ at each $k_{\max}$ in 4 kinds for each problem in case of $n = 100$ and $\theta = \pi/2$. These results elucidate not only the property [I] but also the following property:

[II] Good r values exist around the best r value at each case.

In addition, we observed similar results to Fig. 4 in case of other combination with n and θ.

Table 2. Data for Design of Adjustment Method

i	1	2	3	4	5	6	7
$k_{\max}$	30	50	100	200	400	700	1000
r	0.8570	0.9031	0.9458	0.9689	0.9818	0.9874	0.9899

4 Proposed Adjustment Method for Convergence Rate

At the former section, we have analyzed the properties of convergence rate r as [I] and [II] from various simulation data. The properties suggest it is valid to adjust the convergence rate r on the basis of $k_{\max}$. That is to set a convergence rate value according to the curve through the square data points in Fig.2 (a). This principle has high possibility to stably give a good r value on average at each computational condition although it cannot guarantee to give the best r value.

In this study, we try to formulate this curve in Fig.2 (a) as an approximate function of $k_{\max}$ which makes easier to obtain a value of the convergence rate r according to the curve. This approach is also expected to be the good adjustment method even if $k_{\max} > 1000$ that is out of data.

In order to select basic functions for formulating such approximate function, we focus on the following features of the curve of Fig.2 (a):

1. The mean of the best r approaches 1 as $k_{\max}$ increases.
2. It changes more slowly $k_{\max}$ as $k_{\max}$ increases.

As a function satisfying the above features, we adopt the Gompertz curve[3], which is often used for models of growth curves, formulated as follows:

$$\zeta = f_g(\xi; a, b, c) = ab^{\exp(-c\xi)} \tag{4}$$

which draws the curves in Fig.5 and has the following features when $0 < b < 1, c > 0$: $\zeta \to a$ as $\xi \to \infty$, $\zeta = ab$ at $\xi = 0$, $\zeta \to 0$ as $\xi \to -\infty$.

To get a function fitting the data of Fig.2 (a) shown in Table 2, we formulate the following function composed of three Gompertz curves with $a = 1$:

$$r = f_r(k_{\max}; \boldsymbol{b}, \boldsymbol{c}) = (b_1^{-c_1 k_{\max}} + b_2^{-c_2 k_{\max}} + b_3^{-c_3 k_{\max}})/3. \tag{5}$$

The reason why three Gompertz curves are selected is to increase the freedom of parameters for the fitting.

To derive a curve of (5) through the data of Table 2, we solve the following optimization problem.

$$\min_{\boldsymbol{b},\boldsymbol{c}} \sum_{i=1}^{7} (f_r(k_{\max_i}; \boldsymbol{b}, \boldsymbol{c}) - r_i)^2$$

where $k_{\max_i}$ and r_i respectively mean $k_{\max}$ and r at i-column in Table 2. Consequently, the values of $b_i, c_i,\ i = 1, 2, 3$ in (5) have been derived as follows.

$$b_1 = 0.46,\ b_2 = 0.94,\ b_3 = 0.82, c_1 = 0.037,\ c_2 = 0.0007,\ c_3 = 0.008 \tag{6}$$

Table 3. Results of Numenical Experiments ($m = 20$)

				$\theta = 45$		$\theta = 90$		$\theta = 120$	
Problem	n	$k_{\max}$		Best r	Proposed r	Best r	Proposed r	Best r	Proposed r
Schwefel	50	50	r	0.914	0.901	0.909	0.901	0.924	0.901
			f_2	365	376	107	122	391	408
		200	r	0.979	0.969	0.970	0.969	0.979	0.969
			f_2	224	247	36.3	38.1	220	240
		700	r	0.993	0.988	0.990	0.987	0.993	0.987
			f_2	82.5	132	16.8	17.6	58.3	104
	200	50	r	0.904	0.901	0.922	0.901	0.923	0.901
			f_2	6242	6272	1745	1993	5915	6045
		200	r	0.974	0.969	0.968	0.969	0.966	0.969
			f_2	4484	4513	674	695	4921	4968
		700	r	0.991	0.987	0.990	0.987	0.992	0.987
			f_2	3530	3565	337	395	3803	4117
2^nminima	50	50	r	0.913	0.901	0.917	0.901	0.925	0.901
			f_3	-2521	-2440	-2566	-2512	-2534	-2470
		200	r	0.976	0.969	0.971	0.969	0.975	0.969
			f_3	-3077	-3011	-2933	-2912	-3167	-3121
		700	r	0.993	0.987	0.989	0.987	0.992	0.987
			f_3	-3303	-3262	-3071	-3050	-3328	-3300
	200	50	r	0.906	0.901	0.900	0.901	0.908	0.901
			f_3	-6000	-5944	-7929	-7909	-6078	-6058
		200	r	0.974	0.969	0.973	0.969	0.973	0.969
			f_3	-9238	-9119	-8663	-8641	-9299	-9119
		700	r	0.992	0.987	0.988	0.987	0.992	0.987
			f_3	-11654	-11064	-8842	-8819	-11788	-11216
Levy	50	50	r	0.866	0.901	0.897	0.901	0.894	0.901
			f_6	49.3	51.8	13.3	13.9	48.6	49.4
		200	r	0.966	0.969	0.962	0.969	0.964	0.894
			f_6	32.3	34.5	6.11	6.5	30.9	32.8
		700	r	0.987	0.987	0.986	0.987	0.987	0.987
			f_6	25.6	25.6	5.17	5.36	24.8	24.8
	200	50	r	0.885	0.901	0.903	0.901	0.878	0.901
			f_6	98.1	100	25.1	25.4	97.1	98.5
		200	r	0.965	0.969	0.968	0.969	0.965	0.969
			f_6	62.5	65.0	13.7	13.9	63.2	64.6
		700	r	0.989	0.987	0.989	0.987	0.989	0.987
			f_6	40.3	40.9	12.1	12.2	40.0	40.7

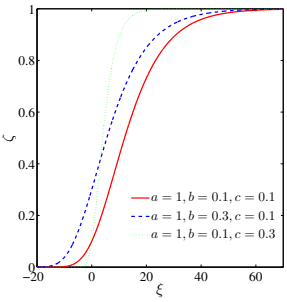


Fig. 5. Gompertz Curve

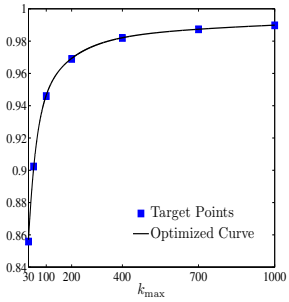


Fig. 6. Curve of Optimized Function (5)

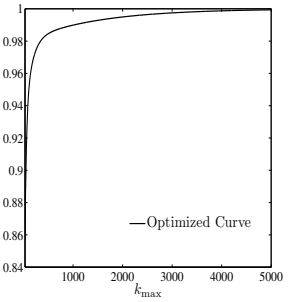


Fig. 7. Curve of Optimized Function (5) until $k_{\max} = 5000$

Using these parameter values, we can observe that the curve of (5) becomes Fig. 6 and fits for the targeted data perfectly. Moreover, from Fig. 7 that shows the curve of (5) until $k_{\max} = 5000$, we can observe it approaches 1 and expect the proposed adjustment method (5) and (6) can be effective over data range.

5 Performance Verification

This section verifies effectiveness of the developed adjustment method (5) and (6) for the convergence rate r by comparing the best convergence rate r in $\Omega = \{0.800, 0.801, \cdots, 0.999\}$ at each computational situation.

The results are shown in Table 3. The "Best r" column shows the best r values in Ω and their search results. The "Proposed r" column shows the r values given by the developed adjustment method and their search results.

From these results, we can observe that the proposed law can give a r value close to the best r value in Ω at each computational situation. At the same time, we also can confirm the search performance is also close to the best one.

6 Conclusion

This paper has statistically analyzed properties of the convergence rate r of the Spiral Optimization and the developed simple adjustment method for r based on the analysis results for the first time. The future task includes analysis on the rotation angle θ and development of its adjustment method.

Acknowledgment. This work was supported by foundation for the Fusion Of Science and Technology (FOST).

References

[1] Tamura, K., Yasuda, K.: Spiral dynamics inspired optimization. Journal of Advanced Computational Intelligence and Intelligent Informatics 15(8), 1116–1122 (2011)
[2] Aiyoshi, E., Yasuda, K. (eds.): Metaheuristics and Their Applications. Ohmsha (2007)
[3] Winsor, C.P.: The gompertz curve as a growth curve. Proc. National Academy of Sciences 15(1) (1932)

Neglect Benevolence in Human-Swarm Interaction with Communication Latency*

Phillip Walker[1], Steven Nunnally[1], Mike Lewis[1], Andreas Kolling[2], Nilanjan Chakraborty[2], and Katia Sycara[2]

[1] University of Pittsburgh, Pittsburgh, PA 15213, USA
[2] Carnegie Mellon University, Pittsburgh, PA 15213, USA

Abstract. In practical applications of robot swarms with bio-inspired behaviors, a human operator will need to exert control over the swarm to fulfill the mission objectives. In many operational settings, human operators are remotely located and the communication environment is harsh. Hence, there exists some latency in information (or control command) transfer between the human and the swarm. In this paper, we conduct experiments of human-swarm interaction to investigate the effects of communication latency on the performance of a human-swarm system in a swarm foraging task. We develop and investigate the concept of *neglect benevolence*, where a human operator allows the swarm to evolve on its own and stabilize before giving new commands. Our experimental results indicate that operators exploited neglect benevolence in different ways to develop successful strategies in the foraging task. Furthermore, we show experimentally that the use of a predictive display can help mitigate the adverse effects of communication latency.

Keywords: Human-swarm interaction, swarm robotics, human-robot interaction.

1 Introduction

Swarm robotic systems are composed of simple individual units and generate collective behavior that is robust to failure of individual robots [1, 2]. However, for practical use of such systems in large and complex human-supervised missions, key problems that arise in human-swarm interaction need to be understood and solved. In application scenarios, the human operator may be remotely located and there may be communication constraints due to the hardware limitations of the robots (e.g., communication radios of limited power) and the environmental properties (e.g. underwater environments). This will lead to delay in the communication of information between the swarm and the human. The delay in communication results in the human neither knowing perfectly the current state of the swarm nor the effect of her action on the robots. The extant literature

* This research has been sponsored in part by AFOSR FA955008-10356 and ONR Grant N0001409-10680.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 662–669, 2012.

on Human-Swarm Interaction (HSI) [3–10] has not studied the performance and behavior of human operators in the presence of delayed information transmission between the swarm and the human and vice versa. Therefore, in this paper, we create an experimental scenario to study the effects of latency and error on human performance in controlling swarm robotic systems. We also study the use of predictive displays to mitigate the effect of latency.

In our experimental foraging scenario, a human operator guides a swarm to find unknown targets in a given area. The robots have a single behavior, namely flocking, and the operator applies inputs (a) to give a desired direction of flocking to the robots and (b) to enforce cohesiveness among the robots (by activating constraints for attracting neighbors that are far away and repelling neighbors that are very close). In our experiment, each subject performs the mission under three conditions, namely, (a) without any latency (control condition), (b) with equal latency in the human to swarm and swarm to human communication channel (c) the same latency as (b) but with a predictive display. In all conditions, each robot has some error in transforming the orientation heading to its own reference frame (due to localization errors), which is modeled as a Gaussian distribution. Our experimental results indicate that, as expected, there is a degradation of performance due to latency. However, when using the predictive display, the performance of the operators can be as good as it was in the absence of delay (control condition). We also found that the users exhibited different strategies for effectively controlling the swarm.

The human operator needs to influence the swarm without adversely disturbing the swarm (such as breaking it into many small connected components). The effect of an operator command is dependent on swarm state, which gradually evolves to a steady state after a command has been issued. To capture the idea that humans may need to observe the evolution of the swarm state before acting, we investigate a novel concept called *neglect benevolence*, whereby neglecting the swarm to allow for stabilization before issuing new commands may be beneficial to overall mission performance. An analogous but different concept called *neglect tolerance* [11, 12] is used in human robot interaction. For independently operating multiple robots, neglect tolerance is defined as the time a human can neglect a robot without degradation in system performance. For neglect tolerance, it is assumed that the performance of an individual robot degrades with time and hence the attention of the operator needs to be scheduled so that the time between servicing robots is minimized [13, 14]. In contrast, neglect benevolence captures the concept that it may be beneficial to leave the swarm alone for a certain length of time after issuing an instruction to allow the behavior to stabilize (since the swarm state may not degrade monotonically with time). Our results show evidence of neglect benevolence.

2 Task Description

Our study investigates the ability of a human operator to effectively influence a swarm operating under algorithms that require time to exhibit emergent behaviors. In particular, we investigated (1) the effect of communication latency

in human-swarm performance, (2) the effect of predictive displays, and (3) the existence of neglect benevolence as a new notion in HSI. We created a foraging task that requires users to direct a swarm around an open environment using instructions to change swarm heading and flocking constraints. We also use this study to look at the effect of communication latency on this ability.

2.1 The Environment

The overall task of the experiment is to guide the swarm around an open, 100x100 meter environment to find targets of different colors. We use three different environments divided into six regions, with each region containing one of three target frequencies: *low* (0-4 targets), *medium* (5-9 targets), or *high* (10+). The target distribution is different across the search missions that each participant solves, but each environment contains 1 high, 2 low, and 3 medium frequency regions. There are 40 targets in total in each of the three environments, and each participant receives a worksheet indicating the target frequency of each region.

We use Stage v. 3.2.2 [15] to simulate the environment, the targets, and a swarm of 40 differential drive P2AT robots. Robot controllers are implemented using the Robot Operating System (ROS) [16]. Each robot is equipped with a color sensor with a range of 4 meters to detect the colored targets. We also simulate an additional sensor that allows the robots to sense the location of a neighbor within 4 meters. This allows each robot to estimate the direction of motion of its neighbors from repeated observations of their location.

The graphical user interface is also implemented in ROS. This interface displays the known targets and positions of the robots; however, it does not display the region boundaries. During the trial, each robot transmits its position and observations from its color sensor to the user interface. A target is considered found only if six or more robots detect it simultaneously, at which point the target is shown on the map and a counter on the side is incremented.

2.2 Human Influence

Users can influence the swarm with three commands: *stop*, *heading*, and *apply-constraints*. The *heading* command broadcasts a global heading to the swarm. To simulate a localization error, every robot interprets the global heading with respect to a local coordinate frame to compute its goal heading. The orientation of this local coordinate frame differs from the true orientation of the robot by an error sampled from the Gaussian distribution $\mathcal{N}(0, \frac{4\pi}{9})$. Upon receiving the command, the robots turn toward their respective goal heading and begin moving (Fig. 1a). In order to correct for the erroneous interpretations of the global heading, each robot also executes a consensus algorithm at a frequency of 0.5 Hertz. Robots sense the direction of motion of their neighbors and update their goal heading to the average goal heading of their neighbors and themselves (Fig. 1b). Finally, the user can issue an *apply-constraints* command, which applies biologically-inspired flocking constraints similar to those in [1], [2], and [10]. These constraints force robots to repel from each other if they were closer than

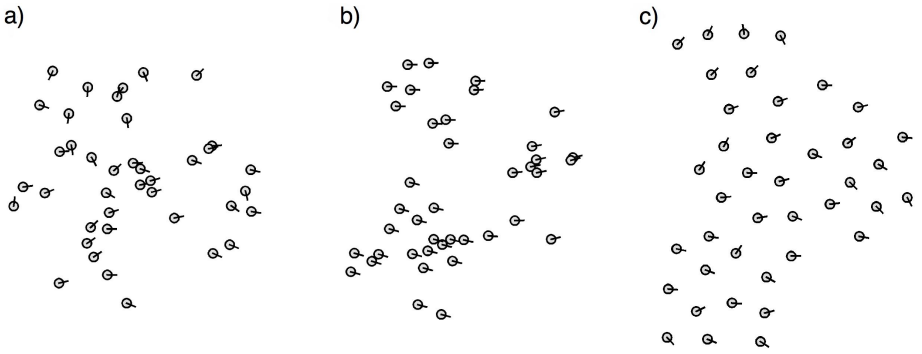


Fig. 1. The swarm in each of the three possible states: when the user first issues a heading command (a), after the consensus algorithm has stabilized (b), and after the user has just issued the constraints command and the robots have adjusted (c)

1.5 meters, and, if none were this close, to attract to neighbors further than 3 meters (fig. 1c). Otherwise, the robots execute the consensus algorithm described previously.

2.3 Experimental Design

The experiment consists of three conditions—the *control*, *latency*, and *predictive* conditions. In all conditions, the operator begins with a swarm of 40 robots positioned randomly in a 10x10 meter box.

In the *control* condition, there is no latency in either of the human-to-swarm or swarm-to-human channels.

In the *latency* condition each channel—operator-to-swarm and swarm-to-operator—has a latency of 10 seconds, meaning that operator-issued commands will not reach the robots until 10 seconds after issuing, and the state of the swarm displayed in the interface for the user is 10 seconds old. Therefore, from the operator's point of view, the swarm will not begin executing the command until 20 seconds after the *heading* instruction is issued, as the message will take 10 seconds to reach the swarm, and the reflection of this command will take 10 more seconds to return to the operator.

In the *predictive* condition, the latency of 10 seconds for each channel remains present; however, the interface gives the user a prediction of where each member of the swarm will be in 20 seconds, or when the next command is received, by taking the heading and speed (which is a constant 0.5 m/s) of each swarm member and extrapolating the robot's position that far in the future. This gives the user a rough estimate of how the entire swarm will look when an issued command actually reaches the swarm.

Each participant has a different environment for each of these conditions, and the order of both the conditions and the maps are randomized for each participant in order to remove any learning biases. 21 participants (8 men and 13 women) were recruited from the greater Pittsburgh area to participate in the

study. Each participant was given a short explanation of the controls and goals of the study and a 10-minute training session to familiarize themselves with the interface, after which they completed each of the three conditions.

3 Results and Discussion

The overall mission performance for each participant is measured in terms of the number of targets found and the coverage of the high-frequency target regions. In the *control* condition participants found 19.86 targets and covered $1.47m^2/s$ of the high-frequency target regions on average, both of which were significantly higher than in the *latency* condition, where participants found 16.71 targets ($p = .021$) and covered $0.98m^2/s$ of the high-frequency regions ($p = .007$). In the *predictive* condition, however, participants found 18.86 targets on average and covered $1.24m^2/s$ of the high-frequency regions, neither of which were significantly different from the *control* condition ($p = .467$ and $p = .196$, respectively). These results show that the latency of 10 seconds significantly impedes performance, but that the predictive display in the *predictive* condition removes this impediment.

An indirect measure of an operator's ability to control the swarm is the swarm's overall connectivity. To determine the overall connectivity of the swarm at any given time, we represented the swarm as a simple graph, G, and used the second eigenvalue of the graph's Laplacian as the connectivity measure. Keeping the swarm connected has two benefits. First, such a swarm is less likely to break off into smaller connected components, which allows the user to meet the six-robot threshold for sensing a target more easily. Secondly, a highly-connected swarm will reach consensus faster, as each robot will have more neighbors to average with during each consensus round.

We see that the *latency* condition had an average connectivity of 0.111, which was significantly higher than in *control* condition, which had an average connectivity of 0.084, $p < .001$. Similarly, average connectivity in the *predictive* condition was also significantly higher than in the *control* condition, with a value of 0.116, $p < .001$, see Fig. 2. This points to the existence of neglect benevolence, as it demonstrates that communication latency helped enforce swarm connectivity by causing operators to wait to see the results of their heading command before deciding whether to issue a new one. As a consequence, each command has a longer duration, thus giving the swarm more time to stabilize after each command. We find this is indeed true, with users issuing significantly more commands on average in the *control* condition ($M = 27.81$) than in the *latency* condition ($M = 17.76$, $p = .028$), and significantly more than in the *predictive* condition ($M = 18.86$, $p = .052$).

To investigate the various behaviors and strategies of the operators, we investigated the duration and timing of the *heading* and *apply-constraints* instructions. We analyzed the average time between a *heading* and a subsequent *apply-constraints* command (hereafter referred to as time to constraints), or the next *heading* command (duration). Because these instructions involve altering

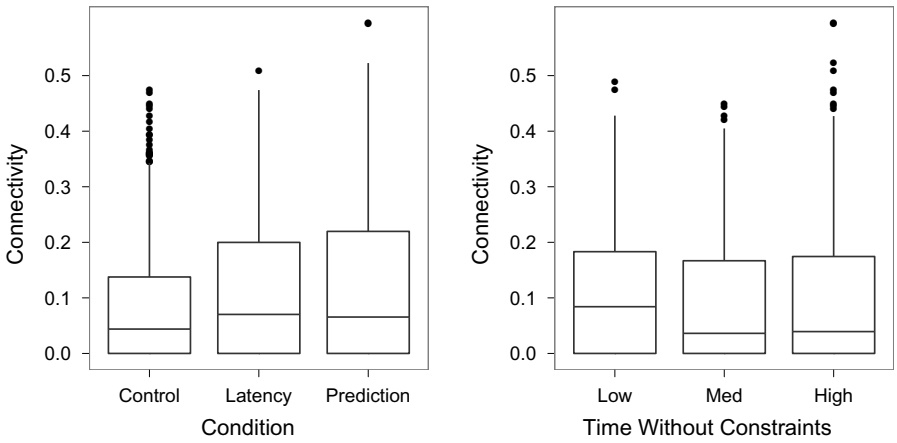


Fig. 2. These graphs display the connectivity of the swarm at the end of each heading command issued. Connectivity is significantly different across conditions (left), with the latency conditions generally having more connected swarms. The average time to constraints also impacts connectivity (right), with participants choosing to enforce constraints more often generally having better connected swarms. The boxes in each figure represent the bottom three quartiles, and the outliers are marked as black dots.

the current state of the swarm at the time they are issued, even two identical commands (i.e., two heading commands with the same heading) can lead to drastically different effects depending on the state of the swarm.

As demonstrated by the previous results, because the number of heading commands were different across condition, the duration of the commands are similarly different. The *apply-constraints* instruction, however, can be more flexible, and operators may decide to issue constraints at any point in time after a *heading* command, or issue a new *heading* command without activating constraints at all. To investigate the effect of the application of constraints, we clustered the data into three equal-sized groups across all missions into a high (100% to 78%), medium (78% to 45%), or low (45% to 0%) group corresponding to the number of heading commands for which constraints were later applied.

Performance in terms of targets found does not differ between these groups, but the total area swept is significantly different, with fewer constraints (low) leading to less overall area covered by at least six sensors ($p = .040$). On the other hand, many constraints (high) lead to a larger error between the heading of the swarm and the operator's goal heading ($p = .011$), and fewer constraints (low) have more heading instructions leading to a consensus (74%, $p < .001$).

These results suggest that operators employ different strategies to find a larger number of targets, with some operators using constraints earlier and more often, increasing coverage at the expense of higher heading errors, while others preferred the opposite.

We expected a difference in heading duration and time to constraints between the two latency conditions and the control condition, as participants must wait 20 seconds to see the results of their commands. Interestingly, however, across all instructions, only 27% in the *latency* condition and 30% in the *predictive* condition have constraints activated later than 20 seconds, neither of which were significantly different from control, meaning that, unlike with the heading commands, operators often issued the constraints prior to seeing the effect of the heading instruction on the swarm. Swarm connectivity was also significantly impacted by the application of constraints as well, with the high constraints group having significantly better connectivity than the medium or low group ($p = .043$), see Fig. 2.

These results provide considerable support for neglect benevolence. Frequent and short commands provide an operator more control, but sacrifice swarm cohesion as reflected in the lower connectivity value and the higher number of connected components. This is largely due to the fact that new heading commands reintroduce error into the swarm members' estimated heading and require several rounds of consensus to stabilize. Activating the constraints too early and often, however, leads to higher heading errors, and thus may make the swarm more difficult for the human to control. We found that operators develop two different strategies around neglect benevolence: either stabilize the consensus and lower the heading error, or maintain swarm cohesion and improve coverage. It appears operators were able to use either method to their advantage and obtain a good performance, and that, while latency can degrade performance overall, it does not impact one strategy more than the other.

4 Conclusions and Future Work

This study provides support for the idea of neglect benevolence, with the commands in the study leading to strategies with different costs and benefits depending on the state of the swarm at the time the commands were issued. Frequent heading commands provided the user more control over the direction and location of the swarm at the expense of total coverage and swarm connectivity. Due to the nature of the swarm algorithms, high position and heading accuracy and high swarm cohesion were not possible simultaneously. Therefore, participants had to decide which characteristics were more important. For the present study, both strategies achieved success; however, other tasks may be better achieved with one or the other. This will be the subject of future study.

Latency had a negative effect on the number of targets found; however, using a predictive display mitigated the negative effects. Latency also seemed to significantly impact the frequency with which an operator issues *heading* commands, but not *apply-constraints* commands. As this is the first study to investigate latency in human-swarm interaction, future work will address latency issues for human control of other tasks and swarm algorithms.

References

1. Reynolds, C.: Flocks, herds and schools: A distributed behavioral model. In: ACM SIGGRAPH Computer Graphics, vol. 21, pp. 25–34. ACM (1987)
2. Couzin, I., Krause, J., James, R., Ruxton, G., Franks, N.: Collective memory and spatial sorting in animal groups. Journal of Theoretical Biology 218(1), 1–11 (2002)
3. Bashyal, S., Venayagamoorthy, G.: Human swarm interaction for radiation source search and localization. In: IEEE Swarm Intelligence Symposium, SIS 2008, pp. 1–8. IEEE (2008)
4. Kolling, A., Nunnally, S., Lewis, M.: Towards human control of robot swarms. In: Proceedings of the 7th International Conference on Human-Robot Interaction. ACM (2012)
5. Coppin, G., Legras, F.: Autonomy spectrum and performance perception issues in swarm supervisory control. Proceedings of the IEEE (99), 590–603 (2012)
6. Cummings, M.: Human supervisory control of swarming networks. In: 2nd Annual Swarming: Autonomous Intelligent Networked Systems Conference (2004)
7. Klarer, P.: Flocking small smart machines: An experiment in cooperative, multi-machine control. Technical report, Sandia National Labs, Albuquerque, NM, United States (1998)
8. Kira, Z., Potter, M.: Exerting human control over decentralized robot swarms. In: 4th International Conference on Autonomous Robots and Agents, ICARA 2009, pp. 566–571. IEEE (2009)
9. Naghsh, A., Gancet, J., Tanoto, A., Roast, C.: Analysis and design of human-robot swarm interaction in firefighting. In: The 17th IEEE International Symposium on Robot and Human Interactive Communication, RO-MAN 2008, pp. 255–260. IEEE (2008)
10. Goodrich, M., Pendleton, B., Sujit, P., Pinto, J.: Toward human interaction with bio-inspired robot teams. In: 2011 IEEE International Conference on Systems, Man, and Cybernetics, SMC, pp. 2859–2864. IEEE (2011)
11. Xu, Y., Dai, T., Sycara, K., Lewis, M.: Service level differentiation in multi-robots control. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2224–2230. IEEE (2010)
12. Dai, T., Sycara, K., Lewis, M.: A game theoretic queueing approach to self-assessment in human-robot interaction systems. In: 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 58–63. IEEE (2011)
13. Mitchell, P., Cummings, M.: Management of multiple dynamic human supervisory control tasks. In: 10th International Command and Control Research and Technology Symposium (2005)
14. Mau, S., Dolan, J.: Scheduling for humans in multirobot supervisory control. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2007, pp. 1637–1643. IEEE (2007)
15. Gerkey, B., Vaughan, R., Howard, A.: The player/stage project: Tools for multi-robot and distributed sensor systems. In: Proceedings of the 11th International Conference on Advanced Robotics, Portugal, pp. 317–323 (2003)
16. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.: Ros: an open-source robot operating system. In: ICRA Workshop on Open Source Software, vol. 3 (2009)

A Comment on Bio-inspired Optimisation via GPU Architecture: The Genetic Algorithm Workload

Paula Prata, Paulo Fazendeiro, Pedro Sequeira, and Chandrashekhar Padole

Instituto de Telecomunicações (IT) Portugal
University of Beira Interior, Department of Informatics, Portugal
pprata@di.ubi.pt, fazendeiro@ubi.pt, morps@live.com.pt,
chandupadole@yahoo.com

Abstract. This paper characterizes a genetic algorithm based on the analysis of the workload of its operators. Different granular parallel implementations of a genetic algorithm in the GPU architecture are compared against the correspondent sequential version. With the help of three benchmark problems, a complete characterization of the relative execution times of the genetic operators, varying the population cardinality and the genotype size, is offered. The best speedups, obtained with large populations, are higher than one thousand times faster than the corresponding sequential version. The assessment of different granularity levels shows that the two-dimensional parallelism supported by the GPU architecture is valuable for the crossover operator.

Keywords: GPGPU, Parallel Genetic Algorithms, OpenCL, data parallelism.

1 Introduction

Evolutionary algorithms (EAs) are adaptive robust methods, biologically inspired, that are widely applicable for search, optimisation, and learning problems [1-4]. Genetic algorithms (GA) are an example of such methods that are commonly used in practical applications. A GA requires a limited amount of knowledge about the problem being solved. Relative evaluation of the candidate solutions is enough and no derivatives of cost functions are required. This can be a definitive advantage when compared with other candidate methods of optimisation, as the derivatives of the involved functional in various real world problems are computationally demanding or don't have a closed form, cf. [5].

These computationally intensive population-based search methods present heavy time requirements, hence reducing their applicability especially in what concerns real time applications. The potential for acceleration of population-based, stochastic function optimisers using Graphic Processing Units (GPUs) has been already verified in a number of recent works (e.g. [6-9] just to mention a few) over a representative set of benchmark problems.

In this work we study the relative workload of the different atomic operations of a GA. The paper presents a complete characterization of the relative execution times in

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 670–678, 2012.

CPU and GPU, varying the population cardinality and the genotype size (number of dimensions of the test functions.) The main focus is on the assessment of the performance gains of a generic GA in GPU instead of studying the performance of a particular GA. The findings of the assessment of the parallelisation potential of different granularity levels (population and population+genotype) altogether with the analysis of data parallelism are also reported in the experimental part.

2 Background

Informally a GA can be conceptualised as an iterative algorithm where the successive epochs mimic in silico, in a search space context, the necessary processes in order to a successfully reproduction of the biological organisms [10]. The evolution of potential solutions over successive generations comprises different phases that succeed each other in continuum until a stopping criterion is met.

Generally speaking, the first phase involves the quantitative evaluation of each chromosome in the population. In the second phase, selection, the chance to mate is given to the potential solutions (to the more fit individuals are concealed more reproductive chances). In this work we used binary tournament selection. In the third phase the genetic material of the selected individuals is exchanged by means of a crossover operator. The result of this reproduction phase is a new offspring population, which replaces (or sometimes competes with) the previous population. Some of these newly born individuals were possibly prone to some mutations. Finally, the new generation replaces the old one. Most GAs use fixed population size so it is necessary to decide which offspring are inserted in the next generation. In this work, after some preliminary tests, we have followed a deterministic elitist approach combining a fitness-based replacement (the 2 parents of higher rank are kept) with and age-based one (the remaining parents are replaced by the offspring).

In [11] the impact of varying the size of the population and the number of dimensions of the problem was already studied for a parallel GA implementation in a GPU. When using global memory speedups up to 55 times were achieved (roulette wheel was used for selection). It should be noticed that the results for constant memory were incorrectly reported whenever the population occupied more than 64KB (the addressable constant memory space) due to an error not detected by the former compiler.

OpenCL is a parallel computing framework that provides a cross-platform (CPU/GPU) programming language [12]. An OpenCL application runs on a platform that consists of a host connected to one or more OpenCL devices. An application consists of a host program that executes on the host, capable of launching functions (kernels) that execute in parallel on OpenCL devices. When a kernel is submitted for execution by the host, an N-dimensional index space (with N=1, 2 or 3) is defined. The same kernel can be executed simultaneously by a high number of threads, each one for a point in the index space. Each thread, viz. each kernel instance, is called a work-item and is identified by its point in the index space. The fitness evaluation is eminently a one-dimensional operation (in the sense that usually each chromosome, taken as a whole, can be evaluated in parallel) whereas the crossover and mutation

operators are, in the general case, 2D operators allowing the parallel treatment of not only each chromosome but also each gene inside it.

The same code can be executed over different data items following a SIMD (Single Instruction Multiple Data) model, thus implementing a data parallel programming model. Additionally, work-items can be organized into work groups that can be executed in a SPMD (Single Program Multiple Data) model. Because the GPU just processes data stored in its memory, the program data must be copied to the global memory of GPU before executing the kernel and results must be copied back to CPU memory. Global memory has very high latency.

3 Experiments and Results

The GeForce GTX 295 used in this work has 240 cores (at 1.24 GHz) and 1GB of global memory. It was programmed with OpenCL version 1.1. The host machine is an Intel Core 2 Quad Q9550 at 2.83 GHz with 4 GB of RAM (o. s. Windows 7, 64 bits.)

The solution quality and obtained speedups of the implemented GA were analysed using Michalewicz, Rosenbrock's and Griewangk's functions [4] three artificial benchmark functions commonly used for GA analysis.

The parameters of the GA were kept constant in the presented experiments. The minimization problems were transformed into maximization through the transformation $f = c/(c + J)$, where J is the objective function to minimize and $c>0$ a proper regularization constant. All the population individuals were subjected to the blend crossover (BLX-alpha) operator chosen due to its suitability to the real-valued chromosome encoding that was used [13]. The probability of mutation was equal to 0.025. Furthermore, to prevent good solutions from disappearing during the evolutionary process it was used an elitist approach maintaining the first and second best solutions.

The population's sizes were varied in a geometric progression from 256 to 262144 with a constant ratio of four and the number of genes was varied from 2 to 16 with a constant ratio of two. Results were obtained as the average of 20 runs with 500 iterations each. Due to space constraints we choose to present only the figures that illustrate the main trends, whenever necessary details are given in the supporting text.

3.1 Sequential and Parallel Workloads

This work starts by characterizing a sequential and a parallel version of the canonical genetic algorithm. This is done calculating the execution time percentages in CPU and in GPU for each main step of both implementations of the algorithm. Table 1 briefly outlines the sequential (Algorithm 1) and the parallel (Algorithm 2) implementations.

In the parallel implementation, four kernels are considered, one for each genetic operation: selection, crossover, mutation and evaluation. In Algorithm 2, these are marked with (k). The calculation of the two best values, marked with (best) and used in the elitist approach is done in CPU since, according to our tests, for the population's sizes studied, it is faster to compute those values in CPU rather than in GPU. Another difference between both implementations is the copy of data from CPU to GPU and vice-versa that just happens in the parallel version.

Table 1. Sequential (alg. 1) and parallel (alg. 2) implementation of the canonical GA

Algorithm 1 Sequential implementation of the canonical GA	**Algorithm 2** Parallel implementation in GPU
1. Pre-processing *1.1 Initialise population* *1.2 Evaluation* *1.3 Compute the two best elements (best)* *2. Repeat until convergence* *2.1 Selection by Tournament* *2.2 Crossover* *2.3 Mutation* *2.4 Evaluation* *2.5 Compute the two best elements (best)*	*1. Pre-processing* *1.1 Initialise and copy population to GPU* *1.2 Evaluation (k)* *1.3 Copy the fitness vector from GPU to CPU* *1.4 Compute the two best elements (best)* *2. Repeat until convergence* *2.1 Copy the two best elements to GPU (indices)* *2.2 Launch kernels* *2.2.1 Selection (k)* *2.2.2 Crossover (k)* *2.2.3 Mutation (k)* *2.2.4 Evaluation (k)* *2.3 Copy the fitness vector from GPU to CPU* *2.4 Compute the two best elements (best)*

Fig. 1 presents the graph with the relative execution time in CPU for the Michalewicz function, considering the largest population studied (262144 individuals) and varying the number of dimensions of the problem for 2, 4, 8 and 16 genes. The relative execution times in GPU are presented in Fig.2 whereas the absolute execution times in CPU and GPU are shown in Table 2, for the same function and populations.

Considering the sequential execution in CPU, from Fig. 1 and Table 2, it can be seen that more than 50% of the time is spent in the evaluation whereas the crossover operator comes in second place, spending about 27% of the time. Three steps of the algorithm, crossover, mutation and evaluation depend on the number of genes. For these operations it can be observed that their absolute execution time grows proportionally to the number of genes. Selection mainly depends on the fitness vector size but the number of genes also has a measurable effect since the selected individuals are copied back to the population's vector. The greater the number of genes, the greater is the time of copy. Finally, computing the best values is done over the fitness vector being independent of the number of genes, i.e. the absolute execution time remains constant whereas the relative time slightly decreases. In the sequential version for CPU, the time spent in computing the best values is less than 0.5% for all the populations studied. The graphs for the other population's sizes in CPU are very similar which means that in CPU the time portion spent in each operation is almost independent from the population's size.

In the first parallel implementation for GPU, uni-dimensional kernels were used. The time to copy the fitness vector from GPU to CPU is now included in the "best" step. As should be expected, the absolute time to compute the best values is constant as this step remains to be done in CPU, and the time to copy the fitness vector is also constant for the same size of population (see Table 2). The relative workload of the "best" step is now an important slice of the overall execution time because the parallel implementation of the genetic operators drastically decreases their execution time. However, when the number of genes increases, the relative workload of the "best" step has a significant reduction: for the largest population, it decreases from 44% (with 2 genes) to 8% (with 16 genes).

From Fig. 2 and Table 2 it can be seen that considering just the genetic operations, selection and evaluation are the most time consuming kernels. As in the sequential version, when the number of genes grows the absolute and relative times of evaluation increase. The same happens now with the selection step. Crossover is in the third place varying between 6% (2 genes) and 18% (16 genes). Finally, mutation is the fastest kernel with a very small increase in absolute time and a small decrease in the relative time when the number of genes is increased. From the absolute time perspective the major gains are obtained in the evaluation, crossover and mutation steps.

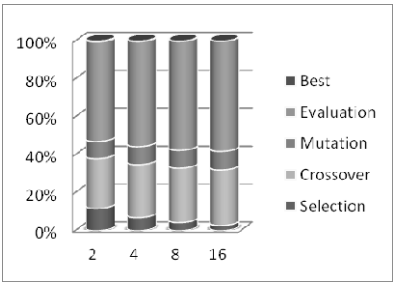


Fig. 1. Execution time percentages for Michalewicz function in CPU (pop. size = 262144) varying the number of genes

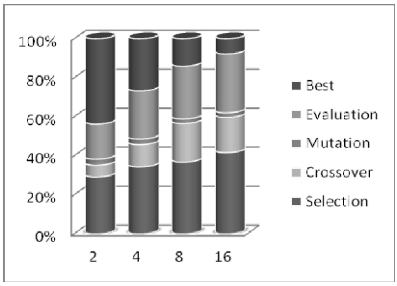


Fig. 2. Execution time percentages for Michalewicz function in GPU (pop. size = 262144) varying the number of genes

Table 2. Single iteration average execution times (ms) by section of code and by the number of genes for the Michalewicz function in CPU and GPU (population's size = 262144)

Sequential execution in CPU (ms)						
Genes	Selection	Crossover	Mutation	Evaluation	Best	Total
2	*44,68*	*100,31*	*33,98*	*200,54*	*1,50*	*381,01*
4	*49,70*	*201,95*	*69,60*	*401,98*	*1,52*	*724,75*
8	*59,58*	*406,31*	*134,58*	*812,76*	*1,51*	*1414,73*
16	*73,72*	*810,95*	*273,20*	*1605,76*	*1,52*	*2765,16*
Parallel execution in GPU (ms)						
2	*1,23*	*0,26*	*0,13*	*0,76*	*1,85*	*4,22*
4	*2,38*	*0,80*	*0,19*	*1,71*	*1,85*	*6,93*
8	*4,81*	*2,60*	*0,31*	*3,54*	*1,84*	*13,10*
16	*9,87*	*4,28*	*0,55*	*7,15*	*1,84*	*23,68*

3.2 Impact in Speedup Values When Varying the Granularity Level

When comparing CPU and GPU times, in all the studied functions, a speedup (CPU time / GPU time) of about one hundred times for the largest population can be observed. For a population of 1024 individuals the speedup is just about twelve times.

To present speedup values that are independent of the host processor a sequential version of the algorithm must be run in the GPU. This version corresponds to the Algorithm 2, where kernels are launched as a unique kernel, using just one work

group with just one work item. Fig. 3a shows the speedups (sequential GPU / parallel GPU) for the three studied functions, Michalewicz (a_1) Griewangk (a_2) and Rosenbrock (a_3), varying the size of the population from 256 to 262144 and varying the number of genes from 2 to 16. It can be observed that, for large populations, speedups greater than one thousand times are achieved. For these populations, when the number of genes increases, i.e. when the computational work of the kernels increases, the main tendency is for the speedup to decrease. For small populations the speedups are much lower (for a population of 256 individuals, the speedups vary from 40 to 108) and the speedups increase as the number of genes grows.

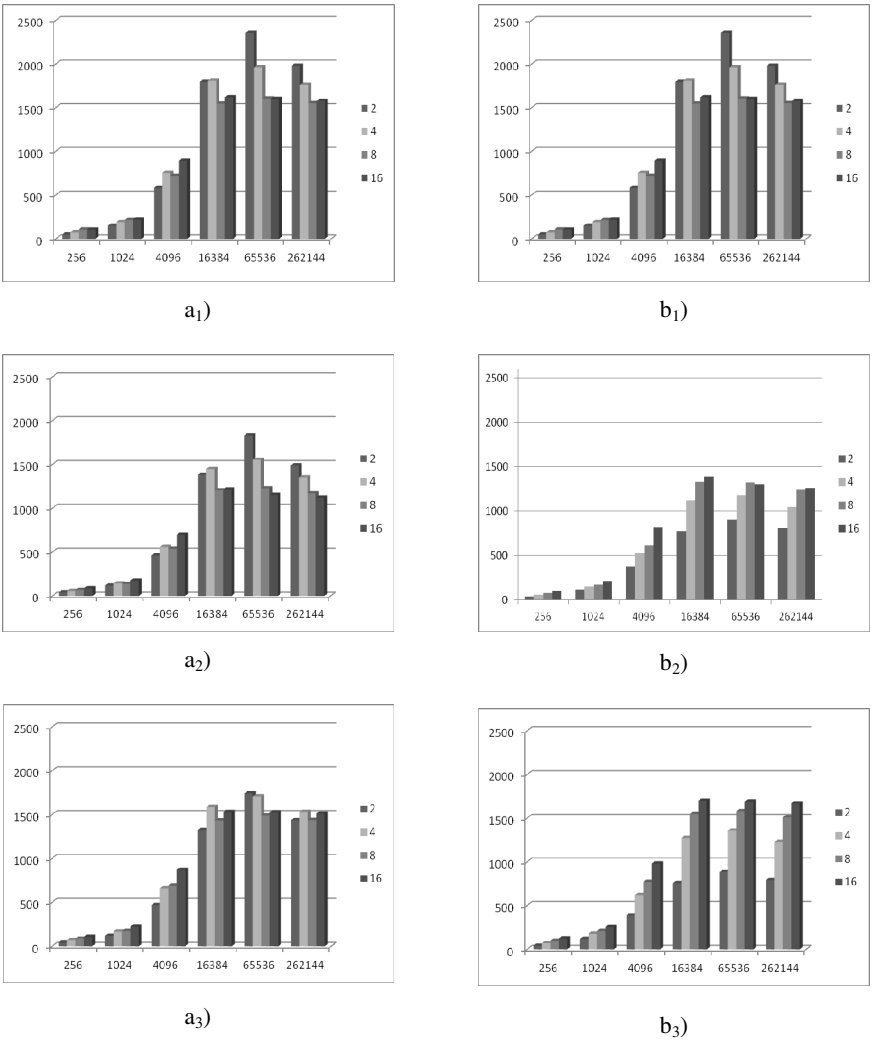


Fig. 3. Speedups (sequential GPU / parallel GPU) for Michalewicz (a_1, b_1), Griewangk (a_2, b_2) and Rosenbrock (a_3, b_3) functions depending on the population's size and the number of genes. Figures (a_i) refer to uni-dimensional crossover kernel and (b_i) to a bi-dimensional one

The next step of this work was trying to improve performance by introducing a new level of parallelism. Since in the crossover and mutation operators the processing is done at the level of each gene, a second dimension can be used in the parallel implementation. By using a bi-dimensional index space for those operators, it is possible to have one work item for each gene of each individual. As for large populations the mutation operator just corresponds to about 2 or 3% of the execution time, the use of a bi-dimensional index space is studied just for the crossover operator. Thus, the kernel for crossover is defined in a way that the number of genes is the first dimension and the population's size is the second dimension. With this version, the absolute execution times obtained for the crossover operator are all the same when the number of genes varies for a fixed population's size. This means that, for this operator the execution time becomes independent from the number of genes. Fig. 3b presents the speedups found with this version, for functions Michalewicz (b_1) Griewangk (b_2) and Rosenbrock (b_3). It can be observed that the speedups for 8 and 16 genes are now greater than in the version with just one dimension. For 2 and 4 genes the bi-dimensional version has lower speedups than the one-dimensional version. This happens for all population's sizes and all the functions. Thus, the two dimensional version is worthier when the threads in the genes dimension have enough work to do. To reinforce this conclusion the speedups using 32 and 64 genes for the Michalewicz function considering populations with 1024 and 262144 individuals are presented in Table 3. For this higher number of genes the two-dimensional version, when opposite to the uni-dimensional one, is advantageous for small populations too.

Table 3. Speedups of one and two dimensional parallel versions for 32 and 64 genes

Michalewicz function				
	sequential GPU / parallel GPU, 1 dim.		sequential GPU / parallel GPU, 2 dim.	
Genes	**1024 individuals**	**262144 individuals**	**1024 individuals**	**262144 individuals**
32	228,26	1469,16	253,92	1728,52
64	231,52	1425,18	260,05	1743,42

3.3 Quality of the Solutions

The quality of the solutions was assessed for the three benchmark functions by performing 20 runs (500 iterations each) of the different GA versions. As illustrated by the values of the average and best fitness in Table 4 (Rosenbrock function) there were no observable statistically significant differences between the quality of the solutions across the sequential and the two previously described parallel implementations. The execution times presented show that the time variability between runs is negligible.

Table 4. Average execution times, fitness average, best fitness and corresponding standard deviations for the Rosenbrock function for a population's size of 262144, varying the number of dimensions (mean values for 20 runs with 500 iterations each)

N. of genes	**Execution time (ms)**	*Standard deviation*	**Fitness Average**	*Standard deviation*	**Best fitness**	*Standard deviation*
Sequential version in GPU						
2	1580043,31	*2,88*	0,855485	*0,000124*	0,999981	*0,000019*
4	3849423,62	*10,54*	0,527343	*0,000474*	0,825193	*0,063785*
8	8380446,55	*22,90*	0,172762	*0,000054*	0,178455	*0,003897*
16	17457554,12	*53,89*	0,069972	*0,000031*	0,072640	*0,000184*
Parallel version in GPU with a uni-dimensional crossover kernel						
2	2023,77	*5,14*	0,855420	*0,000098*	0,999982	*0,000013*
4	3593,58	*4,10*	0,527320	*0,000438*	0,852372	*0,058153*
8	7479,53	*5,36*	0,172728	*0,000043*	0,179335	*0,004173*
16	13307,33	*3,21*	0,069958	*0,000010*	0,072888	*0,000222*
Parallel version in GPU with a bi-dimensional crossover kernel						
2	2846,16	*4,06*	0,855429	*0,000106*	0,999987	*0,000014*
4	3861,88	*4,70*	0,527284	*0,000431*	0,851514	*0,064289*
8	5873,63	*4,15*	0,172722	*0,000037*	0,180928	*0,006966*
16	10115,53	*3,58*	0,069964	*0,000029*	0,072815	*0,000173*

4 Conclusion

It is a well-known fact that the GAs exhibit a well-balanced operation, combining exploration with exploitation. The results reported in the present work shows that the GPU architecture is an affordable alternative in order to efficiently implement population-based search methods. In the case of heavy workloads the speedup gains, regardless of the host machine, are quite impressive. The workload analysis of the parallel versions shows that, as expected, the evaluation is the step of the algorithm with higher computational burden. However operations such as crossover and selection also consume a non negligible portion of processing time. The reported experiments also show that the granularity offered by the GPU architecture is advantageous for the operations presenting functional and data independence at the population+genotype level. These gains increase with the dimensionality of the data.

Regarding our future line of work these findings provide a set of empirical evidences that supports our confidence in adopting this architecture to efficiently implement evolutionary algorithms in order to solve real world problems from the areas of bioinformatics and image analysis.

References

1. Coello, C., Veldhuizen, D.V., Lamont, G.: Evolutionary Algorithms for Solving Multi Objective Problems. Genetic Algorithms and Evolutionary Computation Series, vol. 5. Springer (2002)
2. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. John Wiley & Sons, New York (2001)

3. Goldberg, D.: Genetic Algorithms in search, optimization and machine learning. Addison-Wesley (1989)
4. Bäck, T., Fogel, D., Michalewicz, Z.: Handbook of Evolutionary Computation. Institute of Physics Publishing Ltd., Oxford Univ. Press, Bristol, Oxford (1997)
5. Oliveira, J.V.: Semantic constraints for membership function optimization. IEEE Trans. on Systems, Man, and Cybernetics, Part A: Systems and Man 29(1), 128–138 (1999)
6. Tsutsui, S.: Parallelization of an Evolutionary Algorithm on a Platform with Multi-core Processors. In: Collet, P., Monmarché, N., Legrand, P., Schoenauer, M., Lutton, E. (eds.) EA 2009. LNCS, vol. 5975, pp. 61–73. Springer, Heidelberg (2010)
7. Pospichal, P., Jaros, J., Schwarz, J.: Parallel Genetic Algorithm on the CUDA Architecture. In: Di Chio, C., Cagnoni, S., Cotta, C., Ebner, M., Ekárt, A., Esparcia-Alcazar, A.I., Goh, C.-K., Merelo, J.J., Neri, F., Preuß, M., Togelius, J., Yannakakis, G.N. (eds.) EvoApplicatons 2010, Part I. LNCS, vol. 6024, pp. 442–451. Springer, Heidelberg (2010)
8. de Veronese, L., Krohling, R.: Differential evolution algorithm on the GPU with C-CUDA. In: EEE Congress on Evolutionary Computation, CEC 2010, pp. 1–7 (2010)
9. Lőrentz, I., Andonie, R., Malița, M.: An Implementation of Evolutionary Computation Operators in OpenCL. In: Brazier, F.M.T., Nieuwenhuis, K., Pavlin, G., Warnier, M., Badica, C. (eds.) Intelligent Distributed Computing V. SCI, vol. 382, pp. 103–113. Springer, Heidelberg (2011)
10. Holland, J.: Adaptation in Natural and Artificial Systems. Univ. of Michigan Press (1975)
11. Prata, P., Fazendeiro, P., Sequeira, P.: Towards Cost-Effective Bio-Inspired Optimization: a Prospective Study on the GPU Architecture. In: Panigrahi, B.K., Suganthan, P.N., Das, S., Satapathy, S.C. (eds.) SEMCCO 2011, Part II. LNCS, vol. 7077, pp. 63–70. Springer, Heidelberg (2011)
12. Munshi, A. (ed.): The OpenCL Specification Version: 1.1, Khronos OpenCL Working Group, 385 pages (2011)
13. Eshelman, L., Schaffer, J.: Real-coded genetic algorithms and interval-schemata, vol. 3, pp. 187–202. Morgan Kaufmann, San Mateo (1993)

Optimal Location and Sizing of DG for Congestion Management in Deregulated Power Systems

K. Vijayakumar and R. Jegatheesan

SRM University, Kattankulathur, Chennai, India
kvijay_srm@rediffmail.com

Abstract. In restructured environment, with ever-increasing demand of electricity consumption and increasing open access, transmission line congestion is quite frequent. For maximum benefit and mitigation of congestion, proper sizing and location of distributed generators are necessary. This paper presents a simple method for optimal sizing and optimal placement of distributed generators. In this paper two methods are proposed for solving congestion management problem in a day ahead electricity market. In the first method the objectives considered are the minimization of the congestion cost and transmission line loss so as to relieve congestion in the system. These two objectives: (i) the cost of installation of DG and congestion index (ii) transmission line loss and congestion index, are considered to form a single objective and the problem is solved using Real coded Genetic Algorithm. This method gives only one compromised solution considering both the objectives, which does not provide any choice to the operators. Hence in the second method, both the objectives: congestion cost and transmission line loss are considered separately and solved as a multi-objective problem, using NSGA II method. This method gives a set of pareto optimal solution, so operator has a flexibility in choosing the solution based on the need. The proposed problem is implemented on IEEE 14 bus system.

Keywords: Deregulated Power Systems, Congestion Management, Distributed Generation, Genetic Algorithm.

1 Introduction

Congestion in the transmission system occurs when there is lack of coordination between generation and transmission systems. The inadequate transmission capacity to meet the demands of all customers leads to the more expensive generating units in operation which may cause the electricity markets not be able to operate at its competitive equilibrium and threaten system security. So in the recent years, many studies have been carried out to develop congestion management in restructured power industry. Line flow sensitivity for line reactance can be utilized for alleviating congestion [1]. In [2] installation of FACTS devices, for congestion management has been discussed. Planning of new power plants as well as constructing new transmission lines or expanding transmission network as long-term propositions for congestion

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 679–686, 2012.

management are presented in [3-4]. In [5] the impact of distributed resources (DR's) on congestion management is discussed. After restructuring of power systems the usage of distributed generation (DG) have increased to a large extend, which plays an increasingly major role in electric power systems operation and planning. Mostly DGs are used to improve system's indicators such as voltage profile [6] and losses [7]. The concept of transmission congestion management through DGs was first introduced in [8]. In Reference [9], a new method based on optimal power flow in which DGs were used as a tool for congestion management is proposed. Most important economic benefits bring about by DG technologies to the power system are modeled and quantified in economic terms in [10].

Conventional optimization methods suffer from the local optimality problem and some of them usually require the function to have good characteristics, such as continuity, differentiability etc., this restricts the application of these traditional methods to a small range of real world problems. So in this paper real coded genetic algorithm is considered for solving the problem. The two objectives cost of DG and real power loss are minimized as a single objective function using GA by optimal sizing and location of DG to relieve congestion in the system.

Congestion management methods available in the literature consider only one objective and provide only one solution which does not provide any choice to the operator. The non-dominated sorting genetic algorithm one of the Pareto-based approach is used widely to solve the multi-objective optimization problem. In the power system the objective of minimizing the cost of DG as well as minimizing the real power loss are essential. So in this paper the multi-objective optimization method NSGA II [11] is used to solve these two objectives simultaneously. The same problem can also be solved using the other latest multi-objective Evolutionary algorithms reported in [14-15].

The proposed method is tested with modified IEEE 14 bus system under severe line outages and overload. The solutions obtained are found to be quite encouraging and satisfactory.

2 Problem Formulation for Optimal Location and Sizing of DG

The main purpose of DGs installation considered in this paper is to minimize the cost of installation of DG and to minimize the real power loss, so as to relieve congestion in the system subjected to various operational and security constraints. The optimal placement and sizing of DG is obtained for the following studies

CASE 1: To minimize the DG cost and the congestion index.

CASE 2: To minimize the real power loss and congestion index.

The above mentioned two objectives are combined to form a single objective problem and solved using GA.

CASE 1 : The objective function for the minimization of DG cost and the congestion index can be expressed by the relation

$$F(1) = C(P_{DG}) + \alpha * CI \tag{1}$$

where

$$C(P_{DG}) = a_{DG} + b_{DG}P_{DG} + c_{DG}(P_{DG})^2 \quad (2)$$

Congestion Index(CI)= $\sum_{ij=1}^{NL}\left(P_{ij} - P_{ij}^{max}\right)^2$ (3)

α = weighting Factor

CASE 2: The objective function for the minimization of real power loss and the congestion index can be expressed by the relation

$$F(2) = E + \beta * CI \quad (4)$$

where

$$E = \sum_{ij=1}^{NL} P_{ij} + P_{ji} \quad (5)$$

where P_{ij} and P_{ji} are the real powers, P_{ij} from bus i to j and P_{ji} from bus j to i respectively

$$P_{ij} = \text{real}(V_i I_{ij}^*) \quad (6)$$

$$P_{ji} = \text{real}\ (V_j I_{ji}^*) \quad (7)$$

Congestion Index(CI)= $\sum_{ij=1}^{NL}\left(P_{ij} - P_{ij}^{max}\right)^2$ (8)

β=Weighting Factor

The set of equality and inequality constraints are

$$P_{Gi} - P_{Di} = \sum_{j=1}^{NB} |V_i||V_j||Y_{ij}| \cos\left(\delta_i - \delta_j - \theta_{ij}\right) \quad (9)$$

$$Q_{Gi} - Q_{Di} = \sum_{j=1}^{NB} |V_i||V_j||Y_{ij}| \sin\left(\delta_i - \delta_j - \theta_{ij}\right) \quad (10)$$

$$P_{Gj}^{min} \le P_{Gj} \le P_{Gj}^{max} \qquad j=1, 2....N_g \quad (11)$$

$$Q_{Gj}^{min} \le Q_{Gj} \le Q_{Gj}^{max} \qquad j=1, 2....N_g \quad (12)$$

$$V_n^{min} \le V_n \le V_n^{max} \qquad n=1, 2....N_d \quad (13)$$

$$P_{DG}^{min} \le P_{DG} \le P_{DG}^{max} \quad (14)$$

Constraints (9) and (10) correspond to active and reactive power balances at all buses. Constraints (11) and (12) provide upper and lower bounds for real and reactive power of generators. Constraint (13) establishes threshold limits for load bus voltages. Constraint (14) provides the real power limit for DG.

2.1 Multi-objective Optimization for Minimizing DG Cost and Line Loss by Optimal Sizing and Placement of DG

The optimal location and sizing of DG to minimize the DG cost and line loss as a multi-objective optimization problem is discussed in this section. Because apart from relieving congestion, the cost of DG and real power loss has to be minimized simultaneously, so as to provide a multiple solution. The objective functions considered for multi objective optimization are (1) and (5) subjected to the constraints (9) - (14).

3 Real Coded Genetic Algorithm

Genetic algorithms use the principle of natural evolution and population genetics to search and arrive at a high quality near global solution. GA is essentially a population based search algorithm. The advantage of this algorithm comes from its ability to exploit historical information structures from previous solution guesses in an attempt to increase performance of future solution structures. The use of floating-point numbers in the GA representation has a number of advantages over binary encoding. The efficiency of the GA gets increased as there is no need to convert the solution variables to the binary type. A typical GA has three phases, i.e., initialization, evaluation and genetic operations, which consists of reproduction, recombination or crossover and mutation. A Simulated binary crossover (SBX) operator, polynomial mutation and Best fitness selection are used in this paper.

4 Algorithm for Optimal Location and Sizing of DG

The step by step procedure for solving the congestion management problem discussed in section 2 by optimal location and sizing of DG using Genetic Algorithm is given below:

1. Initialize the parameters of GA and set generation k=1.
2. Randomly generate the control variables X(k) (i.e. location of load bus no and real power generation of DG) within the limit.
3. Run power flow for the parent population generated in Step 2 and compute bus voltages and line flows. Evaluate the fitness values for the parent population using (1) for minimizing the cost of generation or (4) for minimizing the real power loss respectively.
4. Select parents for recombination.
5. Create new particles using SBX crossover and polynomial mutation operation.
6. Evaluate the fitness values for the new solution vectors using N-R power flow. Combine parent (Np) and child solutions (N_P). Among 2Np individuals, best Np individuals are selected based on their fitness values.
7. Check for stopping criteria. If maximum generation count is reached then go to the next step. Else go to step 4.
8. Print the solution which yields minimum fitness value.

4.1 Multi-objective Optimization Algorithm for Optimal Placement of DG

The step by step procedure used for solving the proposed problem as given below:

(1) Set up NSGA II parameters like population size, number of generations, distribution indices for crossover (mu), and mutation (mum). Here mu and mum are taken 20 each.
(2) Read line data, bus data, incremental and decrement bidding costs for each generator. When applying evolutionary computation algorithm, the first step is to decide the control variables embedded in the individuals. In this work, control variables are location and size of DG. Hence the control variables are generated randomly satisfying their operational constraints.
(3) For each chromosome of population, calculate objective function-1 using (1) and objective function-2 using (5).
(4) The equality and inequality constraints are handled in the Newton-Raphson Power Flow.
(5) Non-domination sorting of population is carried out. Then tournament selection is applied to select the best individuals based on crowding distance.
(6) Crossover and Mutation operators are carried out to generate offspring (Qt) and the new vectors obtained must satisfy the limits if not set it to the appropriate extrema.
(7) Calculate the value of each objective function of Qt and merge the parent and offspring population to preserve elites.
(8) Again perform non-dominated sorting on the combined population based on crowding distance measure and obtain the best new parent population (Pt +1) of size N out of 2N population, so this would be the parents for next generation and this process is carried out till a maximum number of generations are reached.
(9) Finally pareto front is achieved, that is, a set of solutions satisfying both objectives are obtained.

5 Results and Discussion

The proposed algorithm is tested on IEEE 14 bus test system. The data are obtained from [12]. The DG cost characteristics are obtained from [13]. Initially Severity Index (SI) is computed for all the line outages corresponding to the base case generations and loadings, considering the outage of one line at a time and using the expression

$$SI = \sum_{ij=1}^{NL} \left(\frac{P_{ij}}{P_{ij}^{max}} \right)^2$$

The more severe lines are identified based on the severity index. The following three cases are considered for congestion management.

Case 1A is for outage of line 1-5
Case 1B is for outage of line 1-2
Case 1C is for outage of line 2-5 along with 20% increase in real power demand.

Congestion management study is conducted following the step by step procedure presented in section 4 .The cost of DG, sizing of DG and location are obtained from GA and results are tabulated in Table 1. Similarly for minimizing the real power loss, the location and sizing of DG are given in table 2. The convergence characteristics and the best individual for case 1A for minimizing the DG cost and for minimizing the real power losses are given in fig [1] and fig [2].

In most of the cases, it is found that the location of DG is at bus 4 with the size varying between 143-156 kW. Only for case 1C the location is at bus 9 with 174 kW. From this it is evident that, location and size of DG depends on the system and the severity of the congestion. Also it is found that the placement of DG improves the load bus voltages significantly. Instead of single DG, multiple DG can be used for better results.

Table 1. Cost minimization

Cases	Location of DG (Bus No)	Sizing of DG (kW)	DG Cost ($/hr)
1A	4	152.0	2280.35
1B	4	143.65	2154.74
1C	4	155.25	2328.77

Table 2. Loss minimization

Cases	Location of DG (Bus No)	Sizing of DG (kW)	Real Power Loss (kW)
1A	4	155.171	73.42
1B	4	156.714	72.377
1C	9	174.258	77.214

To optimize both cost and loss simultaneously, multi-objective optimization algorithm NSGA-II is used. It provides a set of pareto optimal solutions for the congestion problem, giving the system operator options for judicious decision in solving the congestion. The pareto optimal solution for all the three contingency cases considered are found. The pareto front for case1A is given in fig 3. Three solutions from the pareto front are obtained and presented for all the three cases in table 3.

Table 3. Three solutions from the pareto front

Cases	DG Cost $/hr	Line Loss (kW)
1A	2327.62	73.424
	2312.04	73.425
	2280.99	73.429
1B	2350.72	72.37
	2234.88	72.45
	2166.62	72.57
1C	2514.27	74.598
	2476.28	74.601
	2378.46	74.635

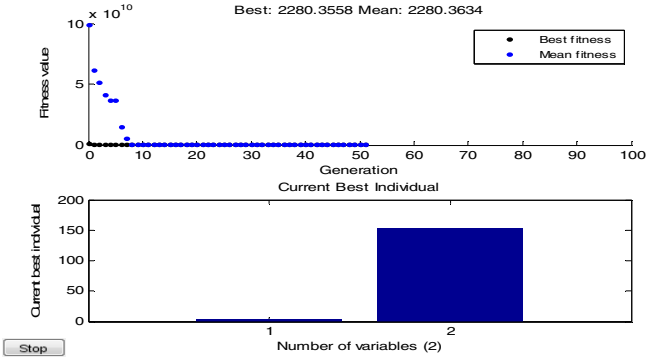


Fig. 1. Convergence characteristics for cost minimization for case 1A

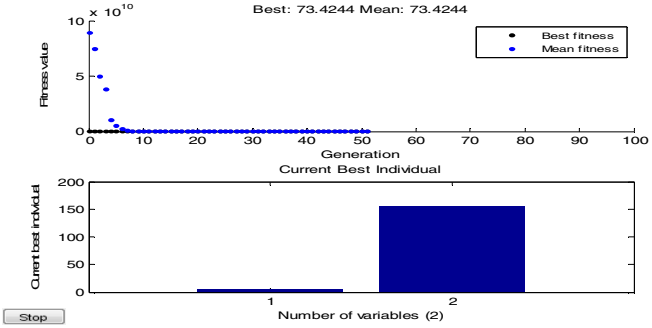


Fig. 2. Convergence characteristics for loss minimization for case 1A

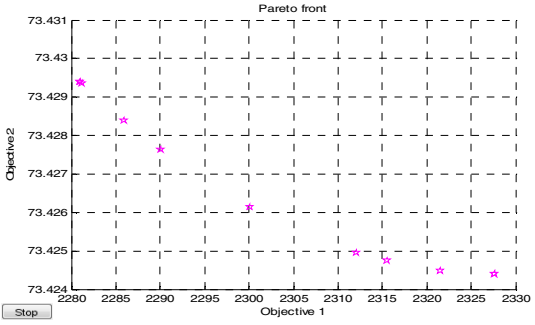


Fig. 3. Pareto front for case1A

6 Conclusion

In this paper two efficient methods are proposed for solving congestion management problem in a day ahead electricity market. This paper presents a simple method for optimal sizing and optimal placement of distributed generators. The first method

gives only one compromised solution considering both the objectives, which does not provide any choice to the operators. When this method is used to find multiple solution, it has to be run many times for finding a different solution in each simulation run. It is time consuming and cannot be used for real time problems. Though the second multi-objective optimization does not guarantee global optimal solution, it provides a very close suboptimal solution. This method also provides a set of pareto optimal solutions for the congestion problem, giving the system operator options for judicious decision in solving the congestion.

References

1. Jibiki, T., Sakakibara, E., Iwamoto, S.: Line flow sensitivities of line reactances for congestion management. In: IEEE Power Eng. Society Meeting, vol. 24, pp. 1–6 (June 2007)
2. Mithulananthan, N., Acharya, N.: Locating series FACTS devices for congestion management in deregulated electricity market. Elect. Power Syst. Res. 77, 352–360 (2007)
3. Shrestha, G.B., Fonseka, P.A.J.: Congestion-Driven transmission expansion in competitive power markets. IEEE Trans. Power Syst. 19, 1658–1665 (2004)
4. Kaymaz, P., Valenzuela, J., Park, C.S.: Transmission congestion and competition on power generation expansion. IEEE Trans. Power Syst. 22(1), 156–163 (2007)
5. Liu, J., Salama, M.M.A., Mansour, R.R.: Identify the Impact of Distributed Resources on Congestion Management. IEEE Trans. on Power Delivery 20 (2005)
6. Gil, H.A., Joos, G.: Models for quantifying the economic benefits of distributed generation. IEEE Trans. on Power Syst. 23(2), 327–335 (2008)
7. Chiradeja, P., Ramakumar, R.: An approach to quantify the technical benefits of distributed generation. IEEE Trans. Power Syst. 19(4), 764–773 (2004)
8. Ahmadigorji, M., Abbaspour T.F., A., Rajabi-Ghahnavieh, A., Fotuhi-Firuzabad, M.: Optimal DG placement in distribution systems using cost/worth analysis. Proceedings of World Academy of Science, Engg. and Tech. 37, 746–753 (2009)
9. Afkousi-Paqaleh, M., Abbaspour T.F., A., Rashidinejad, M.: Optimal locating and sizing of distributed generation for congestion management via harmony search algorithm. In: Proc. Int. Conf. on Elec. Power and Energy Conversion Syst., UAE (November 2009)
10. Afkousi-Paqaleh, M., Abbaspour, A., Rashidinejad, M., Lee, K.Y.: Optimal Placement and Sizing of Distributed Resources for Congestion Management Considering Cost/Benefit Analysis. In: 2010 IEEE Power and Energy Society General Meeting (2010)
11. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II. In: Deb, K., Rudolph, G., Lutton, E., Merelo, J.J., Schoenauer, M., Schwefel, H.-P., Yao, X. (eds.) PPSN 2000. LNCS, vol. 1917, pp. 849–858. Springer, Heidelberg (2000)
12. Zimmerman, R.D., Gan, D.: MATPOWER: A Matlab Power System, Package, Ver.3.2, Power System Engineering Research Center, Cornell University (1997), http://www.pserc.cornell.edu//Matpower
13. Ghosh, S., Ghoshal, S.P., Ghosh, S.: Optimal sizing and placement of distributed generation in a network system. Elect. Power and Energy Sys. 32, 849–856 (2010)
14. Zhao, S.Z., Suganthan, P.N., Zhang, Q.: Decomposition Based Multiobjective Evolutionary Algorithm with an Ensemble of Neighborhood Sizes. IEEE Trans. on Evolutionary Computation 16(3), 442–446 (2012)
15. Mallipeddi, R., Suganthan, P.N., Pan, Q.K., Tasgetiren, M.F.: Differential evolution algorithm with ensemble of parameters and mutation strategies. Applied Soft Computing 11(2), 1679–1696 (2011), doi:10.1016/j.asoc.2010.04.024

A Novel Strategy Adaptive Genetic Algorithm with Greedy Local Search for the Permutation Flowshop Scheduling Problem

Srinjoy Ganguly[1], Swahum Mukherjee[1], Debabrota Basu[1], and Swagatam Das[2]

[1] Dept. of Electronics & Telecommunication Engineering,
Jadavpur University, Kolkata, India
[2] Electronics and Communication Sciences Unit,
Indian Statistical Institute, Kolkata, India
swagatam.das@isical.ac.in

Abstract. This article presents a novel Genetic Algorithm with a greedy local search operator that may solve a wide range of sequencing and scheduling discrete optimization problems efficiently. To analyze its performance, we have tested the algorithm on the Permutation Flow-shop Scheduling Problem (PFSSP). Here we present a novel crossover scheme coupled with an innovative mutation scheme that implements local search to facilitate rapid convergence. This novel GA variant provides better results compared to other heuristics, which is apparent from the experimental results and comparisons with other existing algorithms.

Keywords: Genetic Algorithm, PFSSP, SAGA_GLS, Drastic Mutation, Ordered Crossover, Cyclic Crossover, Ant Colony, Greedy Local Search, Simple Inversion Mutation.

1 Introduction

In computer science, evolutionary computation is a subfield of artificial intelligence (more particularly computational intelligence) that may be used to efficiently solve a wide variety of combinatorial optimization problems. Evolutionary algorithms are stochastic optimization techniques that mimic the adaptive abilities of species on a genetic level and are used in problem solving and data correlation. Three main types of evolutionary algorithms are: (i) Genetic Algorithms [1] (ii) Evolutionary Strategies [2] (iii) Genetic Programming [3]. In this paper we shall frame a new strategy adaptive genetic algorithm (GA) equipped with a greedy local search operator that can efficiently solve the Permutation Flowshop scheduling Problem (PFSSP).

Practical machine scheduling problems are numerous and varied. They arise in diverse areas such as flexible manufacturing systems, production planning, computer design, logistics, communication, etc. The main aim of a scheduling problem is to find sequences of jobs on given machines with the objective of minimizing some function of the job completion times. In a simpler version of this problem, the flow

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 687–696, 2012.

shop scheduling problem (FSSP), all jobs pass through all machines in the same order. FSSP, which has been proven to be among the hardest combinatorial optimization problems, has emerged as one of the major cynosures of research in recent years. As the FSSP belongs to the class of NP-hard problems, there are no known algorithms guaranteed to give an optimal solution and run in polynomial time. Hence, classical optimization methods (branch and bound method [4], dynamic programming [5]) can be used only for small scale tasks. Therefore, more complex tasks must be solved by heuristic methods.

In the past a large number of problem-specific heuristics have been introduced of which the Shifting Bottleneck Procedure [6] has proved to be the most successful one. Johnson's Rule (Johnson, 1954)[7] has been the basis of many flow-shop scheduling heuristics. Palmer (1965) [8] first proposed a heuristic for the flow shop scheduling problem to minimize makespan. The heuristic generates a slope index for jobs and sequences them in a descending order of the index. Campbell *et al.* proposed Campbell, Dudek, Smith (CDS) heuristic [9] which is a generalization of Johnson's two machine algorithm; it generates a set of m-1 artificial two-machine problems from an original m-machine problem, then each of the generated problems are solved using Johnson's algorithm. . Gupta (1971) [10] used the concept of Palmer's "slope index". Nawaz *et al.* [11] proposed that, a job with longer total processing time should have higher priority in the sequence. Hundal and Rajgopal [12] made an improvement in the Palmer's method and CDS. McCormich *et al*. [13] developed a constructive heuristic, known as 'Profile Fitting' (PF). A more comprehensive approach was presented by Leisten [14] for dealing with permutation and non-permutation flowshops with finite and unlimited buffers to maximize the use of buffers and to minimize the machine blocking. Ronconi [15] proposed three constructive heuristics for blocking flow shop problems with makespan criterion. Recently, Ronconi and Henriques [16] studied the minimization of the total tardiness in flow shops with blocking scheduling and presented some constructive heuristics. Caraffa [17] developed a genetic algorithmic approach for solving large size restricted slowdown flow shop problems in which blocking flow shop problems were special cases. Grabowski and Pempera [18] developed two tabu search (TS) and TS with multi-move (TS+M) approaches. Other algorithms are discussed in [20],[21],[22] and [23].

In this paper, we deal with another special version of the problem called a permutation flow shop scheduling problem (PFSSP) where each machine processes the jobs in the same order. In this paper we are mainly concerned about the static FSSPs. Hence, we wish to propose a novel strategy adaptive genetic algorithm coupled with a greedy local search operator (SAGA_GLS) that may be used to solve classical static-flow shop instances of various dimensions, to near optimality.

2 PFSSP Problem Formulation

The Permutation Flowshop Scheduling problem (PFSSP) is one of the most important problems in the area of production management [19]. It can be briefly described as follows:

The assumptions and requirements of the static flow shop scheduling problem considered here are:

- The machine group consists of m machines M_1, M_2,. . . , M_m each performing a different function.
- All machines are available and ready to start processing at $t = 0$.
- There are n independent jobs to be scheduled, identified by integers $1,2,...,n$.
- Each job consists of m operations whose precedence structure is a strict ordering of the operations, where, for each job, the first operation requires machine M_1, the second operation requires M_2, etc.
- The required processing times (denoted $p(i,j)$ for the i^{th} operation of the j^{th} job) are known in advance.
- No interruption of an operation is allowed-each must be scheduled into a single contiguous time interval.

Mathemetical Formulation of the PFSSP

Consider three finite sets J, M, O where
J is a set of jobs 1, … , n
M is a set of machines 1, … , m and
O is a set of operations 1, … , m
Denote -
J_i : the i^{th} job in the permutation of jobs
P_i^k : the processing time of the job $J_i \in J$ on machine k
$(\forall i \in J)(\forall k \in M)$: v_i^k = waiting time (idle time) on machine k before the start of the job J_i
$(\forall i \in J)(\forall k \in M)$: w_i^k = waiting time (idle time) of the job J_i after finishing processing on machine k, while waiting for machine $k+1$ to become free.

Define the following decision variables:

$$\forall i, j \in J : x = \begin{cases} 1, \text{if job } j \text{ is assigned to the } i \text{ th position in} \\ \text{the permutatio n, i.e., } J_i = j \\ 0, \text{otherwise} \end{cases} \tag{1}$$

If we have processing times $p(i,j)$ for job i on machine j, and a job permutation $S = (J_1, J_2, \ldots, J_n)$,then we can calculate the completion times $C(J_i, j)$ as follows:

$$C(J_1,1) = p(J_1,1) \tag{2}$$

$$C(J_i,1) = C(J_{i-1},1) + p(J_i,1); \quad i = 2,....., n \tag{3}$$

$$C(J_1, j) = C(J_1, j-1) + p(J_1, j); \quad j = 2,....., m \tag{4}$$

$$C(J_i, j) = \max\{C(J_{i-1}, j) + p(J_i, j-1); \quad i = 2,....., n; \; j = 2,....., m \tag{5}$$

$$C_{\max}(S) = C(J_n, m) \tag{6}$$

Then the object of FSSP is to find out $\min(C_{\max}(S))$.

3 Key Components of the Genetic Algorithm

The key components of the proposed SAGA_GLS are:

1) Generation of the initial population: The initial population of our genetic algorithm is generated randomly. The size of the initial population may be defined by the user. Let this be denoted by *pop_size.*

2) Fitness function: Let a permutation of operations be represented by ***X.*** As described in the formulation of the permutation flow shop scheduling problem, we may calculate its net makespan. Let that be represented by *C(**X**)*. Hence the value of *f(**X**)* is the inverse of *C(**X**).* Hence, the lesser is the net makespan of the sequence of operations represented by the permutation, the greater is its fitness value.

3) Selection: The selection procedure that we have applied is the traditional *Rank-based Selection*, wherein the chromosomes of the current population are first arranged as per their fitness values in the descending order. Subsequently, the consecutive chromosomes undergo cross-over with each other, with each cross-over operation resulting in a single off spring. Finally, to complete the cycle, the chromosomes with maximum and minimum values of fitness undergo cross-over, thereby ensuring that each chromosome is crossed twice. This is done to ensure a bi-directional flow of quality.

4) Crossover Scheme: The cross-over scheme of our algorithm primarily aims to facilitate the rapid convergence of the population towards the optimal solution. Hence, the cross-over scheme has incorporated two novel features. The first one is that the cross-over scheme actually represents a pool of two cross-over operators. The two cross-over operators which contend with each other throughout the cross-over operation are:

- *Ordered Cross-Over (OX)*
- *Cyclic Cross-Over (CX)*

The second novel feature that we desire to incorporate in the cross-over scheme is: we wish to provide the genes, undergoing cross-over, a rule which they may adopt in deciding which type of cross-over they should undergo. The rule should be such that it should coercively promote that cross-over operation which is both probabilistically as well as empirically favored. Hence, we have adopted the rule utilized by ants in the ACO algorithm as proposed by Dorigo et. Al.[24], wherein each cross-over operation is consider as a path, hence, the path(cross-over operation) with greater concentration of nectar(fitter off springs) is naturally the favored one. The rule is:

Every time two genes undergo cross-over the individual probabilities for each of the two cross-over operations are computed. For ex,

$$prob_{cx} = (rand_{cx})^{A}(\Gamma_{cx})^{B} \tag{8}$$

[A and B are user-defined parameters] where, Γ_{cx} is a parameter which is updated as per the formula:

$$\Gamma_{cx} = \Gamma_{cx} + (\frac{Q}{t_{cx}}) \tag{9}$$

This updating occurs every time two genes of that generation undergo cyclic crossover. Over here, Q is a user-defined parameter and t_{cx} represents the total makespan of the flow-shop permutation represented by the off-spring, obtained by virtue of the cross-over operation. Similarly, ox_{prob} is computed and the cross-over operator with the higher probability is naturally the favored one, and so the pair of genes undergoes that cross-over operation, which has a higher probability.

Ordered Crossover (OX)

The Ordered Cross-Over operator, as proposed by Davis [25] is unique because it lays importance on the order of the operations in the flow-sequence, and not on their positions. It creates an off spring by selecting a substring, determined by randomly generated cut-points, from one of the parents while preserving the relative order of operations of the other parent. For ex, consider the two parent operation sequences: **(A B C D E F G H I J)** and **(B C D G H A E J F I)**.

Now we select two random cut points. Suppose that we select a first cut point between the 2nd and the 3rd bit and a second one between the 7th and the 8th bit. Next, starting from the second cut point of one parent, the rest of the operations are copied in the order in which they appear in the other parent, also starting from the second cut point and omitting the operations that are already present. When the end of the parent string is reached, we continue from its first position. In our example this gives the following children: **(J I |C D E F G| B H A E)** and **(I J |D G H A E| B C F G)**.

As we have imposed the restriction that each cross-over operation shall result in a single offspring, the fitter of the two progeny is selected and hence, labeled as the product of the ordered cross-over operation. Subsequently, the net makespan of the resultant offspring is stored in $\boldsymbol{t_{ox}}$ and the value of $\boldsymbol{\Gamma_{ox}}$ is updated as per the rule, stated at the beginning of this section.

Cyclic Crossover (CX)

The Cyclic Crossover Operator, as proposed by Dagli and Sittisathanchai[26], is unique because it identifies the so-called cycles between the two parent chromosomes. The crossover mechanism may be envisaged via the following example : Let us consider two flow sequences A and B, where **A= (1 3 5 6 4 2)** and **B= (5 6 1 2 3 4)** . Let the first crossover product begin with 1(the starting operation of A). Selection of '1' from A implies that '5' should be selected from **B**,

because we want each operation to be derived from one of the two parents. Hence, **C= 1 _ 5 _ _ _**. This process continues on till after the selection and subsequent insertion of an operation from one of the two parents, the operation in the corresponding position in the other parent is already present in the offspring. After this, the remaining operations are filled in, as per their orders respective to one another in the other string. Hence, **C = (1 6 5 2 3 4)**. Similarly, considering '5' as the starting operation, another offspring D can be obtained in a similar fashion. Hence, **D= (5 3 1 6 4 2)** .As our algorithm imposes a restriction that the result of each crossover operation results in only a single offspring, only the fitter of the two progeny is selected as the final result of the Cyclic Crossover operation. The makespan of the fitter of the two probable flow-shop permutations is stored in t_{cx} and the value of Γ_{cx} is updated as stated at the beginning of this section.

5) Mutation Scheme

- **Forced Mutation(Local Search):** We propose to introduce a novel mutation scheme with greedy traits that may act as a local search operator, which every off spring obtained as a result of the cross-over operation is forced to undergo, in order to enhance the rate at which the population converges to the optimal solution. To do so, we have used a modified version of the *Exchange Mutation Operator*, which is unique primarily because it can change the very fabric of the flow-shop sequence by randomly swapping the positions of two operations (which are randomly selected as well). We wish to induce the requisite greediness into this mutation operator by trying out this mutation operation on every possible pair of operations represented in the flow sequence. From all the probable flow-sequences that are obtained as a result of this operation, the one with the minimum net makespan is selected and if it is fitter than the original flow sequence, it replaces the flow sequence.
- **Drastic Mutation:** The primary aim of this operation is to prevent one of the most common drawbacks of genetic algorithms: convergence to a solution which may be very fit but isn't the optimal solution, by introducing variety. Hence, if the fittest solution of five consecutive generations is the same, then any ten chromosomes, with comparatively poor fitness values, are selected from the current population, and are replaced with randomly generated flow-shop permutations.

6) Final Screening and Subsequent Progression to the next generation:

The last step of each generation is as follows: The Initial population (which is derived from the previous generation) and the new population (which is computed after the crossover and mutation operations) are collected together. Then, the fittest *pop_size* chromosomes are selected. The best-find of the current generation is displayed and the genetic algorithm enters the next generation (iteration).

```
begin  SAGA_GLS
        create the Initial population
        while gen_count<=k
    /*k-Maximum Number of iterations*/
          begin
          Rank-based  Selection
          if  tau_ox>tau_cx
 Ordered Cross-over is invoked
              tau_ox is updated
          else
  Cyclic Cross-over is invoked
            tau_cx is updated
          end
  Forced Greedy Exchange Mutation (Local Search) is invoked
 if repition_consecutiveBest = = true

   /* repition_consecutiveBest holds true if the best fit solution of the last five
generations is same*/

    Drastic Mutation is invoked
          end

      Final Screening & Subsequent Progression to the next generation
     end
        Output the individual best find
end
```

Fig. 1. SAGA_GLS pseudocode

4 Results and Discussion

In our tests for PFSSP, the proposed genetic algorithm was coded in MATLAB 7.10 and the experiments were executed on a Pentium P-IV 3.0 GHz PC with 512MB memory. To test the performance of the proposed genetic algorithm, a comprehensive experimental evaluation and comparison with other powerful methods is presented based on the flow shop benchmark set of Taillard. The benchmark set is composed of 12 subsets of given problems with the size ranging from 20 jobs and five machines to 500 jobs and 20 machines, and each subset consists of ten instances. We treat them as the permutation flowshop scheduling problems with makespan criterion. Table 1 to table 4 summarize the best upper bounds or makespans obtained over 10 runs by various methods like HDDE developed by Liang Wang *et al*., TS+M proposed by Grabowski and Pempera, Ron's algorithm (here denoted as RON) and our proposed SAGA_GLS. Table 1, table 2 and table 3 are categorized according to the no. of machines used in the FSSP instances (no. of jobs n = 20, 50, 100 in each case). Table 4 is used for higher no. of jobs (n = 200, 500). No. of machines and jobs is indicated as $m*n$ in the first column for every particular case.

Table 1. Comparison of upper bounds obtained using SAGA_GLS, HDDE, TS+M and RON for different PFSSP instances (no. of machines m= 5)

5*20				5*50				5*100			
SAGA_GLS	*HDDE*	*TS+M*	*RON*	***SAGA_GLS***	*HDDE*	*TS+M*	*RON*	***SAGA_GLS***	*HDDE*	*TS+M*	*RON*
1374	1374	1387	1384	3033	3033	3163	3151	**6151**	6291	6639	6455
1408	1408	1424	1411	**3206**	3226	3348	3395	6136	6136	6481	6214
1280	1280	1293	1294	3039	3039	3173	3184	6063	6063	6299	6124
1448	1448	1451	1448	**3128**	3147	3277	3303	5839	5839	6120	5976
1341	1341	1348	1366	**3158**	3192	3338	3272	**5972**	6065	6340	6173
1363	1363	1366	1363	3183	3183	3330	3400	5971	5971	6244	6094
1381	1381	1387	1381	**3022**	3054	3168	3228	6095	6095	6346	6262
1379	1379	1388	1384	3081	3081	3228	3260	5985	5985	6289	6061
1373	1373	1392	1378	**3068**	3929	3068	3104	6234	6234	6559	6474
1283	1283	1302	1283	3146	3146	3285	3264	**6159**	6273	6509	6366

Table 2. Comparison of upper bounds obtained using SAGA_GLS, HDDE, TS+M and RON for different PFSSP instances (no. of machines m = 10)

10*20				10*50				10*100			
SAG	*HDDE*	*TS+M*	*RON*	***SAGA_GLS***	*HDDE*	*TS+M*	*RON*	***SAGA_GLS***	*HDDE*	*TS+M*	*RON*
1698	1698	1698	1736	3667	3667	3776	3913	**7042**	7131	7320	7496
1833	1833	1836	1897	**3507**	3523	3641	3798	**6790**	6816	7108	7281
1659	1659	1674	1677	**3504**	3515	3588	3723	6956	6956	7233	7400
1535	1535	1555	1622	**3652**	3685	3786	3885	**7186**	7261	7413	7670
1617	1617	1631	1658	3650	3650	3745	3934	**6824**	6913	7168	7317
1590	1590	1603	1640	3622	3622	3747	3831	**6683**	6739	6993	7301
1622	1622	1629	1634	3778	3704	3778	3957	**6801**	6874	7092	7247
1731	1731	1754	1741	3596	3590	3708	3774	**6882**	6940	7143	7315
1747	1747	1759	1777	3556	3556	3668	3784	**7055**	7133	7327	7631
1782	1782	1782	1847	3642	3642	3729	3928	**6966**	7065	7299	7411

Table 3. Comparison of upper bounds obtained using SAGA_GLS, HDDE, TS+M and RON for different PFSSP instances (no. of machines m = 20)

20*20				20*50				20*100			
SAGA_GLS	*HDDE*	*TS+M*	*RON*	***SAGA_GLS***	*HDDE*	*TS+M*	*RON*	***SAGA_GLS***	*HDDE*	*TS+M*	*RON*
2436	2436	2449	2530	4516	4516	4627	4886	7891	7891	8101	8347
2234	2234	2242	2297	4296	4296	4411	4668	7931	7931	8105	8372
2479	2479	2483	2560	4290	4290	4388	4666	**7826**	7935	8071	8265
2348	2348	2348	2399	**4375**	4393	4479	4650	7930	7930	8081	8365
2435	2435	2450	2538	**4276**	4284	4359	4475	**7900**	7944	8074	8304
2383	2383	2398	2467	**4286**	4308	4372	4521	**7886**	7971	8151	8450
2390	2390	2397	2502	**4318**	4325	4402	4576	**7932**	8051	8273	8507
2345	2328	2345	2411	**4322**	4337	4444	4688	**8022**	8102	8248	8584
2363	2363	2363	2421	**4314**	4332	4423	4532	**7966**	8007	8116	8341
2323	2323	2334	2407	**4432**	4439	4609	4846	8050	8050	8261	8489

Table 4. Comparison of upper bounds obtained using SAGA_GLS, HDDE, TS+M and RON for different PFSSP instances (for higher no. of jobs)

10*200				20*200							
SAGA_GLS	*HDDE*	*TS+M*	*RON*	***SAGA_GLS***	*HDDE*	*TS+M*	*RON*	***SAGA_GLS***	*HDDE*	*TS+M*	*RON*
13756	13756	14220	14113	**14927**	15057	15334	15579	**36637**	37172	37860	8334
13320	13621	14089	14127	**15002**	15284	15522	15728	**36936**	37485	38044	8642
13416	13741	14149	14416	**15186**	15360	15713	15915	**36646**	37209	37732	8163
13352	13718	14156	14435	**15082**	15276	15687	16039	**36641**	37291	38062	8625
13360	13721	14130	14119	**14986**	15183	15443	15938	**36598**	37232	37991	8492
13196	13474	13963	13909	**15113**	15223	15472	15911	**37002**	37513	38132	8551
13644	13925	14386	14563	**15106**	15296	15522	15898	**36524**	37121	37561	8179
13512	13863	14256	14329	**15141**	15310	15540	16022	**36841**	37202	37750	8664
13329	13659	13954	13923	**15034**	15243	15394	15817	**36647**	37116	37730	8339
13467	13744	14224	14435	**15116**	15284	15523	15969	**36862**	37492	38014	8540

5 Conclusions and Further Work

In this paper we have used GA in a different approach and investigated its suitability in a discrete optimization problem, namely FSP. Here we have mainly looked for better convergence, speed and accuracy. To meet these goals, we have incorporated several novel features in our GA. Firstly, we have adopted a new adaptive cross-over scheme wherein, the mode of cross-over isn't fixed, instead, depends entirely upon the nature of the chromosomes undergoing cross-over. Accordingly, the cross-over algorithm to which they respond more favorably is coercively promoted, without by any means, limiting the heuristic nature of the algorithm. Secondly, a novel greedy exchange mutation operator is introduced which every off spring is forced to go through. This is done in order to facilitate the increased rate of convergence. At the same time, we have also introduced the novel concept of drastic mutation whose sole aim is to prevent the convergence of the members of the population to a solution which is fit in its own right, but isn't the optimal one. Thus, using this algorithm we have optimized the speed and accuracy of the search process. Last but not the least, we may also add that the algorithm designed by us is not specific to the definition of this problem alone, but on the other hand, is a general combinatorial optimizer which can be used for various problems.

References

[1] Holland, J.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975)
[2] Fogel, L.J.: Atonomous Automata. Ind. Res. 4, 14–19 (1962)
[3] Bremermann, H.J., Rogson, M., Salaff, S.: Search by Evolution. In: Maxfield, M., Callahan, A., Fogel, L.J. (eds.) Biophysics and Cyberntic Systems, pp. 157–167. Spartan Books, Washington (1965)
[4] Taha Hamdy, A.: Operations Research-An Intoduction, 8th edn. pp. 355–361. Pearson Education Inc. (2003) ISBN: 0131889230
[5] Richard, B.: Dynamic Programming. Princeton University Press (2003)

[6] Joseph, A., Egon, B., Daniel, Z.: The Shifting Bottleneck Procedure for Job-shop Scheduling. Management Science 34(3) (March 1988); printed in USA
[7] Baker Kenneth, R., Dan, T.: Principles of Sequencing and Scheduling, pp. 230–234. John Wiley & Sons Inc. (2009)
[8] Panneerselvam, R.: Production and Operations Management. Business and Economics. Prentice Hall India (PHI) (2006)
[9] Campbell Herbert, G., Dudek Richard, A., Smith Milton, L.: A heuristic algorithm for the n-job m-machine Sequencing Problem. Management Science 16(10), 879–885 (1970)
[10] Gupta, J.N.D.: Operational Research Quarterly (1970-1977) 22(1), 39–47 (1971)
[11] Muhammad, N.: A heuristic algorithm for the n-jon m-machine flow shop sequencing problem. Elseiver Omega 11(1), 91–95 (1983)
[12] Hundal Tejpal, S., Jayant, R.: An extension of Palmer's heuristic for the flowshop scheduling problem. International Journal of Production Research 6(6), 1119–1124 (1988)
[13] McCormich, S.T., Pinedo, M.L., Shenker, S., Wolf, B.: Sequencing in an assembly line with blocking to minimize cycle time. Operations Research (37), 925–936 (1989)
[14] Leisten, R.: Flowshop sequencing problems with limited buffer storage. International Journal of Production Research (28), 2085–2100 (1990)
[15] Ronconi, D.P.: A note on constructive heuristics for the flowshop problem with blocking. International Journal of Production Economics (87), 39–48 (2004)
[16] Ronconi, D.P., Henriques, L.R.S.: Some heuristic algorithms for total tardiness minimization in a flowshop with blocking. OMEGA—International Journal of Management Science 37(2), 272–281 (2009)
[17] Caraffa, V., Ianes, S., Bagchi, T.P., Sriskandarajah, C.: Minimizing makespan in a blocking flowshop using genetic algorithms. International Journal of Production Economics (70), 101–115 (2001)
[18] Grabowski, J., Pempera, J.: The permutation flow shop problem with blocking. A tabu search approach. Computers and Operations Research (35), 302–311 (2007)
[19] Blazewicz, J., Ecker, K.H., Schmidt, G., Weglarz, J.: Scheduling Computer and Manufacturing Processes. Springer, Berlin (1996)
[20] Ishibuchi, H., Misaki, S., Tanaka, H.: Modified Simulated Annealing Algorithms for Flow Shop Sequencing Problem. European Journal of Operational Research 81, 388–398 (1995)
[21] Turki, U.A., Fedjki, C., Andijani, A.: Tabu Search for a Class of Single-Machine Scheduling Problems. Computers & Operations Research 28, 1223–1230 (2001)
[22] Yamada, T., Reeves, C.R.: Permutation Flowshop Scheduling by Genetic Local Search. In: Proc. of the International Conference Genetic Algorithms in Engineering Systems: Innovations and Applications, GALESIA 1997, Glasgow, pp. 232–238(1997)
[23] Koulamas, C.: A New Constructive Heuristic for the Flowshop Scheduling Problem. European Journal of Operational Research 105, 66–71 (1998)
[24] Dorigo, M., Gambardella, L.M.: Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computing, 744–752 (1997)
[25] Davis, L.: Applying Adaptive Algorithms to Epistatic Domains. In: Proceedings of the International Joint Conference on Artificial Intelligence, pp. 162–164 (1985)
[26] Cihan, D., Sinchai, S.: Genetic Neuro-Scheduler for Job Shop Scheduling. International Journal of Production Economics 41(1-3), 135–145 (1993)

Estimation of Autocorrelation Space for Classification of Bio-medical Signals

Mihir Narayan Mohanty[1,*] and Aurobinda Routray[2]

[1] ITER, Siksha 'O' Anusandhan University, Bhubaneswar, Odisha, India
[2] Department of Electrical Engineering, IIT Kharagpur, WB, India

Abstract. Classification of biomedical signals is a complex task, but the analysis is very useful in medical diagnosis. In this paper we estimate the autocorrelation matrix of some brain signal by embedding the autocorrelation cone using Linear Matrix Inequalities (LMI). The minimum sample window has been chosen for the improved computational complexity. The partitioning of the space has been carried out using support vector machines. This method has been tested on different EEG signals recorded on subjects performing a multiplication, thought for composition of a song. The base signature has been recorded while the subject apparently was not doing anything.

Keywords: Biomedical brain signals, EEG, Classification, Autocorrelation space, Support Vector Machines (SVM), Linear Matrix Inequaities (LMI), Singular Value Decomposition (SVD).

1 Introduction

The classification of EEG pattern plays an important role in Brain-computer interface (BCI) systems. Most BCI systems make use of mental tasks that lead to distinguishable electroencephalogram (EEG) signals of different classes [1-4]. However, EEG data is very noisy and has different types of artifacts. Moreover it consists of signal mixtures of several brain sources and noise sources which make the problem of classification even more difficult.

Several attempts have been made to build a BCI system based on EEG signal. The classification problem can be divided into:

(1) feature extraction
(2) classification.

Some methods like Principal Component Analysis, Fourier analysis, adaptive autoregressive methods have been proposed for feature extraction [5-6]. Similarly, Artificial Neural Networks (ANN), Hidden Markov Models (HMM), Linear Discriminant Analysis (LDA) have been used for classification, by many authors. Also the optimization has been applied for the features in literature [7].

* Corresponding author.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 697–704, 2012.

In this paper, SVM has used for classification of EEG signals. It is an effective machine learning method proposed by Vapnik [8] for classification. SVM in its simplest form is a two-class classifier whose goal is to find a hyper plane such that maximum number of points of same class are on same side while maximizing the distance of either class from the hyper-plane. For feature extraction, we find the auto-correlation matrix of EEG signal followed by Singular Value Decomposition (SVD) [9]. The singular values thus obtained are used as feature.

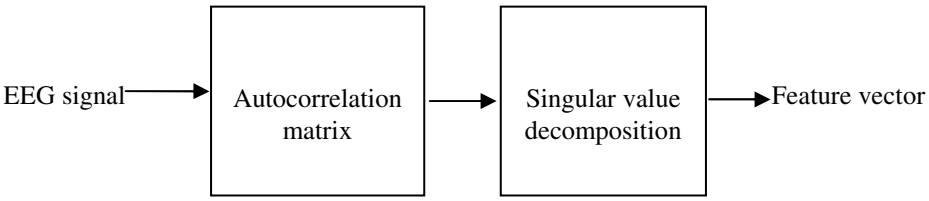


Fig. 1. Method for feature extraction

2 Approximation of Autocorrelation Matrix

Autocorrelation sequence estimation has been done for embedding or approximating the autocorrelation sequence using frequency-domain and LMI characterization. The robust optimization techniques and interior point algorithms have only been studied in the noise-free deterministic and convex setting [10-11]. For the non-stationary signal case, major technical issues are:

(a) Convergence and convergence speed;
(b) Robustness to modeling error and noise statistics.

Estimation of auto-correlation sequence can be described as the following optimization problem [10-12]:

$$\text{minimize } \|x-\hat{r}\|^2$$
$$\text{subject to } x \succeq 0,$$

with $x \in R^{n+1}$, in which the $\hat{r}$ is the given approximation for autocorrelation sequence and x is the estimate. $\succeq$ represents with respect to auto-correlation cone.

Frequency domain approach:

$$\text{minimize } \|x-\hat{r}\|^2$$
$$\text{subject to } x_0 + 2\sum_{k=1}^{n} x_k \cos(k\omega_i) \geq 0,\ i=0,...,N$$

This method works well provided N is sufficiently large (typically N=15n). However, this method is not exact, it is possible to satisfy the given constraints and still have X(ω)<0 for some ω.

LMI embedding:

$$\text{minimize } \|x-\hat{r}\|^2$$

$$\text{subject to } \begin{bmatrix} P & \tilde{x} \\ \tilde{x}^T & x_0 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 0 & P \end{bmatrix} \succeq 0$$

which has variables x and $P \in S^n$.

Both of these methods suffer from the fact that a large number ($O(n^2)$) of auxiliary variables are introduced. The implementation of algorithms can be obtained here.

Singular value decomposition as a noise reduction scheme:

Let X be a $m \times n$ matrix. The equation of the singular value decomposition of X is the following:

$$X = USV^T \tag{1}$$

where U is a $m \times m$ matrix, S is a $m \times n$ diagonal matrix, and V^T is also $n \times n$ matrix. The columns of U form an orthogonal basis, i.e.

$$u_i . u_j = \begin{cases} 0 \text{ if } i \neq j \\ 1 \text{ if } i = j \end{cases} \text{ where } u_i \text{ is a column of U}$$

Similarly, the rows of V form an orthogonal basis. S is diagonal matrix with the diagonal elements as the singular values of the matrix X. For a symmetric matrix X, U = V

Therefore, $X = USU^T$

$$X = [\vec{u}_1\ \vec{u}_2\ \vec{u}_3 \ldots \vec{u}_m] \begin{bmatrix} \sigma_1 & 0 & 0 \ldots . 0 \\ 0 & \sigma_2 & 0 \ldots . 0 \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ 0 & 0 & \sigma_m \end{bmatrix} [\vec{v}_1\ \vec{v}_2\ \vec{v}_3 \ \ldots \vec{v}_n]^T \tag{2}$$

The LMI characterization allows us to represent the constraint $x \in \mathbb{C}_{n+1}$ exactly by introducing new matrix variables Y or P. Hence, problem can be equivalently stated as

$$\begin{aligned} &\text{minimize } \|x-\hat{r}\|^2 \\ &\text{subject to } x_k = \text{Tr}\, E^k\, Y, k = 0,\ldots,n \\ &\qquad Y \succeq 0 \end{aligned} \tag{3}$$

with variables $x \in \Re^{n+1}$ and $Y = Y^T \in \Re^{(n+1)\times(n+1)}$. An alternative form is

$$\begin{aligned} &\text{minimize } \|x-\tilde{x}\|^2 \\ &\text{subject to } \begin{bmatrix} P & \tilde{x} \\ \tilde{x}^T & x_0 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 0 & P \end{bmatrix} \succeq 0 \end{aligned} \tag{4}$$

which has variables x and $P = P^T \in \Re^{n\times n}$. In both cases we obtain a cone linear program with a linear matrix inequality and a second order cone constraint. No approximation is involved here, in contrast with the sampling method. The drawback of the LMI representation is the large number ($O(n^2)$) of auxiliary variables that have to introduce.

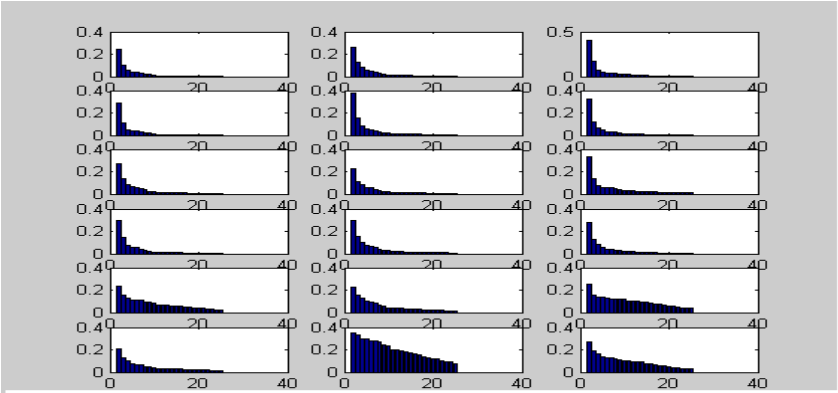


Figure : Feature vectors calculated are shown. Three columns correspond to three different mental tasks and six rows correspond to six different electrodes.

3 Separation of the AR Space Using Support Vector Machine

Support Vector Machine is a powerful, accurate, and efficient classifier. Multi class Support Vector Machines (SVMs), though accurate, are not preferred in applications requiring great classification speed, due to the number of support vectors being large as well as the complexity is more.

Binary classifiers are easy to implement with good computation speed. It is capable of dealing with high dimensional input features with theoretical bounds on the generalization error. Classifiers based on SVM have a few parameters requiring tuning. These are simple to implement and are trained through optimization of a convex quadratic cost function, which ensures a global and unique solution. Moreover, the trained SVM is defined only by the most informative training data (support vectors) and hence is sparse compared to the training data. Only a few extreme vectors from each class are required to fit the separating hyper plane in the feature (transformed) space [13-14].

Let T_N be set of N labeled data points in an M-dimensional hyperspace:

$$T_N = [(\mathbf{x_1}, d_1), \cdots, (\mathbf{x_N}, d_N)] \in (\mathbf{X} \times \mathbf{D})^N \tag{5}$$

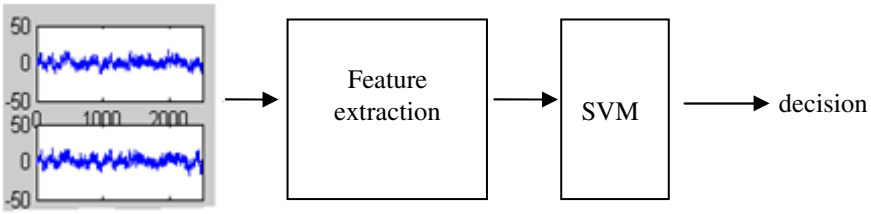


Fig. 2. Method for Classification

in which

$\mathbf{x_i} \in \mathbf{X}$, where $\mathbf{X}$ is the input space and $d_i \in \mathbf{D}$ where $\mathbf{D} = \{-1, +1\}$ is the label space. The problem is formulated to design a function ψ such that

$$\psi : \mathbf{X} \rightarrow \mathbf{D} \qquad \text{predicts } d \text{ from the input } \mathbf{x}. \tag{6}$$

Under normal circumstances $\mathbf{X}$ cannot be partitioned by a linear decision boundary. However, '$\mathbf{X}$' can be transformed into an equal or higher dimensional feature space for making it linearly separable (Cover's Theorem) [13]. Now the problem of finding a *nonlinear* decision boundary in $\mathbf{X}$ has been transformed into a problem of finding the optimal *hyperplane* for separating the two classes. The hyperplane in this transformed domain (called the feature space) can be parameterized by $(\mathbf{w}, b)$ pair as:

$$\sum_{j=1}^{P} w_j \phi_j (\mathbf{x}) + b = 0 \tag{7}$$

The mapping $\phi(\cdot)$ need not be computed explicitly; instead, an inner product Kernel [14] of the form

$$\left\langle \phi(\mathbf{x_i}), \phi(\mathbf{x_j}) \right\rangle = K(\mathbf{x_i}, \mathbf{x_j}) \tag{8}$$

can be used for finding the optimal hyperplanes.

Given a training set T_N find the optimal values of the weight vector $\mathbf{w}$ and bias b such that they satisfy the constraint

$$d_i \left(\mathbf{w^T x_i} + b \right) \geq 1 - \zeta_i, \text{ for } i = 1, 2, \cdots, N \tag{9}$$

$\zeta_i \geq 0$, for all i, and such that the weight vector $\mathbf{w}$ and the slack variables ζ_i minimize the cost functional

$$\Phi(\mathbf{w}, \zeta) = \frac{1}{2} \mathbf{w^T w} + C \sum_{i=1}^{N} \zeta_i \tag{10}$$

where C is user-specified positive parameter. The dual of this optimization problem can be formulated as

$$\text{maximize } Q(\alpha) = \sum_{i=1}^{N} \alpha - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \mathbf{x_i^T x_j} \tag{11}$$

$$\text{subject to} \qquad \sum_{i=1}^{N} \alpha_i d_i = 0 \tag{12}$$

with $0 \leq \alpha_i \leq C \quad \text{for } i = 1, 2, \cdots, N$

$\mathbf{\Phi}$ maps input space to a similar or higher dimensional space where classification is linear. The optimum weights can be calculated from the optimized values of the dual variables α.

$$\mathbf{w_0} = \sum_{i=1}^{N} \alpha_{0,i} d_i \phi(\mathbf{x_i}) \tag{13}$$

where, $\mathbf{w_0}$ is the optimum value of the weight vector – the first element being the optimum value of the bias and $\boldsymbol{\alpha}_0$ is the optimum vector representing the dual variables. The decision function can be written in the feature space as

$$y(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i d_i K(\mathbf{x_i}, \mathbf{x}) + b \tag{14}$$

where, $K(\mathbf{x_i},\mathbf{x})$ represents the Kernel [13-14].

The final classification can be obtained through,

$$\text{if } y(\mathbf{x}) > 0 \ \mathbf{x} \text{ is in class 1}$$
$$y(\mathbf{x}) < 0 \ \mathbf{x} \text{ is in class 2}$$

4 Experiment

The experimental EEG data for this paper has been obtained from the eeg machine Model No: Quest 321, Polysomnograph. There are 10 subjects carrying out three different mental activities.

- Sitting Idle
- Performing a Complex Multiplication
- Thought for a Song

Different lobes of our brain are responsible for various types of activities. As example, the Frontal lobe is associated with planning, reasoning, movement, emotion and problem solving etc; the Parietal lobe is associated with movement, recognition, perception of stimuli etc; the Temporal lobe is associated with recognition and perception of auditory stimuli, memory, and speech. Thus depending on the requirement of this work, the EEG signal are collected from the following locations: F3, F4, T3, T4, P3, P4 follows the international 10–20 systems. The sampling freq was 256 Hz. The machine in the lab used for the work is Model No: Quest 321, Polysomnograph. The six electrodes for recording the EEG generated out of the different mental tasks. The signals from different electrodes are taken to test the accuracy of the classification with each of the above kernels. There are 2560 samples of the signals recorded from each electrode for a subject carrying out any of the above tasks.

5 Result and Discussion

The EEG signal has been recorded at a sampling frequency of 256 Hz. These samples are passed on to the modular SVM structure for classification. For testing the classification algorithm, then different trails were performed. Of these, the electrode for which the feature vector of one signal was most distinguishable from other two signals was selected. SVM was trained using Gaussian kernel with $\sigma = 0.5$, 1.0 or 2.0. Best results were obtained for $\sigma = 0$. It was found that feature vector of signal A was

most distinguishable from signal B and C for electrode six. The binary classifier was trained for two classes- belonging to A or not belonging to A. SVM was trained in Matlab environment using SVM toolbox. For distinguishing B and C, it was seen that for electrode five, the feature vectors had most marked difference. So, SVM was trained for these two classes.

The recorded signals have been divided into two groups i.e. training and testing respectively. The SVM classifier is designed using the training set and tested with the other case. It was found the SVM exhibited very small error for the training set. The result for the test set is shown in Table 1.

Table 1. Test results for EEG signal corresponding to three different mental tasks

	Signal A	**Signal B**	**Signal C**
Accuracy	93.33%	95%	95%

6 Conclusion

It was also concluded that the feature found by partitioning the auto-correlation space can serve as a good parameter for describing a stationary signal. The results of classification using it as feature were also very good. LMI approximation of auto-correlation sequence helped in increasing the classification accuracy further. However, the results were obtained for a small data; the work can be continued for large set of data and also for various patient conditions.

References

[1] Huang, W.-Y., Shen, X.-Q., Wu, Q.: Classifying the number of EEG current sources using support vector machines. In: Proceedings of International Conference on Machine Learning and Cybernetics, vol. 4, pp. 1793–1795 (November 2002)

[2] Garcia, G.N., Ebrahimi, T., Vesin, J.M.: Support vector EEG classification in the Fourier and time-frequency correlation domains. In: Proceedings of First IEEE EMBS Conference on Neural Engineering, pp. 591–594 (March 2003)

[3] Lee, H., Choi, S.: PCA-based linear dynamical systems for multichannel EEG classification. In: Proceedings of the 9th ICONIP 2002, vol. 2, pp. 745–749 (November 2002)

[4] Guyon, Elisseeff, A.: An introduction to variable and feature selection. J. Machine Learning Res. 3, 1157–1182 (2003)

[5] Müller, K.-R., Anderson, C.W., Birch, G.E.: Linear and Nonlinear Methods for Brain–Computer Interfaces. IEEE Trans. on Neural Systems and Rehabilitation Engineering 11(2), 165 (2003)

[6] Pfurtscheller, G., Neuper, C., Schlogl, A., Lugger, K.: Separability of EEG signals recorded during right and left motor imagery using adaptive autoregressive parameters. IEEE Trans. on Neural Systems and Rehabilitation 6(3), 316–325 (1998)

[7] Mohanty, M.N., Routray, A., Kabisatpathy, P.: Optimization of Features using Evolutionary Algorithm for EEG Signal Classification. Int. J. Computational Vision and Robotics 1(3), 297–310 (2010)
[8] Vapnik, V.: The Nature of Statistical Learning Theory. Springer, New York (1995)
[9] Groutage, D., Bennink, D.: A new matrix decomposition based on optimum transformation of the singular value decomposition basis sets yields principal features of time-frequency distributions. In: Proceedings of the Tenth IEEE Workshop on Statistical Signal and Array Processing (August 2000)
[10] Popovici, V., Thiran, J.P.: Pattern recognition using higher-order local autocorrelation coefficients. In: 12th IEEE Workshop on Neural Networks for Signal Processing, September 4-6, pp. 229–238 (2002)
[11] Luo, Z.-Q.: Applications of convex optimization in signal processing and digital communication. Math. Program., Ser. B 97, 177–207 (2003)
[12] William Helton, J., McCullough, S., Putinar, M., Vinnikov, V.: Convex Matrix Inequalities Versus Linear Matrix Inequalities. IEEE Trans. on Automatic Control 54(5), 952–964 (2009)
[13] Haykins, S.: Neural Networks, 2nd edn. Prentice Hall (1999)
[14] Sathiya Keerthi, S., Chapelle, O., DeCoste, D.: Building Support Vector Machines with Reduced Classifier Complexity. Journal of Machine Learning Research 7, 1493–1515 (2006)

Dimension Reduction Using Clustering Algorithm and Rough Set Theory

Shampa Sengupta[1] and Asit Kumar Das[2]

[1] Department of Information Technology,
MCKV Institute of Engineering, Liluah,
Howrah – 711 204, West Bengal, India
[2] Dept. of Computer Sc. & Tech.,
Bengal Engineering & Science University, Shibpur,
Howrah – 711 103, West Bengal, India
shampa2512@yahoo.co.in,
akdas@cs.becs.ac.in

Abstract. In real world, datasets have large number of attributes but few are important to describe them properly. The paper proposes a novel dimension reduction algorithm for real valued dataset using the concept of Rough Set Theory and clustering algorithm to generate the reduct. Here, projection of dataset based on two conditional attributes C_i and C_j is taken and K-means Clustering algorithm is applied on it with K = number of distinct values of decision attribute D of the dataset to obtain K clusters. Also the dataset is clustered into K-groups using Indiscernibility relation applied on the decision attribute D. Then the connecting factor k of combined conditional attributes (C_i C_j) with respect to D is calculated using two cluster sets and attribute connecting set ACS = {($C_iC_j \overset{k}{\rightarrow} D$) for all C_i, $C_j \in C$, Conditional attribute set, and D (Decision attribute)} is formed. Each element ($C_iC_j \overset{k}{\rightarrow} D$) $\in ACS$ implies that C_i and C_j connecting together partition the objects that yields (k*100) % similar partitions as made on D. Now an undirected weighted graph with weights as the connecting factor k is constructed using attribute connecting set ACS. Finally based on the weight associated with edges, the important attributes, called reduct are generated. Experimental result shows the efficiency of the proposed method.

Keywords: Dimension Reduction, Clustering algorithm, Rough Set Theory, Attribute Similarity, Indiscernibility Relation, Undirected weighted graph.

1 Introduction

Feature selection [1] and reduct generation [2] are frequently used as a pre-processing step to data mining and knowledge discovery [3]. It selects an optimal subset of features from the feature space according to a certain evaluation criterion. In recent years, dimension of datasets has increased rapidly in many applications which bring great difficulty to data mining and pattern recognition. This enormity may cause

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 705–712, 2012.

serious problems to many machine learning [4] algorithms with respect to scalability and learning performance. Therefore, feature selection and reduct generation become very necessary for data analysis when facing high dimensional data. Rough Set Theory (RST) [5, 6], a new mathematical approach to imperfect knowledge, is popularly employed to evaluate significance of attributes and helps to find the reduct.

But finding reduct by exhaustive search of all possible combinations of attributes is an NP-Complete problem and so some heuristic approach should be applied. Our previous work [7] developed a dimension reduction algorithm for discretized datasets using Minimum Spanning Tree (MST) method for generation of reduct. In this paper, a novel reduct generation method is proposed by combining the concept of indiscernibility relation [5, 6] of RST, clustering algorithm [8] and graph theory[9].

The concept of Indiscernibility relation and Clustering approach was used to make partitions of objects into equivalence classes. From the dataset by taking two conditional attributes at a time, partitioning of objects is done using K-means clustering algorithm [8]. Also using Indiscernibility relation partitioning of objects is done based on decision attribute only. Based on two sets of partition, connecting factor between two conditional attributes is computed and an attribute connecting set (*ACS*) containing all pair-wise connection of the attributes with respect to Decision attribute is obtained. Then attribute connection of *ACS* having connecting factor less than average connecting value are removed and an undirected weighted graph called Attribute Connecting Graph (*ACG*) is constructed based on the reduced set *ACS*. The *ACG*, therefore, represents the total connecting structure of the connecting set *ACS*. The connect factor between two attributes C_i and C_j is k means that both C_i and C_j together partitioned the objects which is (k*100) % similar to that obtained only by the decision attribute D. Now the sum of weights associated with the edges incident on a vertex is considered as the weighted degree of the vertex. The vertex with maximum weighted degree is removed with the adjustment of weighted degree of the vertices adjacent to it and stored the vertex in the reduct set. The process is repeated for modified graph until all the edges are removed from the graph, forming a compact set of attributes, called reduct.

The rest of the paper is organized as follows: Dimension Reduction using Rough Set Theory is demonstrated in section 2. Section 3 shows the experimental result of the proposed method and finally conclusion of the paper is stated in section 4.

2 Dimension Reduction Using Rough Set Theory

The proposed method computes equivalence classes of objects of the dataset using the concept of Indiscernibility relation and clustering technique by considering the decision attribute and conditional attributes respectively, which helps to measure the degree of similarity among the conditional attributes. Based on the similarity of attributes a weighted undirected graph is formed which finally generates multiple reducts.

2.1 Partitioning of Objects and Undirected Weighted Graph Construction

The objects are partitioned by two different ways:

(a) Based on decision attribute using indiscernibility relation

Let *DS = (U, A, C, D)* be a decision system where *U* is the finite, non-empty set of objects and $A = C \cup D$ such that *C* and *D* are set of condition and decision attributes respectively. Each attribute $a \in A$ can be defined as a function, described in (1).

$$f_a: U \rightarrow V_a, \forall a \in A \tag{1}$$

where, *Va*, the set of values of attribute *a*, is called the domain of *a*.

For any $P \subseteq A$, there exists a binary relation *IND (P)*, called indiscernibility relation and is defined in (2).

$$IND(P) = \{(x, y) \in U \times U \mid \forall a \in P, f_a(x) = f_a(y)\} \tag{2}$$

Where, $f_a(x)$ denotes the value of attribute *a* for object *x* in *U*. Obviously *IND* (*P*) is an equivalence relation which induces equivalence classes. The family of all equivalence classes of *IND* (*P*), i.e., partition determined by *P*, is denoted by *U*/*IND* (*P*) or simply *U*/*P* and an equivalence class of *U*/*P*, i.e., block of the partition *U*/*P*, containing *x* is denoted by *P*(*x*).

Thus for decision attribute D, the equivalence classes are *U*/*D* obtained by *IND (D)* using (2). Let $U/D = \mathrm{CL}^{\mathrm{D}} = \{CL_1^D, CL_2^D, \ldots\ldots\ldots, CL_k^D\}$.

(b) Applying clustering algorithm on the projections of dataset

Let $C = \{C_1, C_2, \ldots\ldots, C_n\}$ be the set of conditional attributes. Now projection on the dataset DS for two attributes C_i and C_j using equation (3) to obtain the projected dataset (*PDS*).

$$PDS = \prod_{C_i, C_j}(DS) \tag{3}$$

So *PDS* contains same number of objects as *DS*. Now the dataset *PDS* is clustered using K-means algorithm with *K* as the number of distinct values of decision attribute *D*. Let the classes obtain by C_i and C_j are $CL^{ij} = \{ CL_1^{ij}, CL_2^{ij}, \ldots\ldots\ldots CL_k^{ij} \}$ for all i, j = 1, 2, ……, n; i < j.

Attributes C_i and C_j are totally connected with respect to *D* if there are one to one correspondence among the elements of CL^D and CL^{ij}. But in real situation, it rarely occurs and so connecting power of attributes C_i and C_j is measured by introducing the connecting factor $\delta_f^{i,j}$ by equation (4) which measures the degree of connectivity of attributes between each other with respect to decision attribute.

$$\delta_f^{i,j} = \frac{1}{K} \sum_{CL_t^{ij} \in CL^{ij}} \frac{1}{CL_t^{ij}} \max_{\forall\, CL_P^D \in CL^D} \{CL_t^{ij} \cap CL_P^D\} \tag{4}$$

So $\delta_f^{i,j} = 1$; if C_i and C_j are totally connected with respect to *D*

< 1; otherwise

Thus for n conditional attributes, there are $\frac{n(n-1)}{2}$ pair wise connection of attributes with respect to D for the decision system DS in the form $\{C_i C_j \xrightarrow{\delta_f^{i,j}} D\}$. Let the attribute connecting set ACS = $\{C_i C_j \xrightarrow{\delta_f^{i,j}} D\}$ which consists of all possible pair wise connection of attributes. Now the average connecting factor δf is computed and the elements $C_i C_j \xrightarrow{\delta_f^{i,j}} D$ with $\delta_f^{i,j} < \delta f$ are discarded and the rest is considered as the modified attribute connecting set. Now from the modified ACS = $\{C_i C_j \xrightarrow{\delta_f^{i,j}} D\}$ a weighted undirected graph ACG = (V, E) is constructed as follows:

- For each element $C_i C_j \xrightarrow{\delta_f^{i,j}} D \in ACS$
  - (i) C_i and C_j are considered as vertices of the graph G i.e. $V = V \cup \{Ci\} \cup \{Cj\}$ Where $V = \{\emptyset\}$ initially.
  - (ii) An edge (C_i, C_j) is drawn with weight $\delta_f^{i,j}$ i.e. $E = E \cup \{(Ci, Cj)\}$ where $E = \{\emptyset\}$ initially. Thus E is a proper subset of $V \times V$.

This graph is called the attribute connecting graph *ACG* which represents how the attributes are strongly connected to represent a decision system.
The overall algorithm is described below:

```
Algorithm: Weighted_Undirected_Graph_Formation(DS,ACG)
Input: DS =(U,A,C,D)where C={C1,C2,…Cn}
Output: ACG =(V,E)
Begin
  CL^D= {CL_1^D, CL_2^D, ……., CL_k^D} using (2), where k=|D|
  δf=0 /*Average connection factor*/
  For i=1 to n {
     For j=i+1 to n {
        PDS=∏Ci,Cj(DS)
        CL^ij= {CL_1^ij, CL_2^ij, ……, CL_k^ij} by K-means algorithm on PDS
        Compute δ_f^{i,j} using equation (4)
        δf = δf + δ_f^{i,j}
     }
   }
   δf = 2δf / n(n-1)  /*Average Connecting Factor*/
   V = {∅}, E = {∅}
   For i=1 to n
      For j=i+1 to n
         If (δ_f^{i,j} >δf) {
            V = V ∪{Ci} ∪ {Cj};
            E = E ∪{(Ci, Cj) with weight δ_f^{i,j}}
         }
  Return ACG = (V, E)
End
```

2.2 Reduct Generation

The undirected weighted graph *ACG = (V, E)* has the weighted edges. The weight of an edge indicates the classification power of the attributes corresponding to the terminal nodes of the edge. Higher the weight of an edge indicates better the classification power of the combined attributes (nodes).Now a new term degree of connection of a node is defined as follows:

Definition: Degree of Connection of a Node v_i

Let *ACG = (V, E)* be an undirected weighted graph and $v_i \in V$ be a node. Then the degree of connection of v_i denoted by $dc(v_i)$ is defined as

$$dc(v_i) = \frac{1}{deg\ (vi)} \sum w_{ij} \ / \ (v_i, v_j) \in E \text{ and } w_{ij} \text{ is the weight of } (v_i, v_j) \quad (5)$$

Where, $deg\ (v_i)$ is the degree [9] of the vertex v_i.

Here higher the degree of connection implies the corresponding attribute is more important. So the reduct is formed using following steps:

- Initially the attribute associated with the node with highest degree of connection is considered as reduct.
- Then the vertex is removed from the attribute connecting graph (ACG). As the vertex is removed, so the 'degree of connection' of the vertices incident on the removed vertex are reduced by the weight associated with the corresponding edge.
- Thus the graph *ACG* is modified and the new attribute is added to the reduct and repeat the same process until all the edges are removed or the graphs become empty.

Here multiple reducts will be generated if more than one vertex has the highest degree of connection at some iteration. For example if after certain iteration the reduct set R = $\{C'_1, C'_2, \ldots, C'_r\}$ and for next iteration j-vertices v'_{r+1}, v'_{r+2},... v'_{r+j} has the highest degree of connection then after this iteration the reduct is R = $\{(C'_1, C'_2, \ldots C'_r, C'_{r+1}), (C'_1, C'_2, \ldots C'_r, C'_{r+2}), \ldots (C'_1, C'_2, \ldots C'_r, C'_{r+j})\}$. Thus, for a single reduct in previous iteration and j-vertices of highest degree of connection in *ACG*, *j*-number of reducts is obtained in the next iteration. This process provides us multiple numbers of reducts at the end of the iteration.

The detail algorithm for multiple reduct generation is given below:

```
Algorithm: Multiple_Reduct_Gen(ACG,RED)
Input: ACG = (V,E) with weight of edge(v_i,v_j)∈E as
wt(v_i,v_j)
Output: multiple reduct set RED
Begin
   RED=∅
   Repeat {
     For node v_i ∈ V {
        Compute degree of v_i as deg (v_i)
        Compute degree of connection of v_i using(5)
     }
```

```
      Let V' = {v'_{r+1}, v'_{r+2}, .... v'_{r+j}}, the vertex set of highest
             degree of connection
       RED = RED × {{C'_{r+1}}, { C'_{r+2}}, ....{C'_{r+j}}}
             /*here C'_{r+i} is the attribute corresponding to
               Vertex v'_{r+i} ∀i=1, 2, j*/
      /*Modify ACG*/
      For each vertex v'_{r+i} ∀i=1, 2, j {
        Let incident vertices are {v_1^i, v_2^i, ..... v_p^i}
        For each k=1 to p
          deg_con (v_k^i) = deg_con (v_k^i) - wt (v'_{r+i}, v_k^i)
        Remove v'_{r+i} from ACG = (V, E)
          /*So associated edges are removed*/
      }
   }Until (E= = ∅)
   Return (RED)
End
```

3 Experimental Results

The proposed method computes multiple reducts for real valued datasets collected from UCI machine learning repository [10]. The benchmark datasets like Wine, Heart and Glass contain 13, 13 and 10 conditional attributes respectively. The attributes are abbreviated by letters A, B, and so on, starting from their column position in the dataset. The proposed method generates reducts for the above mentioned datasets are listed in Table1 with their accuracies for various classifiers, based on 10-fold cross-validation using 'Weka' tool obtained at http://www.cs.waikato.ac.nz/~ml/.

Table 1. Reducts of datasets with accuracies given by various classifiers

Datasets	Reducts	Accuracy of various classifiers					
		Naïve Bayes	**SMO**	**KSTAR**	**Bagging**	**J48**	**PART**
Wine	ABCGJLM	95.50	97.19	96.62	96.06	95.38	93.82
	ABGJKLM	96.62	97.75	97.19	96.06	95.38	95.69
	ACFGJLM	96.94	96.06	94.94	96.50	96.38	94.38
	AFGJKLM	96.62	97.19	100.0	98.87	98.87	99.43
	Average Accuracy	96.42	97.04	97.18	96.87	96.50	95.83
Heart	ACDEFJM	55.94	57.29	54.80	57.34	52.79	55.19
	ACDEJKM	55.89	58.34	52.65	57.34	53.49	53.84
	ACDEFLM	57.59	58.74	53.79	56.99	58.04	56.89
	ACDEKLM	58.64	59.74	55.84	57.34	57.69	58.64

Table 1. (*continued*)

	ADEFHLM	55.94	56.29	52.95	55.94	56.99	54.44
	ADEHJKM	58.49	57.34	52.80	56.64	53.49	51.69
	ADEFHLM	55.59	59.04	51.60	59.44	56.99	52.04
	ADEHKLM	59.69	59.74	52.55	58.04	53.14	53.90
	Average Accuracy	57.22	58.31	53.37	57.38	55.33	54.57
Glass	AEGI	92.99	81.30	93.45	98.13	97.66	97.66
	ADEI	87.85	84.43	92.52	98.13	98.72	97.19
	Average Accuracy	90.42	82.86	92.98	98.13	98.19	97.42

Table 2. Accuracy Comparison of PRP, CFS and CON methods for real datasets

Classifier	**Wine (13)**			**Heart(13)**			**Glass(10)**		
	PRP	**CFS**	**CON**	**PRP**	**CFS**	**CON**	**PRP**	**CFS**	**CON**
	(7)	**(11)**	**(6)**	**(7)**	**(6)**	**(8)**	**(4)**	**(7)**	**(5)**
Naïve Bayes	96.42	97.11	95.12	57.22	55.44	56.74	90.42	91.12	94.13
SMO	97.04	96.04	95.36	58.31	59.18	58.13	82.36	78.50	79.77
KSTAR	97.18	98.04	94.94	53.37	53.44	56.39	92.98	92.52	93.39
Bagging	96.87	96.17	93.78	57.38	56.54	56.04	98.13	96.53	96.27
J48	96.50	96.26	95.32	55.33	54.09	53.84	98.19	94.72	95.26
PART	95.83	97.07	95.42	54.57	53.19	54.14	97.42	95.19	96.26
Average Accuracy	96.64	96.78	94.99	56.03	55.31	55.88	93.25	91.43	92.51

The proposed method (PRP) and well known dimensionality reduction methods, such as, Correlation-Based Feature Selection (CFS) method [11] and Consistency Subset Evaluator (CON) method [12] have been applied on the dataset for dimension reduction and the reduced datasets are classified on various classifiers. Original number of attributes, number of attributes after applying various reduction methods and

the accuracies (in %) of the real valued datasets are computed and listed in Table 2, which shows the efficiency of the proposed (PRP) method for datasets. Accuracies obtained by proposed method, listed in Table 2 are the average accuracies computed in Table 1. Table 2 shows that the average accuracies using proposed method are comparable with that of the existing methods and at the same time the method reduces 50% attributes on average, whereas CFS and CON reduce 34% and 47.5% attribute respectively.

4 Conclusion

In the paper, firstly, rough set theory is applied for partitioning of objects using discrete values of decision attribute, and then, clustering algorithm is used for partitioning of objects based on real valued conditional attributes. Thus, an integration of clustering technique and rough set approach is applied on real valued dataset for dimension reduction. Finally, an undirected weighed graph is constructed and transformed to a null graph to obtain the final reducts with time complexity $O(V^2)$, where |V| is the number of vertices of the graph. So, the proposed model can produce reduced feature set of a decision system effectively with better performance, as described in Table 2. Future enhancement to this work is to formation of classifiers from multiple reducts and finally ensemble them to generate an efficient classifier.

References

1. Della Pietra, S., Della Pietra, V., Lafferty, J.: Inducing features of random fields. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(4), 380–393 (1997)
2. Jensen, R., Shen, Q.: Fuzzy-Rough Attribute Reduction with Application to Web Categorization. Fuzzy Sets and Systems 141(3), 469–485 (2004)
3. Zhong, N., Skowron, A.: A Rough Set-Based Knowledge Discovery Process. Int. Journal of Applied Mathematics and Computer Science 11(3), 603–619 (2001); BIME Journal 05(1) (2005)
4. Alpaydin, E.: Introduction to Machine Learning. PHI (2010)
5. Pawlak, Z.: Rough set theory and its applications to data analysis. Cybernetics and Systems 29, 661–688 (1998)
6. Thangavel, K., Pethalakshmi, A.: Dimensionality reduction based on rough set theory: A Review. Journal of Applied Soft Computing 9(1), 1–12 (2009)
7. Das, A.K., et al.: Reduct Generation by Formation of Directed Minimal Spanning Tree using Rough Set Theory. In: INDIA 2012 (2012)
8. Hartigan, J.: Clustering Algorithms. Wiley, New York (1975)
9. Bang-Jensen, J., Gutin, G.: Digraphs: Theory, Algorithms and Applications. Springer, ISBN 1-85233-268-9
10. Murphy, P., Aha, W.: UCI repository of machine learning databases (1996), `http://www.ics.uci.edu/mlearn/MLRepository.html`
11. Hall, M.A.: Correlation-Based Feature Selection for Machine Learning, PhD thesis, Dept. of Computer Science, Univ. of Waikato, Hamilton, New Zealand (1998)
12. Liu, Setiono, R.: A Probabilistic Approach to Feature Selection: A Filter Solution. In: Proc. 13th Int'l Conf. Machine Learning, pp. 319–327 (1996)

Connectivity Differences between Human Operators of Swarms and Bandwidth Limitations*

Steven Nunnally[1], Phillip Walker[1], Michael Lewis[1], Andreas Kolling[2], Nilanjan Chakraborty[2], and Katia Sycara[2]

[1] University of Pittsburgh, Pittsburgh, PA 15213, USA
[2] Carnegie Mellon University, Pittsburgh, PA 15213, USA

Abstract. Human interaction with robot swarms (HSI) is a young field with very few user studies that explore operator behavior. All these studies assume perfect communication between the operator and the swarm. A key challenge in the use of swarm robotic systems in human supervised tasks is to understand human swarm interaction in the presence of limited communication bandwidth, which is a constraint arising in many practical scenarios. In this paper, we present results of human-subject experiments designed to study the effect of bandwidth limitations in human swarm interaction. We consider three levels of bandwidth availability in a swarm foraging task. The lowest bandwidth condition performs poorly, but the medium and high bandwidth condition both perform well. In the medium bandwidth condition, we display useful aggregated swarm information (like swarm centroid and spread) to compress the swarm state information. We also observe interesting operator behavior and adaptation of operators' swarm reaction.

1 Introduction

Swarm robotic systems consisting of many simple individual units with limited communication capabilities (e.g., limited radio power) may operate in a wide range of environments from indoor to outdoor underwater environments. Thus, swarm robots may operate under conditions where communication bandwidth is limited. Moreover, depending on environmental conditions, there could be differential capacity of inter-robot communication, or human to robot communication. Furthermore, as swarm systems are usually made of simple units, their localization capability may be poor. These limitations lead to two key challenges in human swarm interaction, namely, (a) the state information of the robots available to the human may not be accurate and (b) there may be a mismatch between the intent of the operator and the robots understanding of the human intent. Due to the localization error, any point in the reference frame of the operator will be erroneously interpreted by a robot as some other point. Thus, any effort by the

* This research has been sponsored in part by AFOSR FA955008-10356 and ONR Grant N0001409-10680.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 713–720, 2012.

operator to move the swarm towards a desired goal will be misinterpreted by a robot, thus creating an intent mismatch between the human and the robot. Current human-swarm interaction (HSI) literature [1–9] does not consider the above aspects of HSI and assumes perfect information transfer between the human and the robots. To close this gap, in this paper, we conduct controlled experiments to study the effect on performance of human-swarm *intent mismatch* and error in swarm state displayed to the operator in supervisory control of swarm robotic systems.

In our experimental scenario, a human operator has to guide a robotic swarm to find unknown targets in a given area. The area is divided into a finite number of regions (whose boundaries are unknown to the operator) and the operator has to match the targets found to the corresponding regions. The robots have a single behavior, namely achieving consensus on direction of motion [10, 11]. The humans can guide the swarm by giving them a point in the environment towards which the robots have to travel. The robots are assumed to have a localization error and the robot position and orientation is assumed to be a Gaussian distribution. In our experiment, each subject performs the mission under three conditions (that are presented to them in a random order), namely, (a) low swarm-to-human bandwidth and low intra-swarm bandwidth (low bandwidth condition), (b) low swarm-to-human bandwidth and high intra-swarm bandwidth (medium bandwidth condition) and (c) high bandwidth between swarm and operator (high bandwidth condition). For the low bandwidth condition, we assume that only one robot can send its state information at each time instant. For the medium bandwidth condition, the swarm members communicate among themselves to estimate their mean orientation and standard deviation of orientation, which is displayed on the screen. In the high bandwidth condition, all the robots send their position and orientation information to the operator. Our experimental results indicate that, as expected, there is a degradation of performance in the low bandwidth condition compared to the high bandwidth condition. However, in the medium bandwidth condition, in which operators had additional information about the standard deviation of orientation (and thereby whether the robots were moving in the direction the human desired), they performed as well as the high bandwidth condition.

2 Experimental Design

The study described below has three within subject conditions with twenty five participants. The user study explores three levels of bandwidth: low, medium, and high. Participants controlled thirty virtual robots in the robot simulator, Stage, to find targets distributed in an open environment [12]. The open environment, displayed on-screen, is divided into six heterogeneous regions, given to the participants on paper. Each region contains exactly one target and the goal is to match all six different colored targets to each region. The study used the Robot Operating System (ROS) as the controller for the robots in Stage [13].

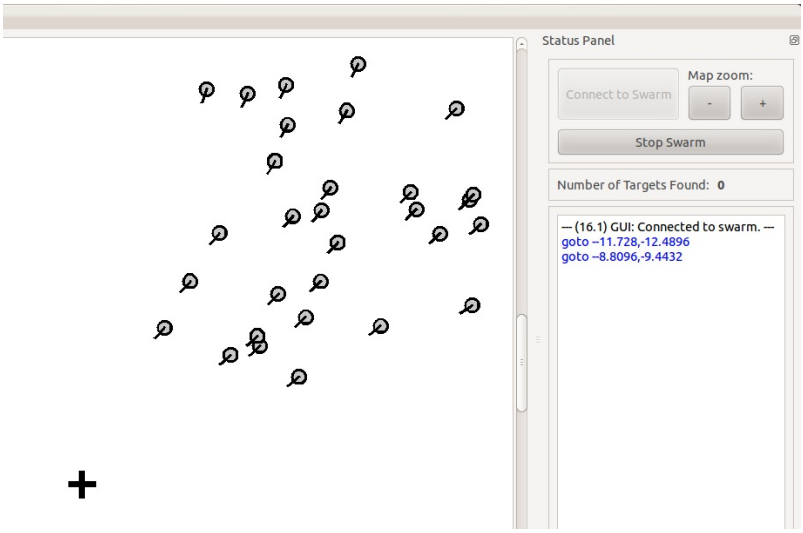


Fig. 1. The GUI used for the study. The left side shows the robots' estimated position and the right side shows the viewport via which the study participants issued commands. The + shows the endpoint (goal) of the "head-towards" user command.

The operator is given an interface, see figure 1, that displays the states of the swarm from a birds eye, orthographic view. A robots estimated position is displayed as a circle. A line pointing out the front of the robot indicates the current robot heading. The operator can zoom in and out and scroll to manipulate this viewport. The operator can issue two commands: "head-towards" and "stop". The "head-towards" command is given with a mouse click in the viewport. The "stop" command is issued as a button press.

2.1 Robot Algorithms

The study includes error models for location and orientation, as well as algorithms for the effect of commands on the robots, both as individuals and as a swarm. Location error is simulated with a smoothed, bounded, Gaussian model, with standard deviation of 1.0 meter and mean at the ground truth. The robot shifts in interpolated steps to the sampled error location, then the error is resampled. The location error should make discovery of targets near borders difficult so the participant must use many robots to diminish the error or explore all possible regions for other targets, eliminating regions from the list of possibilities since each regions has one target. When a "head-towards" command is received the robot samples a Gaussian model for its orientation, with standard deviation of $\pi/3$ radians and mean at the orientation vector at the "head-towards" point. The simulation of errors creates a more realistic scenario that considers the constraints of low-cost swarm robots.

The "stop" command is trivial, as all robots halt forward motion. The "head-towards" command starts with the orientation error described above. The robots then move forward at 0.5 m/s and start a standard consensus algorithm, receiving their neighbors' headings within a communication range of 4.0 meters. Each robot averages their own and their neighbors headings, and adjust their heading to match this average. Robots may lose connectivity before consensus is reached. Due to the nature of Gaussian noise models, the consensed heading will be in general erroneous. It will, however, be closer to the requested "head-towards" point if a greater number of robots are connected. At any point the operator can decide whether the consensus direction is not acceptable and issue a new command, repeating the process.

2.2 Procedure

25 paid participants from the University of Pittsburgh participated in the study. The participants were familiarized with the task and the robot algorithms, and were shown how to use the GUI to issue commands. Every time a robot member of the swarm was close to a target, the robot icon (a circle) on the display would turn the corresponding color, visible to the participants. At the end of each session, the participants were asked to match the color target to each region. Participants were urged to only record a non-answer if they never saw that color target, but if seen the participants were instructed to guess the region. The importance of maintaining one connected swarm of robots was stressed for maintaining lower consensed orientation errors. Participants were told that new "head-towards" commands issued before consensus was reached could adversely affect the connectivity of the swarm. Participants were then given ten minutes to adjust to the interface and train for the task.

The study had three experimental conditions with trials lasting ten minutes each. Participants were given the following conditions in a random order: low bandwidth, medium bandwidth, and high bandwidth. In the low bandwidth condition the robot owning the unique token could update its information on the interface. At each update step the token was transmitted to a random neighboring robot, causing robots with more neighbors to update with higher probability. The interface stored the previous 21 updates. In those 21 updates, one robot could update many times and some would not update at all. Reaching consensus could be difficult to observe since it takes a few updates to see robots moving in the same direction.

In the medium bandwidth condition the swarm aggregated information on location and heading using its local communication network. The standard deviation of the heading was displayed as a proxy variable for the swarm's heading consensus. The standard deviation of the locations was used to create an ellipsoid around the average location to show the general shape and density of the swarm. Up to four robots could update every half second, which allowed smaller groups that break communication with the main swarm to update their information. Sensed targets were displayed as a colored percentage beside the aggregate display of the number of robots in that group that could sense that color target.

Finally, the third condition was the high bandwidth condition, where all robots updated their position (and the updated position was shown on the screen) every half second. The participant determined when consensus had been reached by observing the movement of the individual robots in the swarm.

3 Results

The data analysis showed differences in the connectivity of the robot swarm across participants and conditions. Connectivity is measured using the second eigenvalue of the communication graph's Laplacian matrix. Since the number of zero eigenvalues shows the number of connected components in a graph and single robots break away early and often, the second eigenvalue is measured on the largest connected component of the graph as long as that group contains over half of the robots.

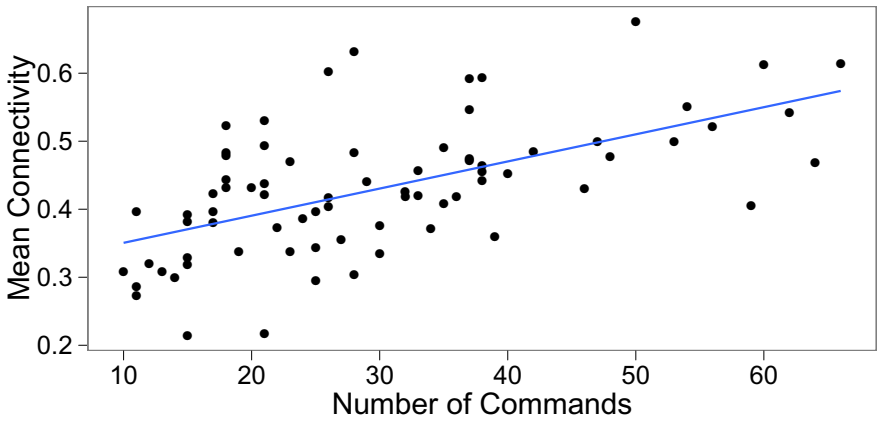


Fig. 2. Each point represents a trial in the experiment. This compares the connectivity throughout each trial to the number of commands sent during the trial. More commands create a more highly connected swarm.

In the instructions participants were warned that every command gave the swarm a chance to break connectivity. Yet, figure 2 shows that connectivity increased with the frequency of commands ($p<0.001$). This increase in connectivity stems from a difference in participant behavior, see figure 3. Participants who placed reference points closer to swarm enhancing connectivity also issued commands more frequently ($p<0.001$) keeping the swarm moving. Such a special command, close to the centroid of the swarm, rather than reducing connectivity, can actually improve it. This effective "small and frequent" strategy emerged in a subset of participants despite instructions encouraging infrequent commands.

Between conditions, the low bandwidth condition has a lower average connectivity compared to the other conditions ($p<0.01$ of the medium condition and

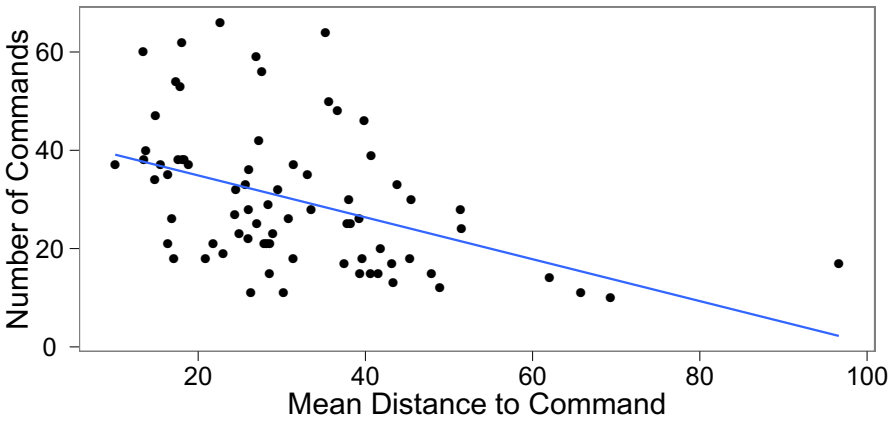


Fig. 3. Each point represents a trial in the experiment. The figure compares the average distance from the swarm's centroid to each command throughout each trial to the number of commands sent during the trial. More commands have less of a distance to each command explaining the correlation between number of commands and connectivity.

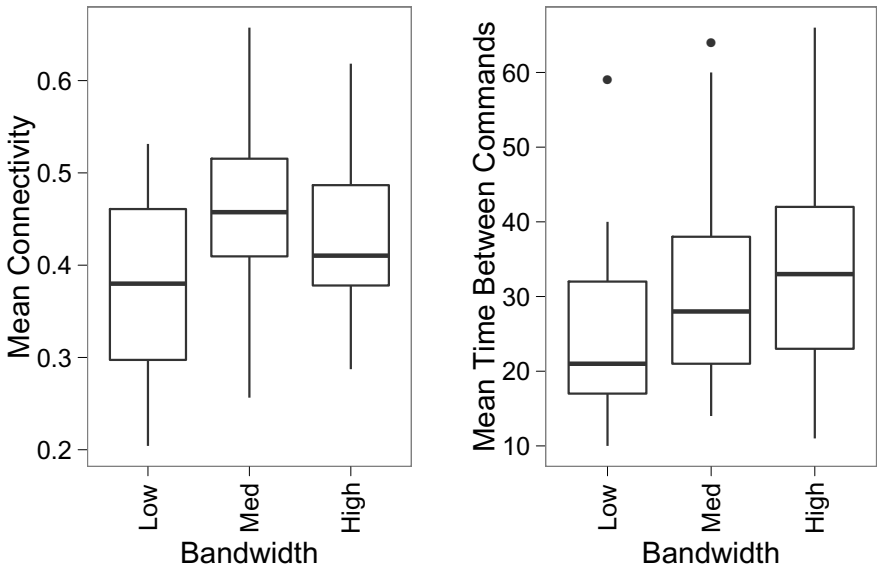


Fig. 4. These graphs display differences of the low condition compared to the other two with regards to connectivity and frequency of commands. Connectivity is lowest in the low bandwidth condition (left). Number of commands over the ten minute trials is lowest in the low bandwidth condition (right). There are 25 participants per condition in each graph.

$p<0.05$ of the high condition), see figure 4. The differences between the medium and high bandwidth are not significant. The low bandwidth condition creates an environment where the largest connected component either contains fewer than half of the robots or is sparsely connected. Participants also give fewer commands in the low bandwidth condition ($p<0.03$ of both medium and high condition), see figure 4. This explains the lower connectivity of the low bandwidth condition as shown in earlier results. The most probable cause of the fewer commands in the low condition case is the time participants must wait for enough robots to update in order to gain knowledge of the state of the swarm before making decisions and giving new commands.

4 Conclusions and Future Work

Despite the large errors in localization for our swarm robots, we have shown that operators can successfully interact with the swarm and its consensus algorithms. The observed operator behavior revealed a variety of interaction strategies, with operators adapting their behavior to the swarm, even in the short time the participant used the system. This was especially interesting for operators that issued frequent commands and then discovered to use some of these commands to improve connectivity, mitigating the otherwise negative effect. This suggests that human operators can, in fact, learn and adapt to swarm dynamics and adapt their instructions to improve the swarm's behavior and state.

In addition, we investigated the effects of bandwidth limitations on the interaction. We have shown that a medium bandwidth condition, which only shows aggregated state information from the swarm, is sufficient for a successful interaction in our foraging scenario. Additional information available in the high bandwidth condition about every individual robot did not improve the interaction with the operator. This result encourages an emphasis on aggregate statistics when considering operator interactions with a swarm. In contrast, the low bandwidth condition not only had a low spatial resolution but also a low temporal resolution. This affected the interaction negatively and lead to worse performance. In future work, we plan to further explore how different environmental constraints affect the performance in human supervisory control of swarms.

References

1. Cummings, M.: Human supervisory control of swarming networks. In: 2nd Annual Swarming: Autonomous Intelligent Networked Systems Conference (2004)
2. Klarer, P.: Flocking small smart machines: An experiment in cooperative, multi-machine control. Technical report, Sandia National Labs., Albuquerque, NM, United States (1998)
3. Kira, Z., Potter, M.: Exerting human control over decentralized robot swarms. In: 4th International Conference on Autonomous Robots and Agents, ICARA 2009, pp. 566–571. IEEE (2009)

4. Bashyal, S., Venayagamoorthy, G.: Human swarm interaction for radiation source search and localization. In: IEEE Swarm Intelligence Symposium, SIS 2008, pp. 1–8. IEEE (2008)
5. Fields, M., Haas, E., Hill, S., Stachowiak, C., Barnes, L.: Effective robot team control methodologies for battlefield applications. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009, pp. 5862–5867. IEEE (2009)
6. Naghsh, A., Gancet, J., Tanoto, A., Roast, C.: Analysis and design of human-robot swarm interaction in firefighting. In: The 17th IEEE International Symposium on Robot and Human Interactive Communication, RO-MAN 2008, pp. 255–260. IEEE (2008)
7. Goodrich, M., Pendleton, B., Sujit, P., Pinto, J.: Toward human interaction with bio-inspired robot teams. In: 2011 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 2859–2864. IEEE (2011)
8. Kolling, A., Nunnally, S., Lewis, M.: Towards human control of robot swarms. In: Proceedings of the 7th International Conference on Human-robot Interaction. ACM (2012)
9. Coppin, G., Legras, F.: Autonomy spectrum and performance perception issues in swarm supervisory control. Proceedings of the IEEE (99), 590–603 (2012)
10. Reynolds, C.: Flocks, herds and schools: A distributed behavioral model. In: ACM SIGGRAPH Computer Graphics, vol. 21, pp. 25–34. ACM (1987)
11. Couzin, I., Krause, J., James, R., Ruxton, G., Franks, N.: Collective memory and spatial sorting in animal groups. Journal of Theoretical Biology 218(1), 1–11 (2002)
12. Gerkey, B.P., Vaughan, R.T., Howard, A.: The player/stage project: Tools for multi-robot and distributed sensor systems. In: International Conference on Advanced Robotics, Coimbra, Portugal, pp. 317–323 (2003)
13. Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., Ng, A.: Ros: an -open source robot operating system. In: International Conference on Robotics and Automation, Kobe, Japan (2009)

Analysis of Emergent Selection Pressure in Evolutionary Algorithm and Machine Learner Offspring Filtering Hybrids

Mark Coletti[1] and Guido Cervone[2]

[1] Computer Science Department, Evolutionary Computation Laboratory
[2] Department of Geography and Geoinformation Science
George Mason University
4400 University Drive, Fairfax, Virginia, United States of America
{mcoletti,gcervone}@gmu.edu

Abstract. When evolutionary algorithms are applied to problems with computationally intensive fitness functions a limited budget of evaluations is usually available. For these types of problems minimizing the number of function evaluations becomes paramount, which can be achieved by using smaller population sizes and limiting the number of generations per run. Unfortunately this leads to a limited sampling of the problem space, which means finding adequate solutions is less likely. Evolutionary algorithms (EA) can be augmented with machine learners (ML) to more effectively explore the problem space. However, a "well-tuned" evolutionary algorithm strikes a balance between its constituent operators. Failure to do so could mean implementations that prematurely converge to inferior solutions or to not converge at all. One aspect of such "tuning" is the use of a proper selection pressure. Introducing a machine learner into an EA/ML hybrid introduces a new form of "emergent" selection pressure for which practitioners may need to compensate. This research shows two implementations of EA/ML hybrids that filter out inferior offspring based on knowledge inferred from better individuals have different emergent selection pressure characteristics.

Keywords: evolutionary computation, machine learning, hybrid.

1 Introduction

There exist several problems that can be solved using evolutionary algorithms that require very lengthy fitness function evaluation, like those that require running a complex simulation [1–3]. For this class of problems minimizing the number of these computationally burdensome function evaluations is paramount. This typically entails smaller population sizes and a limited, fixed budget for evaluations. Unfortunately these smaller populations and fitness evaluation budgets mean that it is more difficult to find viable solutions because the solution space will not be adequately explored. However, one means of compensating for these smaller populations and fixed evaluation budgets is to augment an evolutionary algorithm with a machine learner.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 721–728, 2012.

A machine learner can act as a filter for inferior offspring. For example, rules that characterize the worst individuals can be used to ensure all new individuals are generated so that they do not satisfy any of the above rules, which essentially "walls off" areas of the search space deemed unviable [4]. A similar approach was used by LEM(KNN) that uses K-Nearest-Neighbors clustering of the "best" and "worst" individuals, and then iteratively generates offspring until offspring closest to the "best" cluster were created [5]. An iterative algorithm that uses learning was introduced to identify ensemble members for atmospheric simulations [3]; it is similar to LEM(KNN) except that all individuals are generated all at once in a batch, and an additional domain knowledge filter was used to further ensure the viability of individuals.

However, a "well-tuned" evolutionary algorithm strikes a balance between all its component operators [6]. For example, if the selection pressure is too high with correspondingly weak perturbation operators, then the algorithm risks prematurely converging on inferior solutions. Conversely, if the selection pressure is too weak with wildly erratic perturbation operators, then the algorithm may not converge on any solution. So, it is important for practitioners to be aware of striking the best balance between all operators; and, that blending a machine learner with an evolutionary algorithm may introduce new forces that will have to be commensurately counterbalanced.

One intuition is that, in a sense, adding a machine learner may increase selection pressure, though hopefully in an intelligent manner. Indeed, LEM has a fairly strong "emergent" selection pressure independent of any selection pressure induced by more traditional EA operators and so practitioners need to to be mindful of this when implementing LEM [7].

What we address in this paper is whether similar "emergent" selection pressure exists with the EA/ML hybrid that uses the learner as a filter for viable offspring. To this end, we explore two EA/ML offspring filter implementations: one that iteratively generates offspring one at a time akin to LEM(KNN), and the other that generates offspring and then filters them in a single batch.

2 Selection Pressure of "Single" Mode Offspring Filter

The first EA/ML offspring filter we examine is similar to LEM(KNN) in that it, too, learns from the best and worst individuals and then iteratively creates offspring until offspring are generated closest to the "fittest". However, since we are interested in determining if there is some form of "emergent" selection pressure, we differ our implementation to minimize sources of selection pressure from traditional EA selection operators; and to that end our implementation uses deterministic parent selection and non-overlapping generations. That is, each parent creates one offspring, and then all the offspring replace the parents for the next generation. The notion here is that any evidence of selection pressure will be due to the influence of the machine learner since fitness bias due to parent or survival selection is absent.

Algorithm 1 shows how this machine learner-based filtered method "single mode" was implemented. First the existing parents of the ith generation, P_i, are

Algorithm 1. "Single" mode machine learner offspring reproduction

```
P_i ← initial()                                          ▷ Create initial population
repeat
    sort(P_i)                                            ▷ Sort parents of ith generation by fitness
    learn(best(P_i), worst(P_i))                         ▷ Learn from best and worst individuals
    for currentParent = 0 → μ − 1 do
        tries ← 10000          ▷ Number of attempts to create viable child before giving up
        repeat
            I ← mutate(clone(P_i[currentParent]), σ, ρ)
            class ← classify(I)
            tries ← tries − 1
        until class = "best" ∨ tries = 0
        P_{i+1}[currentParent] ← I
        currentParent ← currentParent + 1
    end for
until halt() = true
```

sorted by fitness and then the best and worst subpopulations are given to the machine learner, which then infers something about these "best" and "least" fit individuals based on their respective genomes. Then, for each new parent we wish to create, from a total of μ parents, we will repeatedly try to create a new offspring, I, and assigned that to be a new parent in the next set of parents, P_{i+1}. This is done by cloning the current parent, $P_i[currentParent]$, then mutating it with the scale factor, σ, and probability, ρ. The newly created individual, I, is then classified by the machine learner based on the "best" and "worst" parents from which it previously learned. If the ML deems the newly created individual as being "best", then it is assigned to be a new parent; otherwise the clone and mutate process is repeated until either a viable individual is created or the number of tries is exhausted. In the latter case, the individual is assigned as a new parent even though it is not classified as "best".

Unlike LEM(KNN) we did not use K-Nearest-Neighbors clustering, but instead chose two different machine learners, JRip and J48, which are Weka implementations of the RIPPER and C4.5 machine learners [8–10]. We used ECJ in our implementation and some of its ready-made "ECSuite" fitness functions, such as the Rastrigin, Rosenbrock, and Griewangk functions[11]. Table 1 enumerates the run-time parameters for the experiments used for this paper.

Figure 1 shows sample results obtained using the "single" filtering mode for the Griewangk (left) and Median (right) functions. The figure shows boxplots summarizing the average of 30 runs in terms of Fitness (vertical axis) and number of generations (horizontal axis). Although only two sets of runs are shown for brevity, all results are similar, either showing no improvement (Figure 1 left) or slow improvement (Figure 1 right). There is also no significant difference with respect to which machine learning program is used nor if ML rule pruning is applied or not.

Table 1. Run-time parameters for experiments

Parameter	Value
Parents	100
Children	100
Parent selection	deterministic
Domain range	[-4,4]
Genome size (variables)	10
Mutation	Gaussian
Mutation rate	0.6
Mutation scale	0.3
Generations	100
Runs	30
Error function	spheroid, griewangk median, noisy-quartic rastrigin, rosenbrock, step
ML Classifier	J48 JRip

Discussion

The results show that the emergent selection pressure for the "single" mode of these machine learner filtered offspring systems is very weak, particularly for difficult problems. This is because that even though knowledge inferred from the "best" individuals is used to pass over inferior generated offspring, all the parents — including inferior ones — generate offspring. And, if an inferior individual is too far from the optima it may find it difficult to generate viable offspring. Indeed, it may have to generate numerous candidates before meeting success, if at all.

Figure 2 shows exactly this by depicting the number of offspring accepted for the Rastrigin function in "single" mode, and the number of aborted offspring. Recall that the brood size is one so each parent ultimately contributes a single offspring. Given that we need 100 new parents, it can be shown that many tries may need to be attempted before a viable offspring is generated; and, even then, there is no guarantee of success, and the inferior child becomes a parent anyway in the next generation. That is, if all the parents were able to successfully contribute offspring to the next generation, then the "NumPassed" would always be 100. That "NumPassed" is less than a 100 in any of the portrayed generations indicates that some parents were unable to create offspring that passed the machine learner's test, but became a parent anyway because the number of tries was exhausted.

3 Selection Pressure of "Batch" Mode Offspring Filter

Algorithm 2 delineates how the "batch mode" machine learner filter method works, which is very similar to the implementation used used to identify ensemble members in an atmospheric simulation [3]. As with the "single mode" described

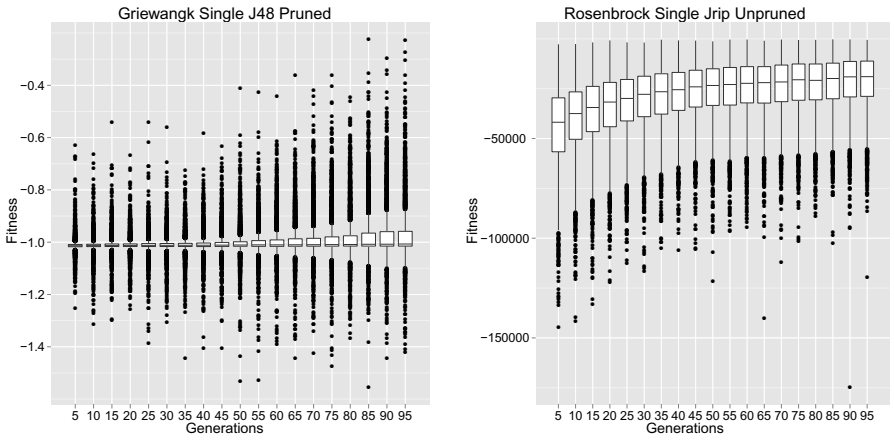


Fig. 1. Single mode runs on Griewangk and Rosenbrock fitness landscapes

in Algorithm 1, the parent population is sorted and then the machine learner learns from the best and worst parent subpopulations. However, unlike the "single mode," each parent will create a single offspring by mutating a clone of itself; then *all* these offspring are placed into a new population, P_c. Then only those offspring that are classified as "best" are copied over the parents in the original population from the "bottom up." That is, given that the parent population is sorted, the worst parents will be over-written by children deemed "best" by the machine learner.

Algorithm 2. "Batch" mode machine learner offspring reproduction

$P_i \leftarrow initial()$ ▷ Create initial population
repeat
  $sort(P_i)$ ▷ Sort parents of ith generation by fitness
  $learn(best(P_i), worst(P_i))$ ▷ Learn from best and worst individuals
  $viableChildren \leftarrow 0$
  for $currentChild = 0 \rightarrow \lambda - 1$ **do** ▷ Create λ children in one batch
    $I \leftarrow mutate(clone(P_i[currentChild]), \sigma, \rho)$
    $class \leftarrow classify(I)$
    if $class =$ "$best$" **then**
      $P_c[viableChildren] \leftarrow I$
      $viableChildren \leftarrow viableChildren + 1$
    end if
    $currentChild \leftarrow currentChild + 1$
  end for ▷ Over-write parents with "best" offspring from bottom up
  $currentChild \leftarrow 0$
  for $currentParent = \mu - 1 \rightarrow \mu - 1 - viableChildren$ **do**
    $P_i[currentParent] \leftarrow P_c[currentChild]$
    $currentParent \leftarrow currentParent - 1$
    $currentChild \leftarrow currentChild + 1$
  end for
  $P_{i+1} \leftarrow P_i$ ▷ Next generation contains mix of better parents and children
until $halt() = true$

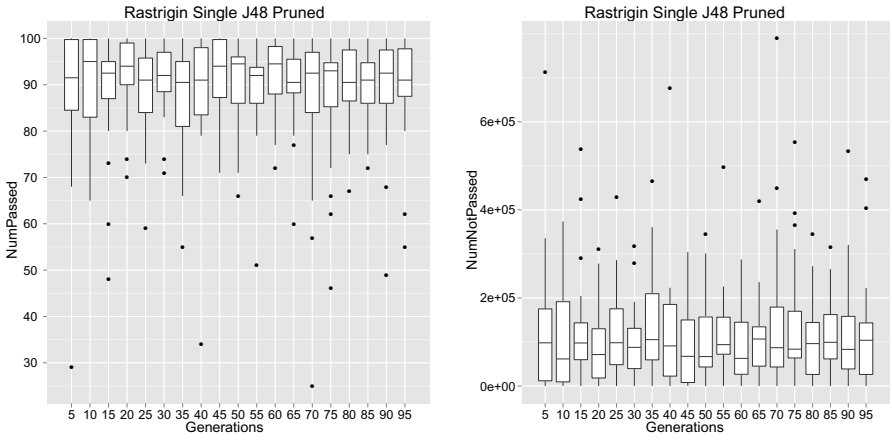


Fig. 2. Aggregate offspring accepted and rejected for Rastrigin function "single" mode

Figure 3 shows results using the "batch" filtering mode for just the Griewangk (left) and Median (right) landscapes; the rest of the plots have been elided for brevity but exhibit identical characteristics. That is, unlike Figure 1, the populations quickly converge to solutions, solutions that are generally far better than those found using "single" mode indicating a comparatively stronger selection pressure.

Discussion

By contrast, we have shown that "batch" version of the filtered offspring system has much higher selection pressure than the "single" version. There are two places where the "batch" implementation introduces selection pressure. First, even though *all* the parents produce exactly one offspring as in the "single" mode, the machine learner is used to select which ones survive to the next generation. Second, a form of truncation survival selection manifests when these surviving offspring replace the worst set of parents keeping the better ones. However, unlike the typical truncation selection operator that always keeps the n best, this form of truncation varies in the number of of the best parents kept based on the number of offspring that passed the ML's muster. For example, Figure 4 shows the number of offspring that passed and did not pass for the Rastrigin fitness landscape for "batch" mode; all the children that passed over-wrote inferior parents leaving the remaining parents to survive for another generation.

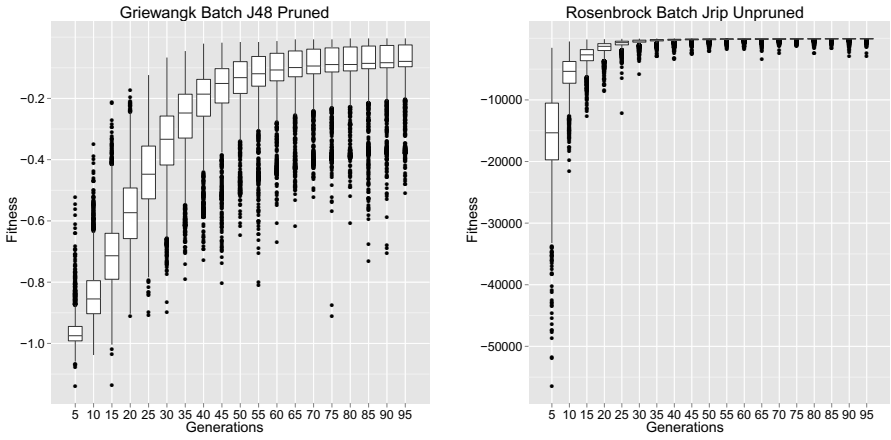


Fig. 3. Batch mode runs on Griewangk and Rosenbrock fitness landscapes

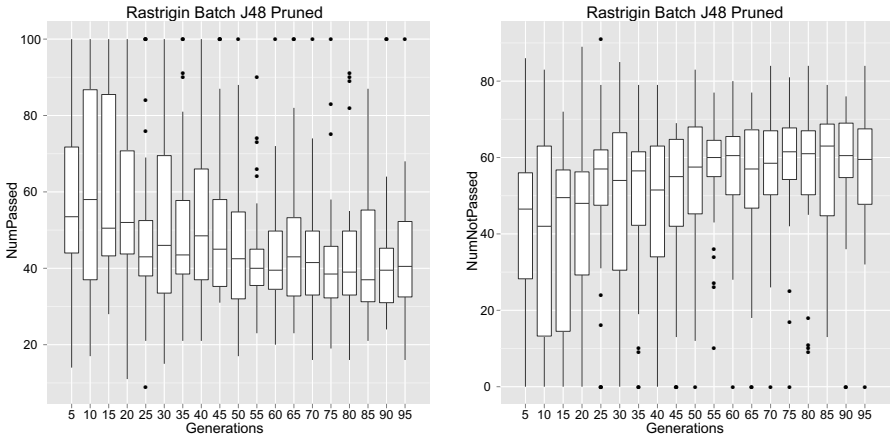


Fig. 4. Aggregate offspring accepted and rejected for Rastrigin function "batch" mode

4 Conclusions and Future Work

We have demonstrated that the "batch" EA/ML offspring filter implementation improves performance over the "single" mode by adding intrinsic selection pressure derived from using the ML to select fitter children and keeping the best parents. However, the "single" mode will probably be improved by adding a more traditional selection operator to augment its relatively weak selection pressure. For example, some form of fitness proportional parent selection may go a long ways to improving overall performance.

This paper only used the machine learners JRip and J48. Though their relative performance characteristics were similar, a more detailed analysis is warranted with other methods with a different inductive bias.

Previous work [7] has shown that training set size and keeping prior knowledge in a similar EA/ML hybrid can have an influence on performance. Future research will investigate their role with these EA/ML offspring filtering systems.

References

1. Cervone, G., Franzese, P.: Non-Darwinian Evolution for the Source Detection of Atmospheric Releases. Atmospheric Environment 45(26), 4497–4506 (2011)
2. Cervone, G., Franzese, P., Keesee, A.P.K.: Algorithm quasi-optimal (AQ) learning. Wiley Interdisciplinary Reviews: Computational Statistics 2(2), 218–236 (2010)
3. Lattner, A.D., Cervone, G.: Ensemble modeling of transport and dispersion simulations guided by machine learning hypotheses generation. Computers & Geosciences (2012)
4. Sebag, M., Schoenauer, M., Ravise, C.: Inductive Learning of Mutation Step-Size in Evolutionary Parameter Optimization. In: Angeline, P.J., McDonnell, J.R., Reynolds, R.G., Eberhart, R. (eds.) EP 1997. LNCS, vol. 1213, pp. 247–261. Springer, Heidelberg (1997)
5. Sheri, G., Corne, D.W.: The simplest evolution/learning hybrid: LEM with KNN. In: IEEE Congress on Evolutionary Computation, CEC 2008, pp. 3244–3251 (June 2008); IEEE World Congress on Computational Intelligence
6. De Jong, K.: Evolutionary Computation: A Unified Approach. The MIT Press, Cambridge (2006)
7. Coletti, M.: The Effects of Training Set Size and Keeping Rules on the Emergent Selection Pressure of Learnable Evolution Model. In: Genetic and Evolutionary Computing Conference Proceedings (2012)
8. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update. SIGKDD Explorations 11(1) (2009); Weka Machine Learning Project
9. Cohen, W.W.: Fast Effective Rule Induction. In: Twelfth International Conference on Machine Learning, pp. 115–123 (1995)
10. Quinlan, J.R.: C4.5, Programs for Machine Learning. Morgan Kaufmann Publishers (1993)
11. Luke, S., Panait, L., Balan, G., Paus, S., Skolicki, Z., Popovici, E., Sullivan, K., Harrison, J., Bassett, J., Hubley, R., O'Beirne, J.: ECJ, http://cs.gmu.edu/~eclab/projects/ecj/

OpenCL Implementations of a Genetic Algorithm for Feature Selection in Periocular Biometric Recognition

Paulo Fazendeiro, Chandrashekhar Padole, Pedro Sequeira, and Paula Prata

Instituto de Telecomunicações (IT) Portugal
University of Beira Interior, Department of Informatics, Portugal
fazendeiro@ubi.pt, chandupadole@yahoo.com,
morps@live.com.pt, pprata@di.ubi.pt

Abstract. This paper explores OpenCL implementations of a genetic algorithm used to optimize the features vector in periocular biometric recognition. Using a multi core platform the algorithm is tested for CPU and GPU, exploring different parallelization levels for each operator of the genetic algorithm. The results show that using the GPU platform it is possible to accelerate the algorithm by several orders of magnitude, with a recognition rate similar to the one obtained in the sequential version. The results also show that it is possible to use only a small portion of the features without any degradation of the classifier's recognition rate.

Keywords: GPGPU, Parallel Genetic Algorithms, Biometric Recognition, Periocular Recognition, OpenCL, Data Parallelism.

1 Introduction

Biometrics has been widely investigated and used effectively in several applications: authentication in highly restricted areas, attendance record in office premises, citizenship identification and verification and forensics. Different biometric traits such as face, iris, fingerprint and gait, do provide the flexibility to choose one or combine more than one modality for recognition, depending on the availability and feasibility associated with the objectives of application. Periocular biometrics refers to the recognition using information from the facial region in the immediate vicinity of the eye [1]. Despite of utilizing the iris region as a part, periocular biometrics can be preferred over the iris recognition for several reasons reported in existing literature.

In this paper, we assess the effectiveness of a genetic algorithm (GA) to select a set of feature elements present in the feature vector of periocular region. The genetic algorithm was applied to optimize the length of feature vector so that recognition performance can be maximized and computation required for classification can be brought down to optimum level. We experimented with genetic algorithm for periocular recognition problem using periocular dataset, UBIPr [2] with various "covariates" present in it. The feature type used for the optimization problem was Linear Binary Pattern (LBP) [3]. LBP was initially used for measuring the local image contrast [3], it

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 729–737, 2012.

has since been applied in several pattern classification problems [4-5]. We carried out the process of optimization with a small set of classes (10 classes) and tested the optimized solution with a larger number, 100 classes, in order to examine the robustness of the optimized feature vector. Moreover we present and analyze different parallel implementations of our training algorithm.

2 Evolutionary Algorithms Design for Periocular Recognition

Evolutionary algorithms (EAs) are adaptive robust methods, biologically inspired, that are widely applicable for search, optimization, and learning problems [6-9]. The ability to work in each generation, with a whole set of potential solutions, with each one being optimal in some sense, allied with the capability to explore a multimodal environment makes them well suited for multi-objective optimization problems [6-7].

These computationally intensive population-based search methods present heavy time requirements, hence reducing their applicability especially in what concerns real time applications. The potential for acceleration of population-based, stochastic function optimizers using Graphics Processing Units (GPUs) has been already verified in a number of recent works (e.g. [10-14] just to mention a few) over a representative set of benchmark problems. Genetic algorithms (GAs) [15] form a particular kind of EAs, inspired by the Darwinian principle of the survival of the fittest, widely used in the actuality.

The aim of our work is to use a GA to synthesize a k-nearest neighbor (k-NN) algorithm using a minimum set of features extracted from images of the periocular region. For each image it was extracted a set of 1536 LBP features per each eye resulting in a vector of 3072 features. Following the methodology presented in [16] the distance between each instance of a biometric feature set is defined as the sum of the differences of the corresponding feature set values using Manhattan Distance pairwise multiplied by a feature mask. If the feature mask is set to 0 to a given feature, then that particular feature does not affect the distance. Likewise real values between 0.0 and 1.0 determine the relative strength of a particular feature.

The proposed steady state genetic algorithm performs a combination of feature selection altogether with weighting of the selected features. The real-valued chromosome encoding [17] results in 3072 real values representing the feature mask.

2.1 Fitness Assessment

The first phase of the GA involves the quantitative evaluation of each chromosome in the population. This value determines the probability that an individual has to be selected and to carry its genetic material for the next phase (to the more fit individuals are concealed more reproductive chances).

We considered a gallery feature set with 10 classes. Per each class, the database has 15 samples (probe feature set) deduced from images obtained in 3 distinct poses (frontal, left and right) at 5 distinct ranges. The gallery feature is composed of 10 elements resulting from the average of six samples per class (3 poses at 2 different ranges).

Since we are aiming at a high recognition rate altogether with a confined number of features the fitness function of each chromosome is given by adding to the recognition rate percentage (RR) a penalty term equal to the percentage of features (PF) considered in the feature mask (the ones with weight greater than zero).

$$fitness = RR - PF \tag{1}$$

In order to obtain the RR for a given feature mask (chromosome) the biometric distance of each one of the 150 samples towards the 10 elements of the gallery feature set is computed. The predicted class of each sample is equal to the one of the closest (having minimum biometric distance) class of the gallery. If the predicted class matches the real class, the number of the correctly identified samples of the chromosome is increased by one. Finally RR is computed as the ratio of the correctly identified samples and the cardinality of the probe set (150). The PF value is simply given by the ratio between the number of genes with weight greater than zero and the maximum number of features (3072).

2.2 Parallelization of Genetic Operators

The parallel design of a GA implementation must consider the different granularity levels subsumed in all the different steps. The fitness evaluation is eminently a one-dimensional operation (in the sense that usually each chromosome, taken as a whole, can be evaluated in parallel) whereas the crossover and mutation operators are, in the general case, 2D operators allowing the parallel treatment of not only each chromosome but also each gene inside it. However for the particular problem of feature selection/weighting a special remark should be made regarding the fitness evaluation. In this particular case it is possible to envision different granularity levels: one-dimensional with each chromosome evaluated in parallel, two-dimensional with the parallel treatment of each chromosome and each class inside it or even three-dimensional with the parallel treatment of each sample of a given class of a given chromosome.

3 Experiments and Results

OpenCL is a framework for parallel programming that has the advantage of being platform-independent. An OpenCL application is a program executing in the host that can launch functions (kernels) to be executed on OpenCL devices [18]. When a kernel is submitted for execution, it is defined an N-dimensional index space called NDrange, with N=1, 2 or 3. A high number of kernel instances (threads) can be executed simultaneously, each one for a point in the index space. Each kernel instance is called a work-item and is identified by the corresponding point in the index space. The same code can be executed over different data items following a Single Instruction Multiple Data model. Work-items can be organized into work-groups. Several work-groups can be executed in a Single Program Multiple Data model.

In this work the host machine was an Intel i7 860 processor (4 cores, 8 threads), at 2.8 GHz, with 8GB of RAM, running the 64 bits Windows 7 operating system. The GPU was an AMD Radeon HD 6990 with 2GB of global memory, and 1536 cores at 830 MHz. Host and GPU were programmed with OpenCL version 1.1.

The parameters of the GA were kept constant in the presented experiments. The selection relied on the binary tournament operator. All the selected individuals were subjected to the blend crossover (BLX-alpha) operator chosen due to its suitability to the real-valued chromosome encoding that was used [17]. The probability of mutation was equal to 0.05. Furthermore, to prevent good solutions from disappearing during the evolutionary process it was used an elitist approach maintaining the first and second best solutions. The choice of all the above numeric values of the parameters was based on an extensive set of preliminary experiments, with the speed of convergence rate being the main guiding mechanism. The population's size was 128 with a fixed chromosome length of 3072. Results were obtained as the average of 20 runs with 20 iterations each. In the reported experiments the training was conducted on a database with ten classes (15 samples per class) whereas an extended database with 100 classes (the previous ten plus ninety new ones) was used for testing of the twenty best synthesized solutions.

3.1 CPU Implementation

Prior to build a parallel version of the algorithm, the sequential execution was characterized. This was done, calculating the execution time percentages in CPU for each main step of the algorithm which is briefly outlined in Algorithm 1. For a direct comparison with the parallel version, the steps that will be parallelized are already implemented as kernels in the sequential version (they are marked with (k) in Algorithm 1.)

The sequential version is implemented in OpenCL, running sequentially in just one work group, with just one work item.

Algorithm 1. Optimization algorithm

1 – Initialize the first population
2 – Initialize the representative feature vector of each class
3 – Evaluation
    3.1 – Predict the class of each sample and check recognition success (*k*)
    3.2 – Compute the percentage of features used for each chromosome
    3.3 – Compute the new fitness vector
4 – Compute the two best elements (*best*)
5 – Repeat until convergence
    5.1 – Selection (*k*)
    5.2 – Crossover (*k*)
    5.3 – Mutation (*k*)
    5.4 – Evaluation (*3.1 (k), 3.2, 3.3*)
    5.5 – Compute the two best elements (*best*)

Table 1 shows the execution times for the main steps of the algorithm, in milliseconds, and the corresponding percentages. As can be seen, most of the computational effort is concentrated in the evaluation kernel (99.9%). The next step was to build parallel versions varying the level of parallelism. The parallel algorithm considers four kernels: evaluation (just the step 3.1) selection, crossover and mutation. In the first parallel version we have considered a two-dimensional index space for the evaluation kernel. One dimension for the classes and another for the chromosomes. Each kernel instance handles all the samples of one class per each chromosome. The other three kernels are one-dimensional kernels, considering a work item per individual.

Table 1. Single iteration execution times (milliseconds) for the sequential version in CPU and corresponding percentages

Sequential execution in CPU (ms)						
	Evaluation	Best	Selection	Crossover	Mutation	Total
Execution time per iter. (ms)	32096.5	10.5229	2.31662	9.08570	8.64061	32127.1
Relative percentage per iter.	99.905	0.033	0.007	0.028	0.027	100.000

In the first parallel version with evaluation implemented as a two-dimensional kernel three variants are considered: Eval(2D) with selection, crossover and mutation having one-dimensional NDranges; Eval(2D)Cros(2D) differs from the previous version because the crossover operator is implemented with a two dimensional index space; and Eval(2D)Cros(2D)Mut(2D) where the mutation operator also has a two-dimensional index space. To improve the most time consuming operator, another version was developed where a new dimension for the number of samples was added to the evaluation kernel now mapped into a three-dimensional NDrange, Eval(3D)Cros(2D)Mut(2D). The execution times for these parallel versions running in CPU are shown in Table 2.

Table 2. Single iteration execution times (milliseconds) for the parallel versions in CPU

Parallel execution times in CPU (ms)						
	Evaluation	Best	Selection	Crossover	Mutation	Total
Eval(2D)	5253.32	8.69	1.36	3.52	3.78	5270.67
Eval(2D)Cros(2D)	5239.27	8.83	1.42	4.05	3.97	5257.54
Eval(2D)Cros(2D)Mut(2D)	5250.20	8.88	1.46	4.05	4.94	5269.52
Eval(3D)Cros(2D)Mut(2D)	5239.53	8.97	1.46	3.90	5.04	5258.90

As can be seen, the execution time of each iteration decreases from about 32 seconds in the sequential version to approximately 5.2 seconds in all the parallel versions. These results mean a speed up of about 6 times. Moreover, increasing the level of parallelism has no impact, or has a negative effect. With just 4 cores (and 8 threads) increasing the parallel dimensions implies creating more threads (work items) and in some cases a worse execution time.

3.2 GPU Implementation

Next, the same versions studied before, the sequential and the four parallel variants are executed in GPU. The results are presented in Table 3. Comparing the sequential version executed in GPU with the parallel version with a two-dimensional evaluation, and the remaining kernels parallelized at the chromosome level, Eval(2D), we can see a speedup of 1046 times (figure 1). Now the version with 3D evaluation is done in less than half a second and is the fastest one with a speedup of 4125 times the sequential version in GPU.

Table 3. Single iteration execution times (milliseconds) for GPU versions

Execution times in GPU (ms)						
	Evaluation	Best	Selection	Crossover	Mutation	Total
Sequential	1934930,00	10,05	338,03	370,06	681,06	1936329,20
Eval(2D)	1803,87	9,33	13,50	16,65	7,09	1850,44
Eval(2D)Cros(2D)	1802,98	9,41	13,26	1,47	6,67	1833,80
Eval(2D)Cros(2D)Mut(2D)	1805,84	10,32	13,75	1,53	1,35	1832,80
Eval(3D)Cros(2D)Mut(2D)	442,95	9,76	13,66	1,55	1,41	469,34

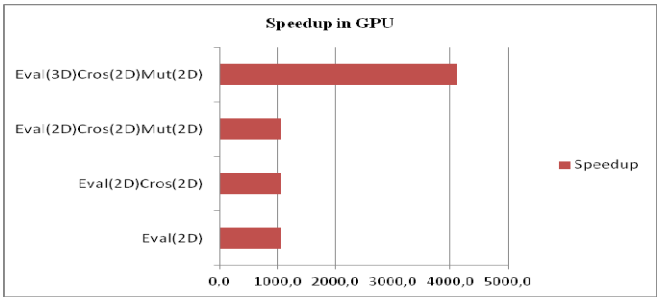


Fig. 1. Speedups, (sequential GPU / parallel GPU), for the parallel variants Eval(2D), Eval(2D)Cros(2D), Eval(2D)Cros(2D)Mut(2D) and Eval(3D)Cros(2D)Mut(2D)

In GPU, the impact of increasing the parallelism level is visible. Adding a new dimension in the crossover and mutation operators has a small but notorious impact. When a thread for each feature is created, the execution time for crossover decreases from 16.6 ms to about 1.5 ms and for mutation decreases from 7 ms to approximately 1.4 ms. As these two kernels are a very small part of the entire work, the two corresponding variants have no impact in the total execution time. Increasing the parallelism in the evaluation kernel has a much bigger impact in the execution time.

3.3 Quality of the Solutions

From the quality of the results standpoint there were no statistically significant differences observed between the different implementations described in this work. In the following set of graphs is depicted the behavior of a selected run for the slower and faster implementations of the GA.

Figures 2 and 3 present the evolution of the average fitness of the entire population as well as the higher value of fitness attained in each iteration. The figures clearly show that in just 20 iterations a steady state is achieved in both implementations.

Figures 4 and 5 present the percentage of selected features for the best 20 chromosomes. The chromosomes are ranked in descending order of fitness (eq. 1). Despite some slight variations on the average there are no statistically relevant differences between the two implementations.

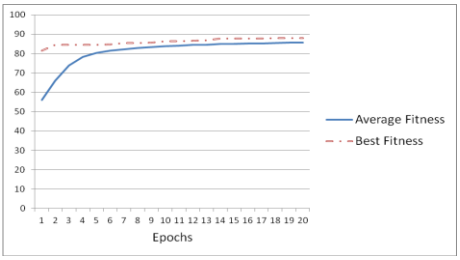


Fig. 2. Evolution of the average and best fitness for the CPU sequential version in CPU

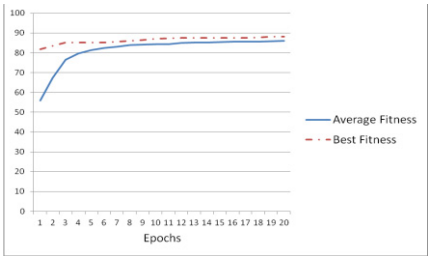


Fig. 3. Evolution of the average and best fitness for the parallel version in GPU, Eval(3D)Cros(2D)Mut(2D)

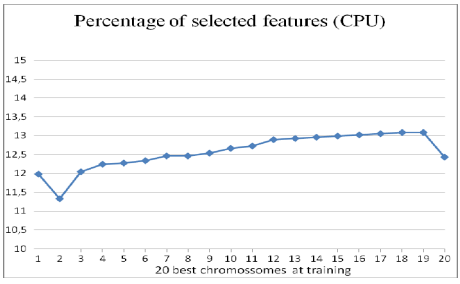


Fig. 4. Percentage of selected features for the sequential version in CPU

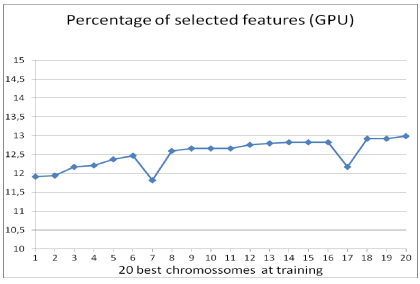


Fig. 5. Percentage of selected features for the Eval(3D)Cros(2D)Mut(2D) version in GPU

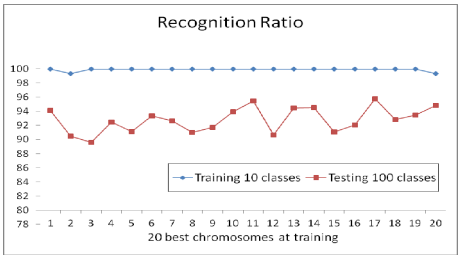


Fig. 6. Recognition ratio for training and testing for the sequential version in CPU

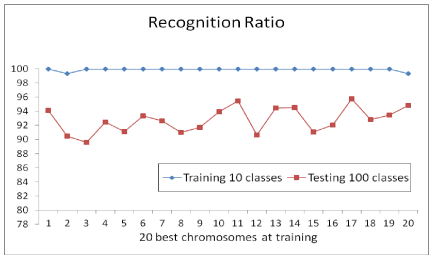


Fig. 7. Rec. ratio for training and testing for the Eval(3D)Cros(2D)Mut(2D) GPU version

Figures 6 and 7 present the recognition ratio for both training and testing situations for the best 20 chromosomes. As before the chromosomes are ranked in descending order of fitness (eq. 1). As before, despite some slight variations on the average there are no relevant differences between the two implementations.
It is worthwhile to mention that the testing recognition value using all the features is 96.4% a value comparable with some solutions of both implementations (in both cases using only a small fraction of the feature set).

4 Conclusion

In this work several OpenCL parallel implementations of a genetic algorithm for selection and weighting of the feature vector to be used in a k-NN classifier for periocular recognition are discussed. Different levels of parallel granularity were considered in CPU and GPU multi core platforms. For the studied problem the performance gains in CPU are independent of the granularity level of the parallel implementation. In GPU, considering an equivalent sequential implementation, the obtained speedup reaches several orders of magnitude for the highest level of parallelization.

From the quality of the results standpoint the consistent small percentage of selected features shows some degree of evidence that the removal of a large set of (non discriminant) features does not degrade the recognition rate thus allowing a simpler and faster feature extraction during the recognition task. Moreover there were no statistically significant differences observed between the different implementations.

Acknowledgements. The financial support given by "FCT-Fundação para a Ciência e Tecnologia" and "FEDER" in the scope of the PTDC/EIA/103945/2008 research project "NECOVID:Negative Covert Biometric Recognition" is acknowledged.

References

1. Park, U., Jillela, R., Ross, A., Jain, A.: Periocular biometrics in the visible spectrum. IEEE Transactions on Information Forensics and Security 6(1), 96–106 (2011)
2. UBIPr. UBI periocular recognition dataset, `http://socialab.di.ubi.ptuni/~ubipr`
3. Ojala, T., Pietikinen, M., Harwood, D.: A comparative study of texture measures with classification based on featured distributions. Pattern Recognition 29(1), 51–59 (1996)
4. Fang, Y., Wang, Z.: Improving LBP features for gender classification. In: Int.'l Conf. on Wavelet Analysis and Pattern Recognition, ICWAPR 2008, vol. 1, pp. 373–377 (2008)
5. Junding, S., Shisong, Z., Xiaosheng, W.: Image retrieval based on an improved CS-LBP descriptor. In: The 2nd IEEE Int.l Conf. on Information Management and Engineering (ICIME), pp. 115–117 (2010)
6. Coello, C., Veldhuizen, D.V., Lamont, G.: Evolutionary Algorithms for Solving Multi Objective Problems. Genetic Alg. and Evolutionary Comp. Series, vol. 5. Springer (2002)
7. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. John Wiley & Sons, New York (2001)

8. Goldberg, D.: Genetic Algorithms in search, optimization and machine learning. Addison-Wesley (1989)
9. Bäck, T., Fogel, D., Michalewicz, Z.: Handbook of Evolutionary Computation. Institute of Physics Publishing Ltd., Oxford Univ. Press, New York, Bristol (1997)
10. Prata, P., Fazendeiro, P., Sequeira, P.: Towards Cost-Effective Bio-Inspired Optimization: a Prospective Study on the GPU Architecture. In: Panigrahi, B.K., Suganthan, P.N., Das, S., Satapathy, S.C. (eds.) SEMCCO 2011, Part II. LNCS, vol. 7077, pp. 63–70. Springer, Heidelberg (2011)
11. Tsutsui, S.: Parallelization of an evolutionary algorithm on a platform with multi-core processors. In: The 9th Int.'l Conf. on Artificial Evolution, pp. 61–73. Springer (2009)
12. Pospichal, P., Jaros, J., Schwarz, J.: Parallel Genetic Algorithm on the CUDA Architecture. In: Di Chio, C., Cagnoni, S., Cotta, C., Ebner, M., Ekárt, A., Esparcia-Alcazar, A.I., Goh, C.-K., Merelo, J.J., Neri, F., Preuß, M., Togelius, J., Yannakakis, G.N. (eds.) EvoApplicatons 2010, Part I. LNCS, vol. 6024, pp. 442–451. Springer, Heidelberg (2010)
13. de Veronese, L., Krohling, R.: Differential evolution algorithm on the GPU with C-CUDA. In: IEEE Congress on Evolutionary Computation, CEC 2010, pp. 1–7 (2010)
14. Lőrentz, I., Andonie, R., Malița, M.: An Implementation of Evolutionary Computation Operators in OpenCL. In: Brazier, F.M.T., Nieuwenhuis, K., Pavlin, G., Warnier, M., Badica, C., et al. (eds.) Intelligent Distributed Computing V. SCI, vol. 382, pp. 103–113. Springer, Heidelberg (2011)
15. Holland, J.: Adaptation in Natural and Artificial Systems. Univ. of Michigan Press (1975)
16. Adams, J., Woodard, D.L., Dozier, J., Miller, P., Bryant, K., Glenn, G.: Periocular Biometric Recognition: Less is More. In: Int.'l Conf. on Pattern Recognition, pp. 205–208 (2010)
17. Eshelman, L., Schaffer, J.: Real-coded genetic algorithms and interval-schemata, vol. 3, pp. 187–202. Morgan Kaufmann, San Mateo (1993)
18. Munshi, A. (ed.): The OpenCL Specification Version: 1.1, Khronos OpenCL Working Group, 385 pages (2011)

Face Authentication Using Supervised Learning Techniques

Amioy Kumar, Rohan Gupta, Akshay Sharma,
Bijaya Ketan Panigrahi, and M. Hanmandlu

Department of Electrical Engineering,
Indian Institute of Technology Delhi,
Hauz Khas, New Delhi, India-130016

Abstract. The growing popularity of face biometrics for human authentication is due to its high user convenience and acceptance. Face recognition has evolved as an important application of biometrics and has been a topic of interest for several machine learning and computer vision communities. However, most of the attempts on face based personal authentication rely on decision threshold for accept or reject of the claimed identity. This paper investigates supervised learning techniques for face verification. The presented approach deals with computation of 5^{th} level Haar wavelet coefficients of image used as feature for training of the classifiers like SVM, fuzzy SVM and KNN. The extracted biometric features are matched to compute genuine and impostor matching scores. The error rates FAR and FRR are then calculated using cross validation of the test set. The experiments are carried out on Yale database of 37 users with 25 images of each user. In our work we obtained FAR and FRR of 0.3285 and 0.1967 respectively which demonstrates the reliability of the proposed work.

Index Terms: Biometric authentication, FAR, FRR, Gabor filter, Haar wavelet, SVM.

1 Introduction

Biometrics based personal authentication has gained popularity because of its high user reliability and convenience. It is aimed at determining the claimed identity based on user's physiological or behavioral traits [1]. The selection of a biometrics trait in the authentication is based upon each person's unique physical or behavioural characteristics [1]. Leading examples are biometric technologies that recognize and authenticate faces, hands, fingers, signatures, irises, voices, and fingerprints The accuracy in any biometric database can be quantified in terms of error rates: false acceptance rate (FAR) and false rejection rate (FRR). The biometric applications demand a tradeoff between these two error rates to meet the requirements varying from forensic to high security applications [2].This can be achieved by setting a suitable threshold. A conventional way to choose the optimal threshold in a system is to obtain a FAR Vs. GAR (GAR=1-FRR) plot, also known as Receiver Operating Characteristics Curve (ROC) and the decision threshold can be chosen from this plot

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 738–745, 2012.

based on the desired error rates [1]. The selected decision threshold is then used to make reject or accept decision on the claimed identity at the verification stage.

Another way of making decision in the biometric verification can be based on pattern classification approach where the biometric features obtained from training samples can be submitted to a *classifier* to train a predictive model by utilizing their properties, and then label them into genuine or imposter classes [3][4][5]. The learned predictive model can be further used at the verification stage for the classification of the claimed identity into any of the acceptance or the rejection classes. A genuine class is meant for acceptance while the imposters are for the rejection during the classification. This kind of verification can be categorized as *supervised learning approach* where the training set is drawn from the population (database) and the system is modeled with the help of the *acceptance* (genuine) and *rejection* (imposter) class labels computed from all the user. The computation of the error rates (FAR/FRR) is based on testing the cross validation of the testing set from the trained classifier. The minimum values of these errors ascertained the performance of the chosen predictive model.

Hand based biometrics have been the most acknowledged for personal authentication. Not only due to its superior performance which is required for the high security applications, but also for their high distinctiveness, user convenience, and acceptance. However, people leave their palm/handprint unconsciously wherever they touch and which increases the possibilities of imposter attacks on these security systems. The growing popularity of face biometrics for human authentication is due to its high user convenience and acceptance. However, most of the attempts on face based personal authentication rely on decision threshold for accept or reject of the claimed identity. This paper therefore investigates supervised learning techniques for face verification. The presented approach deals with computation of 5^{th} level Haar wavelet coefficients of image used as feature for training of the classifiers like SVM, fuzzy SVM and KNN. The error rates FAR and FRR are then calculated using cross validation of the test set. The experiments are carried out on Yale database of 37 users with 25 images of each user. The block diagram of the complete system is shown in Fig. 1.

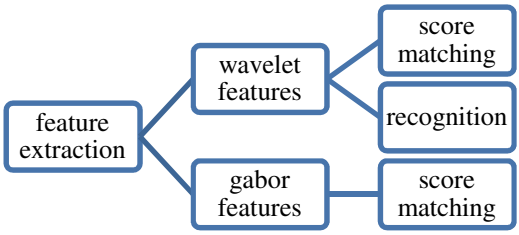


Fig. 1. Block diagram overview of our approach

The rest of this paper is organized as follows: Gabor and wavelet feature extraction computation is discussed in Section 2. Classification techniques are discussed in section 3. The experimental results are carried out in Section 4, and finally the conclusions are drawn in section 5.

2 Feature Extraction

We have considered Yale database of face images [13] with 40 users and 25 images each in our experiments. The sample images of each user are full of pose and illumination variations. Some sample images from the database are as follows:

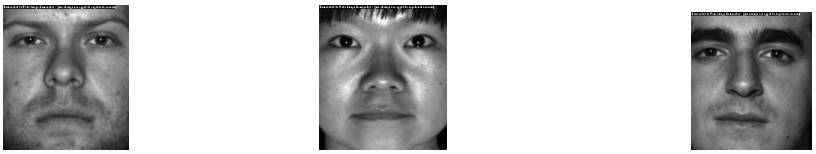


Fig. 2. Cropped images from yale database

2.1 Gabor Filter

Face representation using Gabor features has attracted considerable attention in computer vision, image processing, pattern recognition, and so on. The principal motivation to use Gabor filters is biological relevance that the receptive field profiles of neurons in the primary visual cortex of mammals are oriented and have characteristic spatial frequencies. Gabor filters can exploit salient visual properties such as spatial localization, orientation selectivity, and spatial frequency characteristics[4]. Gabor filter works as a bandpass filter for the local spatial frequency distribution, achieving an optimal resolution in both spatial and frequency domains. The 2D Gabor filter can be represented as a complex sinusoidal signal modulated by a Gaussian kernel function[5] by expression (1):

$$G(x, y, \theta, f) = \exp\left\{-\left(\frac{1}{2}\right)\left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right)\right\} \exp(2\pi f x) \quad (1)$$

Where x = xcos θ + ysin θ, y = ycos θ – xsin θ, θ represents the orientation of Gabor Filter, f represents the frequency of sinusoidal function, σ_x and σ_y are variances along x and y axis respectively.

Fig. 3. Gabor filtered images of different users

2.2 Haar Wavelet Transform

Haar Wavelet like other transforms contains discretely sampled wavelets which captures both frequency and location information. For an input represented by a list of

2*n* numbers, the Haar wavelet transform pairs up input values, storing the difference and passing the sum. This process was repeated recursively, pairing up the sums to provide the next scale: finally resulting in (2*n* – 1) differences and one final sum The discrete wavelet transform has a huge number of applications in science, engineering, mathematics and computer science. Most notably, it is used for signal coding, to represent a discrete signal in a more redundant form, often as a preconditioning for data compression [6]. The Haar transformation technique is used [7] to form a wavelet since it is the simplest wavelet transformation method of all and can effectively serve our interests. In the Haar wavelet transformation method, low-pass (g) filtering is conducted by averaging two adjacent pixel values, whereas the difference between two adjacent pixel values is figured out for high-pass (h) filtering.

$$g = \frac{1}{\sqrt{2}}[1 \quad 1], \qquad h = \frac{1}{\sqrt{2}}[1 \quad -1]$$

The above filters are separately applied to the rows and columns of the face images resulting in four channel filter bank with channels LL, LH, HL, and HH corresponding to filters *g*t * *g*, *g*t * *h*, *h*t * *g*, and *h*t * *h* respectively. The recursive application of this decomposition is used to construct higher level decomposition.

In this work, the enhanced images are decomposed into 5 levels by the Haar wavelets. Next the vertical, horizontal and diagonal coefficients of 4th and 5th level were employed. The coefficients of 1st, 2nd, and 3rd level were almost the same as those of the 4th level and therefore the smallest of them (4th level coefficients) were employed and rest were ignored. The 5th level decomposition offered the most discriminative information and therefore all the coefficients from this decomposition were employed. The phase encoding from the zero crossings of the coefficients formed the binary values of the feature vector. The size of this feature vector was 925 × 504, where rows are signifying different users and columns are signifying haar coefficients. The Hamming distance was then employed to ascertain the matching distance between feature vectors.

3 Classification Techniques

3.1 Support Vector Machine (SVM)

Support vector machine (SVM) is a popular tool for machine learning task. It has been successfully applied in many fields, but the parameter optimization for SVM is an ongoing research issue[13]. Support vector machine (SVM) is based on the structural risk minimization (SRM) principle [8], which makes it less prone to over-fitting. By maximizing the margin between two opposite classes, SVM can find the optimal separating hyper-plane that minimizes the upper bound of the generalization error, which enables SVM to have strong capability of fitting and generalization. By introducing the kernel tricks, SVM has the ability of dealing with infinite or nonlinear features in a high dimensional feature space. With the above attractive features, SVM is regarded as state-of-the-art classifier.

Suppose that we have a set of training samples {(X1, y1),(X2, y2)...(XN, yN)}. Each Xi has a class label $y_i \in \{1,1\}$ which denotes two classes separately. When the

samples are linear separable, the SVM can separate them with a largest margin between the two classes without any wrong separated points. This can be achieved by solving the following quadratic program:

$$\begin{array}{ll} \min & \|w\|^2 \\ y_i(w^T X_i + b) \geq 1 & i = 1,2,\ldots\ldots,N \end{array} \quad (2)$$

Where w is the weight vector and b is the bias term. For a nonlinearly separable case, it is not possible to satisfy all the constraints in (1) . Thus, slack variables ξ_i, i ∈ {1, 2, ...,N} are introduced to measure the amount of violation of the constraints. The QP problem becomes:

$$\begin{array}{ll} \min & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i \\ y_i(w^T X_i + b) \geq 1 - \xi_i & i = 1,2,\ldots\ldots,N \\ \xi_i \geq 0 & i = 1,2,\ldots\ldots,N \end{array} \quad (3)$$

Where C is a parameter which has to be determined beforehand to define the cost of constraint violation, a larger C means a higher penalty is assigned to empirical errors. However, in practice, most of the problems cannot be solved by such a linear classifier; thus, a nonlinear extension is necessary. This can be achieved by mapping the input variable X_i into a higher dimensional feature space and thus using kernel method. Some commonly used kernel functions are polynomial, sigmoid and Gaussian function.

3.2 FuzzySVM

On the basis of the theory of classical SVM , Lin pro-posed the theory of fuzzy support vector machine in Ref. Chun[9]. In classical SVM, each sample is treated equally ; i.e., each input point is fully assigned to one of the two classes. However, in many applications, some input points, such as the outliers, may not be exactly assigned to one of these two classes, and each point does not have the same meaning to the decision surf ace. To solve this problem, fuzzy membership to each input point of SVM can be introduced, such that different input points can make different contribution to the construction of decision surface. The FSVM can achieve good performance since it is an average algorithm. A particular sample in the training set only contributes little to the final result and the effect of outliers can be eliminated by taking average on the samples. We have used single class SVM and Multi class SVM for calculation of FAR and FRR.

3.3 KNN

K-Nearest Neighbour (KNN) classification is a very simple, yet powerful classification method. The key idea behind KNN classification is that similar observations belong to similar classes. Thus, one simply has to look for the class and weigh their class sign a class number to the unknown. The weighing scheme was

majority base d. The number of the nearest neighbors, k, should be designators of a certain number of the nearest neighbours numbers to as chosen to be odd in order to avoid ties. Various distance types are used in KNN Classification. [10]. Verification of a query face image is determined by the class of its k-nearest neighbour. If Class of a query face image is same as output of classifier, the face image is matched otherwise not.

4 Experimental Results

The database used is Yale database comprising of 37 users with 25 images of each user[13]. The output obtained Gabor and wavelet were used for score matching and for further classification. The score matching was done using hamming distance and by single class SVM. Normalized hamming distance has been used as a criterion for classifying a user as genuine or impostor . A normalized Hamming distance used in [11] is adopted to determine the similarity measurement for face image matching. Let A and B are two face images then normalized hamming distance is given by expression (4).

$$D = \frac{xor(A,B)}{N*N} where\ N * N\ denotes\ the\ size\ of\ image \quad (4)$$

We divided the database into two sets training and testing with 10 images for training and 15 images in testing. For the calculation of scores we divided the dataset of 25 images users per user into 10 images and 15 images and which were further divided into 6 and 4 images and 8 and 7 images respectively. Further the genuine and impostor scores were calculated and finally the FAR and FRR values were obtained. In Fig.4 the Receiver operating Characteristic (ROC) plot was obtained which is a graphical representation of FAR and GAR (100-FRR). It is used to set up an optimal threshold for the biometric system. Same approach was adopted for computing the genuine and impostor matching scores and the outputs obtained after applying the Gabor filter, wavelet transform and SVM on the wavelet data on the image datasets. Our results for FAR and FRR are shown in Table.1.

Table 1. FAR and FRR obtained after the application of normalized hamming distance

Technique	FAR(%)	FRR(%)
Gabor output data	0.375	4.864
Wavelet output data	1.76	0
SVM on wavelet output data	0.3285	0.1967

To validate our verification algorithms we applied KNN , SVM and Fuzzy SVM on the wavelet feature matrix. From the Yale database we had considered taken 10 images of each user and we had taken 6 images for training and 4 images for testing. We observed that the KNN and Fuzzy SVM performed better than multi-class SVM and KNN when the parameters K=1 (nearest neighbour) and distance type – correlation were chosen.

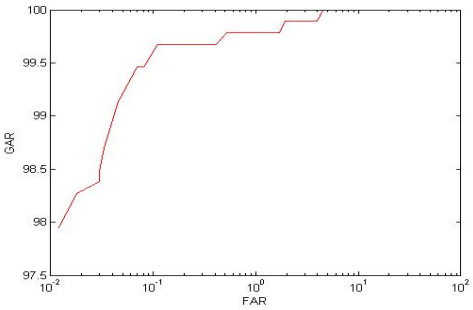


Fig. 4. Plot representing FAR and GAR or ROC curve calculated on wavelet feature matrix

Table 2. demonstrates the recognition rate after the application of KNN using different values of k and using different distances

K	Distance	Recognition rate
2	Euclidean	99.45%
2	Cityblock	99.45%
1	Correlation	99.82%

Table 3. demonstrating the recognition rate obtained using SVM classifier

Classifier	Recognition rate(%)
Multi Class SVM	99.1
Fuzzy SVM	99.35

5 Conclusion

This paper presents a pattern classification approach for face image based biometric recognition. The coefficients obtained after the application of 5th level Haar wavelet reduced the size of the feature matrix significantly, we have also applied Gabor filter to the image database and calculated the FAR and FRR using single class SVM and hamming distance. We observed that the results were better for wavelet output. We also computed the recognition rate on wavelet output using fuzzy SVM and multi-class SVM. Fuzzy SVM resulted in better recognition as compared to multi-class SVM.

References

[1] Jain, A.K., Ross, A., Prabhakar, S.: An introduction to biometric recognition. IEEE Transactions on Circuits and Sytems for Radio Technonlogy 12(1) (January 2004)

[2] Chellappa, R., Wilson, C.L., Sirohey, S.: Human and machine recognition of faces: A survey. Proc. IEEE 83, 705–740 (1995)

[3] Wechsler, H., Phillips, P., Bruce, V., Soulie, F., Huang, T.: Face Recognition: From Theory to Applications. Springer (1996)
[4] Bhuiyan, A., Liu, C.H.: On face recognition using Gabor filter. World Academy of Science, Engineering and Technology 28 (2007)
[5] Barbu, T.: Gabor filter–based face recognition technique. In: Proceedings of the Romanian Academy series A 13(3), 277–283 (2010)
[6] Mallat, S.G.: A wavelet tool of signal processing. Academy Press (1999)
[7] Chang, C.-C., Chaung, J.-C., Hu, Y.-S.: Similar image retrievel based on wavelet transformation. International General of Wavelets, Multiresolution and Information Processing 2(2), 111–120 (2004)
[8] Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer, New York (1999)
[9] Jieng, X., Yi, Z., Lv, J.C.: Fuzzy SVM with a new fuzzy membership function. Neural Compute and Applic. 15, 268–276 (2006)
[10] Bioinformatics toolbox User's Guide by the Math Works, Inc.
[11] Daugman, J.G.: High confidence visual recognition of persons by a test of statistical independence. IEEE Transactions on Pattern Analysis and Machine Intelligence 15(13), 1348–1361 (1993)
[12] Chang, C.-C., Lin, C.-J.: LIBSVM-A libary of Support vector machines. ACM Transactions on Intelligent Systems and Technology 2(3) (2011)
[13] Yale face database, `http://cvc.yale.edu/projects/yalefaces/yalefaces.html`

Design of Non-uniform Circular Antenna Arrays Using Coordinated Bacterial Dynamics and Opposite Numbers

Jaydeep Ghosh Chowdhury[1], Aritra Chowdhury[1],
Arghya Sur[1], and Swagatam Das[2]

[1] Department of Electronics and Telecommunication Engg.,
Jadavpur University,
Kolkata-700 032, India
{jaydeep197,arghyasur1991}@gmail.com,
arit0001@yahoo.co.in,
[2] Electronics and Communication Sciences Unit,
Indian Statistical Institute,
Kolkata-700 108
swagatam.das@isical.ac.in

Abstract. In this paper, a very simple optimization algorithm called Coordinated Bacterial Dynamics and Opposite Numbers (CBDO) is proposed to design a non-uniform circular antenna array with optimum side lobe level reduction. Here the bacterial dynamics is used to determine an optimum set of the excitation current amplitudes of antenna elements and element separations of a non-uniform planar circular antenna arrays which can achieve minimum side lobes levels for a fixed first null beam-width. In our proposed algorithm, there are only three bacterial searching agents. One primary bacterium follows two secondary bacteria to reach global optima. The design results obtained using CBDO shows that it provides better side lobe level reduction than that obtained using modified IWO, basic IWO, DE, PSO and GA.

Keywords: Circular antenna arrays, optimization, side lobe suppression, directivity, opposition-based coordinated bacterial dynamics.

1 Introduction

In many occasions, a single element antenna cannot provide highly directive radiation patterns to meet the demands of long distance communication. To solve this problem, we use antenna arrays containing a number of individual radiating elements in a certain electrical and geometrical configuration [1-4]. To provide highly directive characteristics, the electro-magnetic fields for the elements of the antenna array should add constructively in a desired direction and should add destructively and vanish in the other directions. So it is very important to reduce side lobe levels to design highly directive antenna arrays.

Panduro *et al.* [5] applied the real-coded genetic algorithm (GA) to design circular arrays with maximum side lobe level reduction coupled with the constraint of a fixed

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 746–753, 2012.

beam width. Then Shihab *et al.* in [6] applied the particle swarm optimization (PSO) algorithm to the same problem. Panduro *et al.* [7] compared three powerful population-based optimization algorithms— PSO, GA, and differential evolution (DE) in the design problem of scanned circular arrays. Recently a new optimization algorithm invasive weed optimization (IWO) [8] and an improved version of IWO [9] were applied to the same problem. The experimental results show that our proposed algorithm CBDO has ability to achieve much better results than the results obtained using some of recently developed EAs in designing problem of circular antenna arrays.

2 Coordinated Bacterial Dynamics and Opposite Numbers

2.1 Dynamic Random Search

The primary bacterium occupies a position in the n dimensional space and performs a random search to obtain the optimum solution. The primary bacterium at the k^{th} iteration may be represented as,

$$X^k = (x_1^k, x_2^k, x_3^k, ..., x_n^k) \in R^n, \tag{1}$$

where x_i^k is the component of the X^k on the i^{th} dimension at k^{th} iteration. Now an associated bacterium $\tilde{X}$ is placed from the primary bacterium by dynamic random search. $\tilde{X}$ is associated to X as a mutation of the primary bacterium. A dimension $l \in \{1,2,3,.....,n\}$ is randomly chosen for the mutation where n is the number of dimensions of the primary as well as the associated bacterium. Then at k^{th} iteration the associated bacterium is generated as

$$\tilde{X}^k = X^k + \delta D^k, \tag{2}$$

where $\delta D^k = (0,0,0,.....,d_l^k,0,0,.....,0)$ and d_l^k is randomly chosen between the lower and upper boundary on the l^{th} dimension which is defined by

$$d_l^k = c_1 r_1 (UB - LB), \tag{3}$$

where r_1 is a random number chosen in the range between -1 to 1. So, $r_1 \epsilon$ *[-1, 1]. UB* and *LB* denote the upper and lower boundary respectively. Searching Co-efficient c_1 is a constant which plays a role equivalent to maintaining the diversity of the population, as adopted in conventional Evolutionary Algorithms like DE, PSO etc. In our proposed algorithm, c_1 is set at 0.23.

Similarly, the opposite associated bacterium $\tilde{X}^*$ is expressed as,

$$\tilde{X}^* = X - \delta D^k \quad . \tag{4}$$

2.2 Simplified Quorum Sensing

The primary bacterium can sense the position of the fittest among the associated bacteria by simplified quorum sensing. We have implemented the movement of the bacterium by introducing a velocity to the primary bacterium.

The velocity of the primary bacterium is denoted as $V^k = (v_1^k, v_2^k, v_3^k, ..., v_n^k)$ where v_i^k is a component of the V^k on the i^{th} dimension at k^{th} iteration.

Let FB^k be the position of the fittest bacterium between the associated and opposite associated bacterium. Then the velocity of primary bacterium is updated at the k^{th} iteration as,

$$v_i^k = FB_i^k - x_i^k \tag{5}$$

Where, FB_i^k and x_i^k are the components of the fittest bacterium and primary bacterium on the i^{th} dimension at k^{th} iteration respectively. Thus, the position of the primary bacterium is updated as,

$$X^{k+1} = X^k + V^k \tag{6}$$

2.3 Pseudo Code of CBDO

The pseudo code of CBDO is given as shown:

1. Initialize primary bacterium X randomly within the upper and lower boundaries of the n dimensional search space and evaluate the objective functional value $f(X)$ of the primary bacterium.
2. WHILE *FES*<*max_FE (FES* is the number of fitness function evaluations*)*
3. Select a dimension l randomly.
4. Place the associated bacterium $\tilde{X}$ along the randomly selected dimension.
5. Evaluate the objective functional value of the associated bacterium $f(\tilde{X})$.
6. Increment the number of fitness evaluations *FES* as, *FES*=*FES*+1.
7. Place opposite associated bacterium $\tilde{X}^*$ along same dimension but in different (opposite) direction of associated bacterium.
8. Evaluate the objective functional value of the opposite associated bacterium $f(\tilde{X}^*)$.
9. *FES*=*FES*+1
10. IF $f(\tilde{X}) < f(\tilde{X}^*)$
11. IF $f(\tilde{X}) < f(X)$

$$V = \tilde{X} - X$$
$$X = X + V$$
$$f(X) = f(\tilde{X})$$

12. END
13. ELSE
14. IF $f(\tilde{X}^*) < f(X)$

$$V = \tilde{X}^* - X$$
$$X = X + V$$
$$f(X) = f(\tilde{X}^*)$$

15. END
16. END
17. END WHILE

3 Geometry and Array Factor of Circular Antenna Array

We consider a non-uniform planar circular antenna array where *N* antenna elements are non-uniformly spaced on a circle of radius *r* in x-y plane .

The radiation pattern of this antenna array can be described by its array factor. In the x-y plane the array factor for the circular array can be expressed as,

$$AF(\phi) = \sum_{n=1}^{N} I_n e^{j(kr\cos(\phi-\phi_n)+\alpha_n)} \tag{9}$$

And kr and ϕ_n can be written as

$$kr = \frac{2\pi r}{\lambda} = \sum_{i=1}^{N} d_i \tag{10}$$

$$\phi_n = \frac{2\pi}{kr}\sum_{i=1}^{n} d_i \tag{11}$$

I_n And I_n and α_n are the excitation amplitude and phase of *n*-th element respectively, and d_n is the arc separation distance (in terms of wavelength) between the element *n* and *(n-1)*. And ϕ_n is the angular position of *n*-th element in x-y plane. If we want to direct the main beam in ϕ_0 direction, then the excitation phase of the *n-th* element is chosen as,

$$\alpha_n = -kr\cos(\phi_0 - \phi_n) \tag{12}$$

So the expression for array factor becomes

$$AF(\phi) = \sum_{n=1}^{N} I_n e^{j[kr\{\cos(\phi-\phi_n)-\cos(\phi_0-\phi_n)\}]} \tag{13}$$

Without loss of generality we have chosen $\phi_0 = 0$, i.e. the peak of main beam is chosen to be in the x-direction.

4 Designing the Fitness Function

Our objective is to design a circular antenna array with minimum side lobes levels for a fixed first null beam width (FNBW). To design an antenna array with highly directive characteristics, we have incorporated maximum side lobe level in addition to average side lobe level in the fitness function. Thus the following fitness function is used,

$$Fitness = f_{NU} + f_{SLA} + f_{MSL} \tag{14}$$

$$f_{NU} = \left|AF(\phi_{NULL1})\right| + \left|AF(\phi_{NULL2})\right| \tag{15}$$

$$f_{SLA} = \frac{1}{\pi + \phi_{NULL1}} \int_{-\pi}^{\phi_{NULL1}} \left|AF(\phi)\right| d\phi + \frac{1}{\pi + \phi_{NULL2}} \int_{\phi_{NULL2}}^{\pi} \left|AF(\phi)\right| d\phi \tag{16}$$

$$f_{MSL} = \left|AF(\phi_{MSLL1})\right| + \left|AF(\phi_{MSLL2})\right| \tag{17}$$

Where ϕ_{NULL1} and ϕ_{NULL2} are two angles at the null. We minimize the fitness function at two angles ϕ_{NULL1} and ϕ_{NULL2} defining the major lobe, i.e. the first null beam width $FNBW = \phi_{NULL2} - \phi_{NULL1}$. ϕ_{MSLL1} and ϕ_{MSLL2} are the angles where we get the maximum side lobe level in lower bound $[-\pi, \phi_{NULL1}]$ and in upper bound $[\phi_{NULL2}, \pi]$ respectively. Through minimizing f_{MSL}, we actually minimize the maximum side lobe level and we also minimize average side lobes level by minimizing f_{SLA}. Now we have to find the excitation current amplitudes and arc separations between the elements that minimize the fitness function.

5 Results

We have considered three cases where the numbers of elements are 8, 10 and 12 in circular antenna arrays. We have performed the experiments for a fixed value of first null beam width (FNBW), corresponding to a uniform circular antenna array [5]. For practical consideration the current amplitudes are normalized with the maximum value of current amplitudes being unity.

Table-1 compares the optimum solution obtained using different algorithms in terms of the mean and standard deviation of the best-of-run values for 50 independent runs. Table-2 provides the best values of optimized variables (normalized currents I_n and element separations d_n in terms of wavelength) obtained after 50 independent runs of our proposed algorithm CBDO. Table-3 compares the figures of merit obtained using modified IWO and our proposed algorithm CBDO in the best of 50 independent runs.

Table 1. Mean and standard deviations of the final cost function values

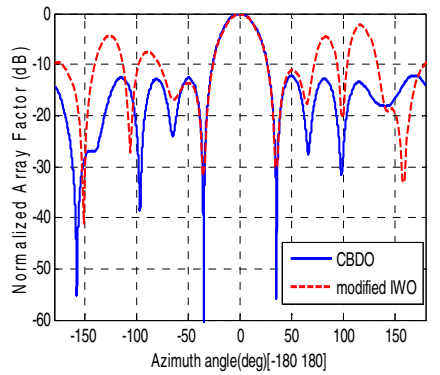
Number of elements	Algorithm	Mean Cost Function Value	Standard deviation for Cost Function
8	CBDO	0.3078	0.0145
	Modified IWO [9]	1.0687	0.0122
	Original IWO [8]	1.1990	0.1726
	DE [7]	1.1355	0.1338
	PSO [6]	1.2079	0.4412
	GA [5]	1.4011	0.7815
10	CBDO	0.4123	0.0184
	Modified IWO [9]	0.6799	0.0294
	Original IWO [8]	0.8705	0.2017
	DE [7]	0.7895	0.0988
	PSO [6]	0.8461	0.2244
	GA [5]	1.1215	0.4996
12	CBDO	0.5042	0.0221
	Modified IWO [9]	0.7288	0.0911
	Original IWO [8]	09873	0.9885
	DE [7]	0.8046	0.2355
	PSO [6]	0.8918	0.9890
	GA [5]	1.1256	0.9929

Table 2. Design variables obtained with CBDO algorithm

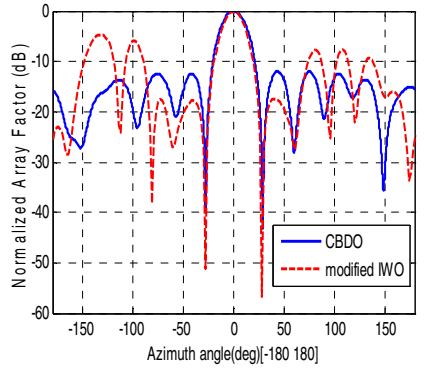
No. of Elements	FNBW	d_n in terms of wavelengths	Normalized I_n
8	70.27	0.3464, 0.5364, 0.3181, 0.7864, 0.6446, 0.8372, 0.7843, 0.3411.	0.7501, 0.3137, 0.5746, 0.7108, 0.8975, 0.3167, 1.0000, 0.1146.
10	55.85	0.4174, 0.8013, 0.3944, 0.9998, 0.3417, 0.2890, 0.9913, 0.3358, 0.9874, 0.2834.	0.4729, 0.7319, 0.3616, 0.8458, 0.0387, 1.0000, 0.4610, 0.5089, 0.6201, 0.6685.
12	46.26	0.5312, 0.8663, 0.8476, 0.6965, 0.8453, 0.5062, 0.5322, 0.5516, 0.9709, 0.9544, 0.9903, 0.9705.	0.3839, 0.0499, 0.3700, 0.3528, 0.6590, 0.9673, 1.0000, 0.7984, 0.6504, 0.0172, 0.3585, 0.8455.

Table 3. Design figures of merit obtained in the best (out of 50) runs on three design instances

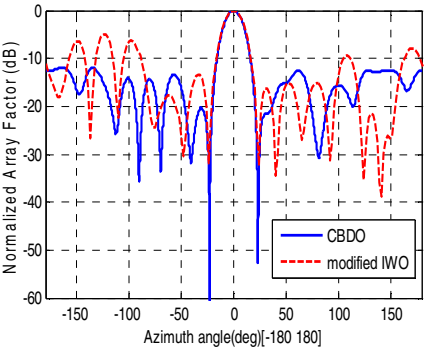
Number of elements	Algorithm	Maximum Side Lobe Level in decibels	Average Side Lobe Level in decibels	Directivity in decibels
8	CBDO	-12.1662	-35.7732	14.22
	Modified Iwo [9]	-02.2797	-24.0958	9.14
10	CBDO	-12.0365	-35.9932	13.92
	Modified Iwo [9]	-04.6371	-34.7538	11.75
12	CBDO	-11.8945	-34.7443	17.04
	Modified Iwo [9]	-05.1054	-31.6817	12.25



(a)

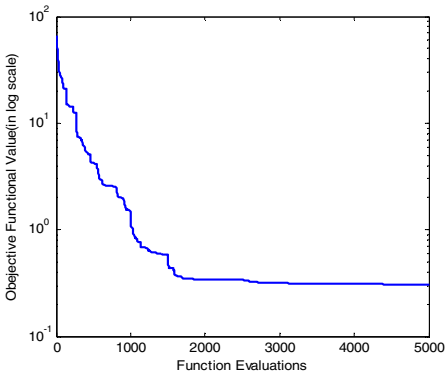


(b)

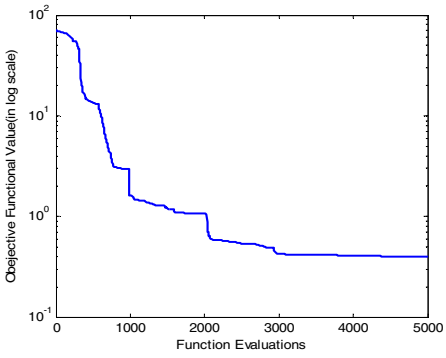


(c)

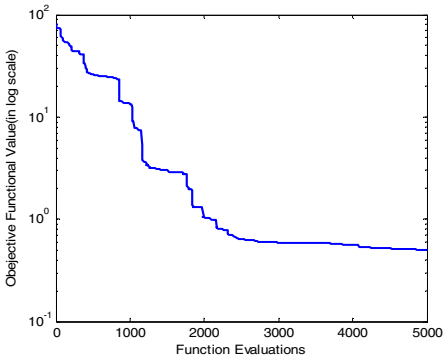
Fig. 1. Normalized radiation patterns for circular arrays of different number of elements obtained using CBDO and modified IWO [9]. (a) For number of elements N=8 (b) for number of elements N=10, (c) for number of elements N=12.



(a)



(b)



(c)

Fig. 2. Convergence characteristics of our proposed CBDO algorithm and modified IWO [9] over three cases of the circular array design problem. (a) For number of elements N=8, (b) for number of elements N=10, (c) for number of elements N=12.

So, from Table-1 and Table-3 it is evident that our proposed algorithm CBDO has the ability to achieve better optimum solutions than the other algorithms. Figure 1 shows that the side lobe levels are suppressed by a great deal as all the side lobes have levels less than -10dB in case of 8 and 10 element arrays. Figure 2 shows that CBDO is a fast algorithm and it has a very high convergence speed.

6 Conclusion

In this paper our proposed algorithm CBDO is used to determine the excitation amplitude and position of each element in circular antenna array to obtain minimum side lobes levels and highly directive characteristics. Radiation patterns obtained using CBDO are better than those obtained from modified IWO [9], original IWO [8], DE [7], PSO [6] and GA [5]. By treating three components f_{NU} , f_{SLA} and f_{MSL} of the cost function as a multi-objective optimization problem, a much better result can be achieved in future.

References

1. Godara, L.C.: Handbook of Antennas in Wireless Communication. CRC, Boca Raton (2002)
2. Chandran, S.: Adaptive Antenna Arrays: Trends and Applications. Springer, New York (2004)
3. Tsoulos, G.V.: Adaptive Antennas for Wireless Communications. IEEE Press, Piscataway (2001)
4. Balanis, C.A.: Antenna Theory: Analysis and Design, New York (1997)
5. Panduro, M., Mendez, A.L., Dominguez, R., Romeo, G.: Design of non-uniform circular antenna arrays for side lobe reduction using the method of genetic algorithm. Int. J. Electron. Commun. (AEU) 60, 713–717 (2006)
6. Shihab, M., Najjar, Y., Dib, N., Khodier, M.: Design of non-uniform circular antenna arrays using particle swarm optimization. J. Elect. Eng. 59(4), 216–220 (2008)
7. Panduro, M.A., Brizuela, C.A., Balderas, L.I., Acosta, D.A.: A comparison of genetic algorithm, particle swarm optimization and the diffrential evolution method for the design of scannable circular antenna arrays. Progr. Electromagn. Res. B 13, 171–186 (2009)
8. Mehrabian, A.R., Lucas, C.: A novel numerical optimization algorithm inspired from weed colonization. Ecolog. Inform. 1, 355–366 (2006)
9. Roy, G.G., Das, S., Chakraborty, P., Suganthan, P.N.: Design of non-uniform circular antenna arrays using a modified invasive weed optimization algorithm. IEEE Transactions on Antennas and Propagation 59(1) (January 2011)

Process Plan and Scheduling Integration for Near Optimal Process Plans in Networked Based Manufacturing Using Controlled Elitist NSGA-II and Territory Defining Algorithms

V.K. Manupati, J.J. Thakkar, Priyabrata Mohapatra, Ajay Kumar, M.K. Tiwari

Department of Industrial Engineering and Management,
Indian Institute of Technology Kharagpur,
Kharagpur-721302, West Bengal, India.
{manupativijay,Priya.nitrkl,akhilesh.nifft}@gmail.com,
jt@iem.iitkgp.ernet.in, mkt09@hotmail.com

Abstract. The networked based manufacturing offers various advantages in current competitive atmosphere by way to reduce the short manufacturing cycle time and to maintain the production flexibility. In this paper, we have addressed a multi-objective problem whose objectives are to minimize the makespan and maximization of machine utilization for generating feasible process plans of multiple jobs in the context of networked based manufacturing system. In more specific, with two powerful multi-objective evolutionary algorithms (MOEAs) namely controlled elitist-NSGA-II (CE-NSGA-II), and territory defining evolutionary algorithm (TDEA), were proposed to find the better performance of the system. With the help of an illustrative example along with two complex scenarios these algorithms has been implemented, tested and compared. Finally, the computational results are analyzed to the benefit of the manufacturer.

1 Introduction

Recent developments in information technology and communication technology have profoundly influenced the manufacturing research and its application. However, the products functionality and complexity are increase in extent and the organizations need to sustain the advantage of huge competitiveness in the market. Hence, automation of knowledge-based manufacturing for attaining effective decisions has become more essential to improve the business market. In order to meet rapid expansion of demands of global customers, there is a need to change the traditional manufacturing system. This can be attained through the adoption of recently emerged manufacturing paradigm named as network based manufacturing or networked manufacturing.

Process planning and scheduling are the two significant functions to be engaged to process various operations of the jobs in a manufacturing system. These functions specifies the decision maker of how, when and in which sequence the operations of

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 754–760, 2012.

the parts are allocated to the manufacturing resources. To realize this, some researchers have realized that there is a greater need to integrate both the functions to achieve better performance of the system. The fundamental idea of the integration of process planning and scheduling functions has been introduced by Chryssolouris and Chan [1]. The above mentioned integration approach have been used by Khoshnevis and Chen [2] to enhance the shop floor performance. Consequently, a feedback mechanism has been introduced for effective coordination between various resources. Khoshnevis and Chen [3] addressed the issues that has been involved in integration approach of manufacturing system functions, to resolve this issues a heuristic approach has been developed and found the potential impact of the integrated system performance has been achieved by reducing the number of late parts, total tardiness, and flow times.

Considering the above features and needs of networked manufacturing, multi-objective evolutionary algorithms (MOEAs) have captured the interest of the number of researchers. Li et al. [4] proposed a mathematical model for multi-objective problem and implemented the integration approach for manufacturing functions. With the help of an evolutionary-algorithm and the integration concept the functions has been optimized. Consequently, with different experimental studies the feasibility and performance of the proposed approach has been verified. Chaube et al. [5] adapted a multi-objective GA (NSGA-II) to generate the optimal process plans in reconfigurable manufacturing system. Here, the allocation of operations on various machines and scheduling of different part types has been performed to find the total completion time of the product and total manufacturing cost.

The remainder of this paper is organized as follows. In section 2, we describe the problem and its notations. In Section 3, we developed an integration approach Section 4 explains the experimentation with different cases and their results. In Section 5 the results and their discussions are detailed. The paper concludes with section 6 which suggests the directions of future work.

2 Problem Description

A series of jobs on order has been submitted by different customers denoted as n. Each job consists of a set of strategies which corresponds to its alternative process plans and each process plan contains a series of sequential operations. Consequently, the jobs with alternative process plans are processed for different operations on a set of alternative machines. However, in the networked manufacturing the machines are geographically distributive which can perform different operations of the jobs; this can be one of the complex tasks of the present problem. Although, the transportation time between two corresponding machines acts as a crucial role for process planning and scheduling tasks. For large scale problems it is difficult to find perfect solutions in a reasonable time. Due to flexibility in networked manufacturing, the integration approach has the potential to generate the near optimal process plans.

Assumptions:

(1) Job Pre-emption is not allowed;
(2) When an operation of a job is being processed on a machine, it cannot be interrupted until finished;

(3) Each machine can handle only one job at a time;
(4) Transportation time is considered. The system is designed in such a way that, after an immediate completion of the operation of a job on a machine, the job is immediately transported to the succeeding machine on its process;
(5) All jobs and machines are simultaneously available at the time zero.

Based on the above mentioned problem with several assumptions the proposed model and its mathematical model are represented:

$$X_{jp} = \begin{cases} 1 & \text{the } p^{th} \text{ flexible process plan of job } j \text{ is selected} \\ 0 & \text{otherwise} \end{cases}$$

$$Y_{jopQrsm} = \begin{cases} 1 & \text{the operation } U_{jop} \text{ precedes the operation } U_{Qrs} \text{ on machine } m \\ 0 & \text{otherwise} \end{cases}$$

Objectives:

(1) Maximum time required to complete all the jobs:

$$Min\ makespan \quad T_j = Max\ C_{jopm}$$
$$j \in [1, N],\ o \in [1, Q_{jp}],\ p \in [1, G_j],\ m \in [1, K] \tag{1}$$

(2) Maximization of machine utilization:

$$Max\ machine\ utilization = Max\ mu = \frac{\sum_{k=1}^{m} mpt_k}{\sum_{k=1}^{m} (mct_k - mst_k)} \tag{2}$$

Subject to:

(1) For the first operation in the alternative process plan p of job j:

$$C_{j1pm} + A(1 - X_{jp}) \geq Pt_{j1pm}$$
$$j \in [1, N],\ p \in [1, G_j],\ m \in [1, K] \tag{3}$$

(2) For the last operation in the alternative process plan p of job j:

$$C_{jQ_{jp}pm} - A(1 - X_{jp}) \leq C_{jopm}$$
$$j \in [1, N],\ p \in [1, G_j],\ m \in [1, K] \tag{4}$$

(3) The different operations of one job can not be processed simultaniously:

$$C_{jopm} - C_{j(o-1)pm_1} + A(1 - X_{jp}) \geq Pt_{jopm}$$
$$j \in [1, N],\ o \in [1, Q_{jp}],\ p \in [1, G_j],\ m, m_1 \in [1, K] \tag{5}$$

(4) Each machine can handle only one job at a time:

$$C_{jopm} - C_{Qrsk} + A(1 - X_q) + AY_{jopQrsm} \geq Pt_{jopm}$$
$$j, Q \in [1, N],\ r \in [1, Q_{jprs}],\ o, p, S \in [1, G_{j,p}],\ m \in [1, K] \tag{6}$$

(5) Only one alternative process plan can be selected of job j:

$$\sum_{1} X_{jp} = 1$$
$$p \in [1, G_j] \tag{7}$$

(6) Each machine can only process one operation at a time:

$$\sum_{m=1}^{k} Z_{jopm} = 1$$
$$j \in [1, N],\ o \in [1, Q_{jp}],\ p \in [1, G_j] \tag{8}$$

3 Territory Defining Evolutionary Algorithm

One of the significant decisions in choosing metaheuristic lies in deciding how to represent the solutions and relate them in an efficient way to the searching space. Representation of input parameters to the operators encoding scheme should be easy to decode so that the cost of the algorithm has been reduced. The problem complexity and the representation of the presented algorithms to the proposed problem are stated in below sections. The flowchart of the territory defining evolutionary algorithm is depicted in figure below.

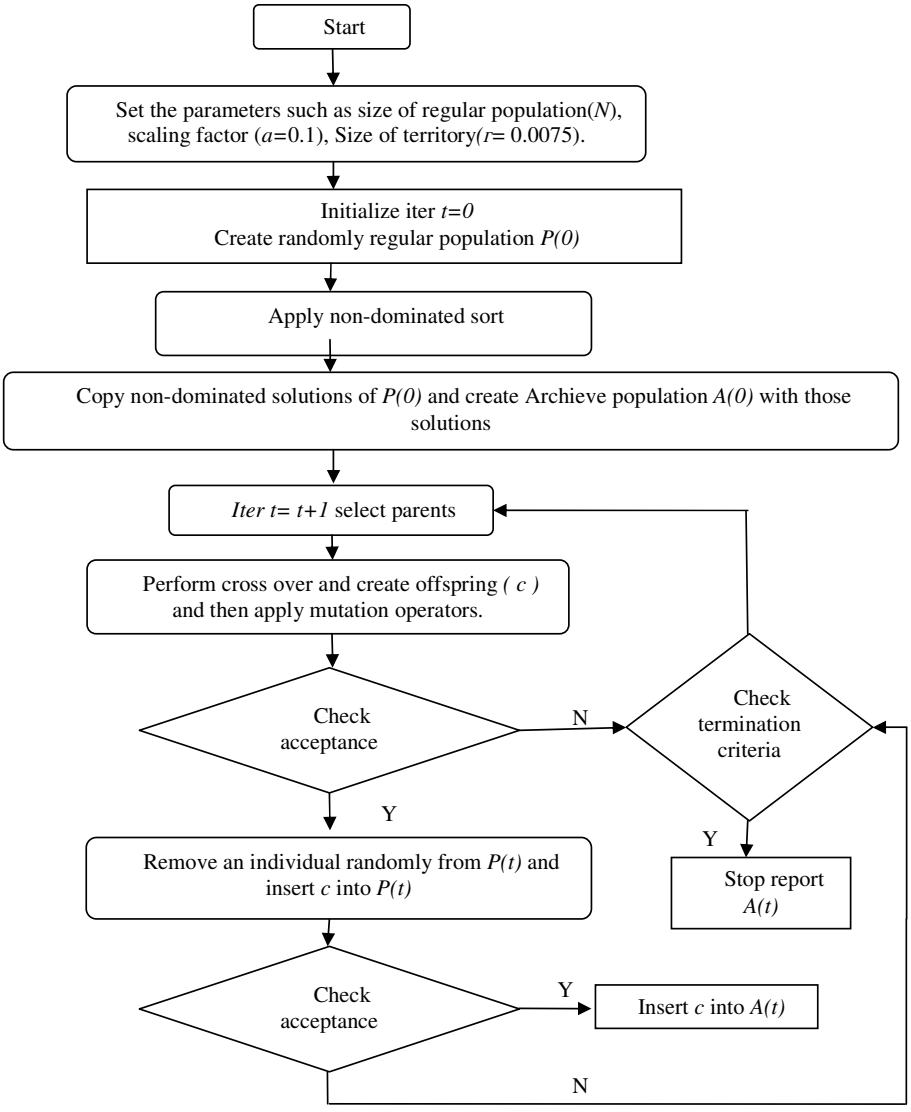


Fig. 1. Flowchart of TDEA

To maintain the diversity among all the population it is necessary to assign better τ value. In general the value of τ ranges between *0* and *1*, whereas the value of τ is small, larger will be the population size. In this paper, we define τ value as 0.075 by considering the number of objectives as our concern.

4 Illustrative Example

To illustrate the effectiveness and performance of the algorithm proposed in this paper, three representative instances (represented by problem $n \times m$) based on practical data have been selected as a test bed to compute. Three problem instances (problem 6×6, and problem 8×8) are taken from Zhou et al. [6] and Shao et al. [7] is used here. We have applied and compared two meta-heuristics CE-NSGA-II, and TDEA in networked manufacturing problems.

Table 1. The Makespan values, process plans and job schedules for 6×6 problem

Job id	TDEA (40th generation)			ControlledElitist-NSGA-II (46th generation)		
	Makespan value	Process plan	Job Scheduling	Makespan value	Process plan	Job Scheduling
1	33	2	[3,2,3,5]	29	2	[1,2,3,5]
2	35	1	[2,3,4,3,4,4]	33	1	[2,3,2,3,4,4]
3	34	1	[3,4,5,3]	33	1	[3,4,2,6]
4	28	2	[1,2,3,5]	32	2	[3,2,4,5]
5	32	2	[2,5,6]	28	1	[1,2,3,5,6]
6	24	3	[1,4,6]	33	3	[1,4,3]

Table 2. The Makespan values, process plans and job schedules for 8×8 problem

Job id	TDEA (39th generation)			ControlledElitist-NSGAII (37nd generation)		
	Makespan value	Process plan	Job Scheduling	Makespan value	Process plan	Job Scheduling
1	23	1	[7,5,6]	26	2	[3,2,3,5]
2	35	2	[1,4,4,7,4,1,4]	32	2	[1,4,4,4,6,8,4]
3	31	2	[1,3,5]	32	2	[8,7,8]
4	34	2	[3,2,4,5]	32	2	[1,8,3,5]
5	32	2	[2,5,3]	29	1	[1,2,3,5]
6	33	2	[3,2,7,6]	26	3	[2,5,6]
7	36	2	[3,4,1,4,1]	33	2	[6,8,4,4,3]
8	33	2	[2,4,8]	32	2	[6,4,6]

5 Results and Discussion

In this paper we have chosen the two conflicting objectives as makespan and machine utilization. Makespan is the time taken to complete the processing of the products with the help of available resources. In order to find the process plans and scheduling plans for minimum makespan, we need to run the heuristic algorithms for multiple generations. The two algorithms have been coded in MATLAB software and the problem is tested on Intel® Core™2 Duo CPU T7250 @2.00GHz, 1.99 GB of RAM.

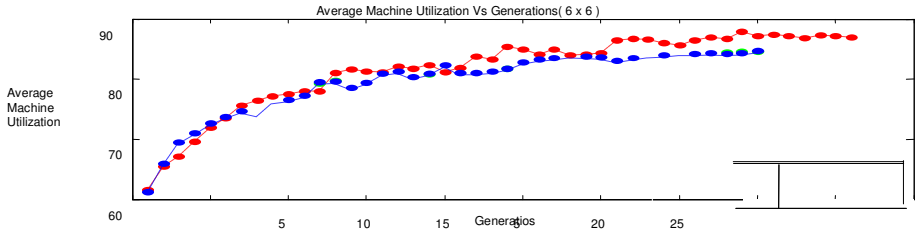


Fig. 2b. Convergence curves of three algorithms for average machine utilization

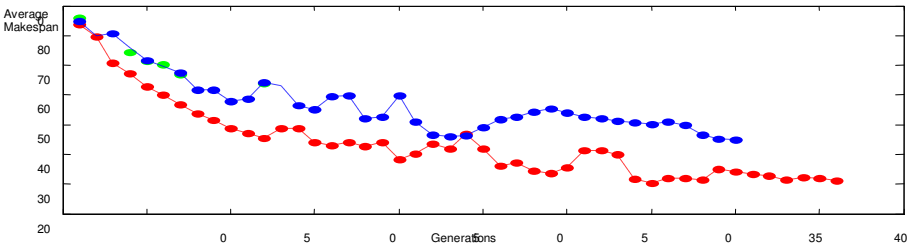


Fig. 2a. Convergence curves of three algorithms for average makespan

Finally, from the results we have realized that, CE-NSGA-II performs better for objective functions and for number of generations, whereas CE-NSGA-II performs better for computational time which is lower in time compared to TDEA algorithms.

The convergence curves shown in Fig. 2a and Fig. 2b illustrates about the average makespan and average machine utilization for 6×6 problem. Similarly, Fig. 3a and Fig. 3b shows the convergence and diversity of solutions of average makespan and average machine utilization for 8×8 problem. Consequently, these graphs also depict the number of generations the solutions have taken for convergence.

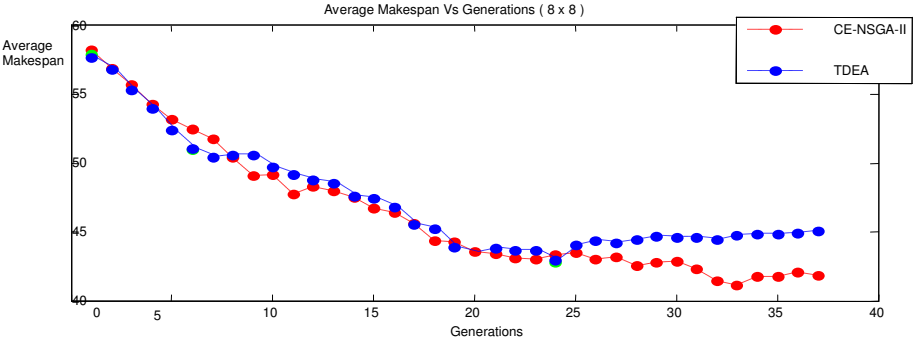


Fig. 3a. Convergence curves of three algorithms for average makespan

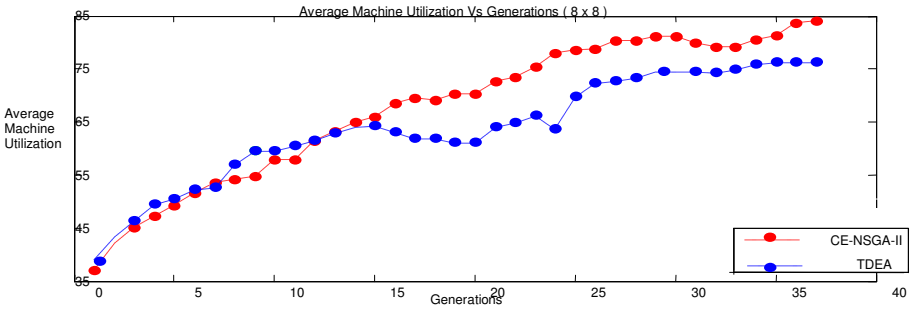


Fig. 3b. Convergence curves of three algorithms for average machine utilization

6 Conclusion

In this paper, we have developed a conceptual model to generate optimal process plans in the context of network based manufacturing environment. Illustrative example shows that CE-NSGA-II algorithm has shown the better synthetic performance in obtaining the near optimal solutions than another algorithm. Consequently, it has shown better computational time and significantly reduces the solution time needed to find the near optimal solutions. As an extension of this research, we plan to evaluate some more performance measures for more practical network based manufacturing problems.

References

1. Chryssolouris, G., Chan, S.: An integrated approach to process planning and scheduling. Annals of the CIRP 34(1), 413–417 (1985)
2. Khoshnevis, B., Chen, Q.M.: Integration of process planning and scheduling functions. Jour. of Inte. Manuf. 1, 165–176 (1990)
3. Khoshnevis, B., Chen, Q.M.: Integration of process planning and scheduling function. In: Proceedings of IIE Integrated Systems Conference and Society for Integrated Manufacturing Conference, pp. 415–420 (1989)
4. Li, W.D., McMahon, C.A.: A simulated annealing-based optimization approach for integrated process planning and scheduling. Int. Jour. of Comput. Integr. Manuf. 20, 80–95 (2007)
5. Chaube, A., Benyoucef, L., Tiwari, M.K.: An adapted NSGA-2 algorithm based dynamic process plan generation for reconfigurable manufacturing system. Journal of Intelligent Manufacturing (2010)
6. Zhou, G.H., Xiao, Z., Jiang, P.Y., Huang, G.Q.: A game-theoretic approach to generating optimal process plans of multiple jobs in networked manufacturing. International Journal of Comput. Integr. Manuf. 23, 1118–1132 (2010)
7. Shao, X.Y., Li, X.Y., Gao, L., Zhang, C.Y.: Integration of process planning and scheduling–a modified genetic algorithm based approach. Comput. and Opera. Res. 36, 2082–2096 (2009)

Teaching Learning Based Optimization for Neural Networks Learning Enhancement

Suresh Chandra Satapathy[1], Anima Naik[2], and K. Parvathi[3]

[1] ANITS, Vishakapatnam
sureshsatapathy@ieee.org
[2] MITS, Rayagada, India
animanaik@gmail.com
[3] CUTM, Paralakhemundi
Kparvati16@gmail.com

Abstract. Evolutionary computation is a collection of algorithms based on the evolution of a population towards a solution of certain problem. These algorithms can be used successfully in many applications requiring the optimization. These algorithms have been widely used to optimize the learning mechanism of classifiers, particularly on Artificial Neural Network (ANN) Classifier. Major disadvantages of ANN classifier are due to its slow convergence and always being trapped at the local minima. To overcome this problem, TLBO (Teaching learning based optimization) has been used to determine optimal value for learning mechanism. In this study, TLBO is chosen and applied to feed forward neural network to enhance the learning process. Two programs have developed, Differential Evolution Neural Network (DENN) and Particle Swarm Optimization with Neural Network (PSONN) to probe the impact of these methods on a Teaching learning based optimization with neural network (TLBONN) learning using various datasets. The results have revealed that TLBONN has given quite promising results in terms of smaller errors compared to PSONN and DENN.

Keywords: Evolutionary computation, Swarm Optimization, Artificial Neural Network.

1 Introduction

Teaching learning based optimization (TLBO) algorithm is an evolutionary algorithm, which was proposed by R.V. Rao in 2011. It is a small and simple mathematical model of a big and naturally complex process of evolution. So, it is easy and efficient. According to [1]-[11] this algorithm is simple and one of the most powerful tools with very fast convergence characteristics for global optimization. In this paper, TLBO is chosen and applied to feed forward neural network to enhance the learning process and compare with Differential Evolution Neural Network (DENN) and Particle Swarm Optimization with Neural Network (PSONN) to probe the impact of these methods on a Teaching learning based optimization with neural network (TLBONN) learning using various datasets.

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 761–769, 2012.

This paper is organized as follows. Section 2 presents a briefly introduction to artificial neural network (ANN), particle swarm optimization (PSO), differential evolution (DE) and teaching learning based optimization (TLBO). The proposed NNTLBO algorithm with NNPSO and NNDE is discussed in Section 3. Simulation results are provided in Section 4 to demonstrate the effectiveness and potential of the new proposed hybrid algorithm. Finally conclusions are included in Section 5.

2 ANN, PSO, DE , TLBO

The major discussion of this section is on Artificial Neural Network (ANN), Particle Swarm Optimization (PSO), Differential Evolution (DE) and Teaching Learning based Optimization (TLBO).

2.1 Artificial Neural Network

Artificial Neural Network (ANN) is the most popular supervised learning technique. In ANN, there are many elements to be considered such as number of input, hidden and output nodes, learning rate, momentum rate, bias, minimum error and activation/transfer function. These elements will affect the convergence of BP learning. The learning consists of the following steps:

1. An input vector is presented at the input layer.
2. A set of desired output is presented at the output layer.
3. After a forward pass is done, the errors between the desired and actual output are compared.
4. The comparison results are used to determine weight changes (backwards) according to the learning rules.

For proper understanding of PSO, DE and TLBO evolutionary optimization techniques, we request to go through papers [15], [16] and [1] respectively.

3 Neural Network Structure for DENN, PSONN and TLBONN

Here weights modification of neural network had been detained by PSO, DE and TLBO and it is thought to be an alternative to BP methods because of their convenience. These algorithms applied with feed forward neural network. For all algorithms, 3-layer ANN is used for classification on this study for all datasets. The network architecture consists of input layer, hidden layer and output layer. The total number of nodes for every layer is different depending on the classification problem. Number of input layer and output layer usually come from number of attribute and class attribute. However there is no appropriate standard rule or theory to determine the optimal number of hidden nodes [12]. There are many suggestions by researcher to determine the suitable number of hidden node. Some suggested techniques are summarized as follows:

1. The number of hidden nodes should be in the range between the size of the input layer and the size of the output layer.
2. The number of hidden nodes should be 2/3 of the input layer size, plus the size of the output layer.
3. The number of hidden nodes should be less than twice the input layer size.
4. Pyramidal shape topology [13].
5. One hidden layer and 2N+1 hidden neurons sufficient for N inputs [13].
6. The number of hidden nodes is selected either arbitrarily or based on trial and error approaches [14].
7. The number of hidden nodes should be n*m where m is the number of input nodes and n is number output nodes [14].

In this study, point 5 and 7 has been used to determine number of hidden node. The dimension of vector in case of DE, or particle in case of PSO or learner in case of TLBO can be calculated using the formula

$$\text{Dimension} = (\text{input}+1)*\text{hidden} + (\text{hidden}+1)*\text{output} \quad (1)$$

and the activation function used to calculate output for each neuron except input neuron. A sinusoidal function is used instead of sigmoid activation function because it leads to what has been termed a "Generalized Fourier Analysis". The sine function takes the trigonometric sine of the input and cosine function takes the trigonometric cosine of the input. Consider a back propagation network with just one output, the learning procedure can be thought of as synthesizing a continuous function $y = g(x)$ by showing it a discrete set of (x, y) pairs. The network configures itself to output a correct value (desired output) for each example input pair. When a previously unseen input pattern is presented to the network, the network in effect performs a non-linear interpolation and produce an output which a reasonable function value. When a sine or cosine function is used instead of a sigmoid, the learning procedure seems to perform a mode decomposition where it discovers the most important frequency components of the function described by discrete set of input output examples . The expression of this function is described as follows:

$$f(net) = \mathrm{h}(net) \quad (2)$$

where h is sine of cosine function and net is net input in a neuron.

A particle is a complete set of weights. The architecture of the FNN is constant for each particle. A particle's fitness is calculated in the following way. An artificial neural network is set up by using the particle's weights in the FNN architecture. For each sample of the supervised training data, forward propagation of input through the neural network is done to obtain the output and the error of this output with the desired output is calculated. The error values of all training samples are accumulated after squaring and used as the particle's fitness. Higher the error, lower the fitness of a particle. A forward propagation through the network is a computationally expensive task, so the aim is to find the best possible solution using a limited number of forward propagations through the network.

For all problems the initial weights are randomly assigned within a range [0, 1] . The training accuracy of each FNN is measured in the terms of the Root Mean Squared Error (RMSE) according to the following equation:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{p}\sum_{j=1}^{m}(T_{i,j}-F_{i,j})^2}{p*m}} \tag{3}$$

Where p denotes the number of training patterns, m the number of FNN outputs, $T_{i,j}$ the target (or desired) value and $F_{i,j}$ the actual value of the output. The index for patterns is i and that for output is j. For all problems the neural network had one input layer, one hidden layer and one output layer.

4 Experiment and Result

A. Experimental Setup

In all experiments in this section, the values of the common parameters used in each algorithm such as population size and total evaluation number were chosen to be the same. Population size was 50 and the maximum number iteration was 100 for all functions. The other specific parameters of algorithms are given below:

PSO Settings: Cognitive and social componentsc_1,c_2 are constants that can be used to change the weighting between personal and population experience, respectively. In our experiments cognitive and social components were both set to 2. Inertia weight, which determines how the previous velocity of the particle influences the velocity in the next iteration, was 0.5.

DE Settings: In DE, F is a real constant which affects the differential variation between two Solutions and set to F = 0.5*(1+ rand (0, 1)) where rand (0, 1) is a uniformly distributed random number within the range [0, 1] in our experiments. Value of crossover rate, which controls the change of the diversity of the population, was chosen to be $R = (R_{max} - R_{min}) * (MAXIT - iter)/MAXIT$where R_{max}=1 and R_{min}=0.5 are the maximum and minimum values of scale factor R, iter is the current iteration number and MAXIT is the maximum number of allowable iterations.

TLBO Settings: For TLBO there is no such constant to set.

B. Datasets used

The neural network is tested on five different datasets. These are as follows:

The 2 bit parity – This is a famous nonlinear benchmark which is defined by 4 patterns of 2 inputs and one output. The output is set to one if the number of input bits set to 1 is odd and set to zero otherwise. This is also known as the XOR problem.

The 4 bit parity – This is similar to the 2 bit parity problem. It is defined by 16 patterns of 4 inputs and one output. The output is set to one if the number of input bits set to 1 is odd and set to zero otherwise.

The iris data set - The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; and these other 2 are not linearly separable from each other. There are a total of 150 patterns and 4 input and 3 outputs (one for each class).

The lenses data set – This task consists of determining whether a patient should be given hard contact lenses, soft contact lenses or no contact lenses. There are a total of 24 patterns of 4 input and 3 outputs.

The Haberman's Survival data set – the data set contains 2 classes of 306 instances. So there are a total of 306 patterns of 3 input and 2 outputs.

C. Simulation Strategy

In this paper, while comparing the performance of algorithms, we focus on training accuracy is measured in the terms of the Root Mean Squared Error (RMSE) and convergence characteristics in terms of no.of fitness evaluation . Since the algorithms are stochastic in nature, the results of two successive runs usually do not match. Hence, we have taken 30 independent runs (with different seeds of the random number generator) of each algorithm. The results have been stated in terms of the mean values and standard deviations over the 30 runs in each case.

Finally, we would like to point out that all the experiment codes are implemented in MATLAB. The experiments are conducted on a Pentium 4, 1GB memory desktop in Windows XP 2002 environment.

D. Experimental Results

To judge the accuracy of the PSONN, DENN and TLBONN, we let each of them run for a very long time over every benchmark data set, until the number of FEs exceeded 5000. Then, we note the result as found in table 1 using sine activation function and in table 2 using cosine activation function. At the same time we have noted the RMSE value in different FEs in table 3 and 4 for sine activation function and in table 5 and 6 for cosine activation function.

Table 1. Results of the experiments for NN training (averaged over 30 runs) using SINE function

Dataset	PSONN		DENN		TLBONN	
	Mean	Std	Mean	Std	Mean	Std
2 bit parity	0.0053	0.0055	1.4923e-4	2.1642e-4	1.0063e-7	1.4024e-7
4 bit parity	0.0012	0.0017	1.4083e-4	2.3486e-4	1.0857e-7	1.4963e-7
iris data	8.3371e-5	2.1940e-6	1.9400e-4	1.5993e-4	8.6331e-6	8.5031e-6
lenses data	5.2319e-4	2.2008e-6	0.0013	0.0012	3.1168e-5	1.9736e-5
Survival ata	2.1759e-5	1.5728e-5	1.2067e-5	2.9850e-5	5.4443e-8	6.5821e-8

Table 2. Results of the experiments for NN training (averaged over 30 runs) using COSINE function

Dataset	PSONN		DENN		TLBONN	
	Mean	Std	Mean	Std	Mean	Std
2 bit parity	0.0104	0.0110	1.6254e-4	3.2051e-4	2.0925e-7	3.9495e-7
4 bit parity	8.8731e-4	0.0012	6.4093e-5	8.4232e-5	1.2081e-7	1.3946e-7
iris data	6.6081e-4	5.0269e-4	1.3935e-5	1.0141e-5	1.2506e-6	1.0034e-6
lenses data	0.0044	0.0032	0.0010	0.0010	3.3741e-5	2.9101e-5
Survival ta	5.8250e-5	5.0471e-5	7.5927e-6	9.8357e-6	3.5133e-8	4.4424e-8

Table 3. RMSE value in corresponding FEs using SINE activation function

No. FEs	2bit parity dataset			4bit parity dataset			Iris dataset		
	PSO	DE	TLBO	PSO	DE	TLBO	PSO	DE	TLBO
1	0.0021	0.0016	1.9985e-4	0.0047	0.0013	2.7405e-5	5.4161e-4	4.3411e-4	5.1125e-4
250	0.0021	0.0016	1.3000e-4	0.0047	5.3945e-4	1.3521e-5	5.4161e-4	4.3411e-4	3.5112e-4
500	0.0021	0.0016	2.7668e-5	0.0047	5.3945e-4	3.8312e-6	5.4161e-4	3.8558e-4	1.1126e-4
1000	0.0021	0.0016	2.7668e-5	0.0047	3.6125e-4	2.6865e-7	5.4161e-4	3.8558e-4	7.6619e-5
1500	0.0021	1.0232e-4	5.9518e-6	0.0047	2.2348e-4	2.6865e-7	5.4161e-4	2.5650e-4	5.1103e-5
2000	0.0021	4.7050e-5	3.3312e-6	0.0047	2.2348e-4	2.6865e-7	5.4161e-4	2.5650e-4	5.1103e-5
2500	0.0021	4.7050e-5	2.0012e-7	0.0047	2.2348e-4	1.4360e-7	5.4161e-4	2.5650e-4	5.1123e-6
3000	0.0021	4.7050e-5	1.2312e-7	0.0047	2.2348e-4	1.4360e-7	5.4161e-4	2.5650e-4	5.1123e-6
3500	0.0021	4.7050e-5	1.2312e-7	0.0047	2.2348e-4	1.4360e-7	5.4161e-4	2.5650e-4	2.1126e-6
4000	0.0021	4.7050e-5	1.0001e-7	0.0047	2.2348e-4	1.0092e-7	5.4161e-4	2.5650e-4	2.1126e-6
4500	0.0021	4.7050e-5	1.0001e-7	0.0047	2.2348e-4	1.0092e-7	5.4161e-4	2.5650e-4	1.0019e-6
5000	0.0021	4.7050e-5	1.0001e-7	0.0047	2.2348e-4	1.0092e-7	5.4161e-4	2.5650e-4	1.0019e-6

Table 4. RMSE value in corresponding FEs using SINE activation function

No. of FEs	Haberman's survival			Lenses		
	PSO	DE	TLBO	PSO	DE	TLBO
1	3.3374e-5	1.2763e-6	2.0612e-4	5.2797e-4	5.3316e-4	0.0017
250	3.3374e-5	3.0981e-5	1.6625e-4	5.2414e-4	5.3075e-4	8.6446e-4
500	3.3374e-5	3.0981e-5	5.6624e-5	5.2414e-4	5.2506e-4	5.6979e-5
1000	3.3374e-5	9.7380e-6	2.3312e-5	5.2414e-4	5.2506e-4	5.6979e-5
1500	3.3374e-5	9.7380e-6	2.1991e-7	5.2363e-4	5.2364e-4	2.1630e-5
2000	3.3374e-5	9.7380e-6	5.1126e-8	5.2357e-5	5.2358e-4	2.1630e-5
2500	3.3374e-5	9.7380e-6	1.5107e-8	5.2357e-5	5.2357e-4	1.6651e-5
3000	3.3374e-5	9.7380e-6	1.5907e-9	5.2357e-5	5.2357e-4	6.5612e-6
3500	3.3374e-5	9.7380e-6	1.5907e-9	5.2357e-5	5.2357e-4	2.1162e-6
4000	3.3374e-5	9.7380e-6	1.5907e-9	5.2357e-5	5.2357e-4	2.1162e-6
4500	3.3374e-5	9.7380e-6	1.5907e-9	5.2357e-5	5.2357e-4	2.1162e-6
5000	3.3374e-5	9.7380e-6	1.5907e-9	5.2357e-5	5.2357e-4	2.1162e-6

Table 5. RMSE value in corresponding FEs using COSINE activation function

No. FEs	2bit parity dataset			4bit parity dataset			Iris dataset		
	PSO	DE	TLBO	PSO	DE	TLBO	PSO	DE	TLBO
1	0.0038	5.3822e-4	2.3257e-5	1.4520e-4	2.3121e-5	1.9195e-4	4.4795e-4	1.1165e-4	2.1124e-4
250	0.0038	5.3822e-4	2.3257e-5	1.4520e-4	2.3121e-5	5.1588e-5	4.4795e-4	1.1165e-4	2.1124e-4
500	0.0038	5.3822e-4	2.3257e-5	1.4520e-4	2.3121e-5	7.0456e-6	4.4795e-4	1.1165e-4	2.1124e-4
1000	0.0038	1.9971e-4	1.7198e-5	1.4520e-4	2.3121e-5	4.1669e-6	4.4795e-4	3.4132e-5	1.0560e-5
1500	0.0038	6.6203e-5	8.4330e-6	1.4520e-4	2.3121e-5	4.1669e-6	4.4795e-4	3.4132e-5	1.0560e-5
2000	0.0038	6.6203e-5	3.1771e-6	1.4520e-4	2.3121e-5	4.1669e-6	4.4795e-4	3.4132e-5	1.0560e-5
2500	0.0038	6.6203e-5	1.0184e-7	1.4520e-4	2.3121e-5	1.2204e-7	4.4795e-4	3.4132e-5	7.8669e-6
3000	0.0038	6.6203e-5	1.0441e-8	1.4520e-4	2.3121e-5	1.2204e-7	4.4795e-4	3.4132e-5	1.8769e-6
3500	0.0038	6.6203e-5	6.4876e-9	1.4520e-4	2.3121e-5	1.2204e-7	4.4795e-4	3.4132e-5	1.2506e-6
4000	0.0038	6.6203e-5	6.4876e-9	1.4520e-4	2.3121e-5	6.9832e-8	4.4795e-4	3.4132e-5	1.2506e-6
4500	0.0038	6.6203e-5	6.4876e-9	1.4520e-4	2.3121e-5	6.9832e-8	4.4795e-4	3.4132e-5	1.2506e-6
5000	0.0038	6.6203e-5	6.4876e-9	1.4520e-4	2.3121e-5	6.9832e-8	4.4795e-4	3.4132e-5	1.2506e-6

Table 6. RMSE value in corresponding FEs using COSINE activation function

No. of FEs	Haberman's survival			Lenses		
	PSO	DE	TLBO	PSO	DE	TLBO
1	2.6675 e-5	8.2352e-5	8.0296e-6	0.0088	0.0021	2.8358-4
250	2.6675 e-5	1.7840e-5	3.6534e-6	0.0088	0.0021	1.3000e-4
500	2.6675 e-5	1.5680e-5	3.6534e-6	0.0088	0.0021	8.7104e-5
1000	2.6675 e-5	1.5680e-5	4.1600e-7	0.0088	0.0021	8.7104e-5
1500	2.6675 e-5	1.5680e-5	2.6901e-7	0.0088	2.8519-4	8.7104e-5
2000	2.6675 e-5	8.1262e-5	9.0791e-8	0.0088	0.0014	8.7104e-5
2500	2.6675 e-5	8.1262e-5	7.4727e-8	0.0088	0.0014	8.7104e-5
3000	2.6675 e-5	8.1262e-5	7.4727e-8	0.0088	0.0014	1.4521e-5
3500	2.6675 e-5	8.1262e-5	7.4724e-8	0.0088	0.0014	1.4521e-5
4000	2.6675 e-5	8.1262e-5	7.4724e-8	0.0088	0.0014	1.4521e-5
4500	2.6675 e-5	8.1262e-5	7.4724e-8	0.0088	0.0014	1.4521e-5
5000	2.6675 e-5	8.1262e-5	7.4724e-8	0.0088	0.0014	1.4521e-5

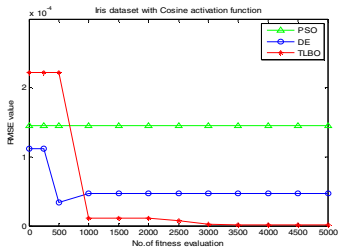


Fig. 1.

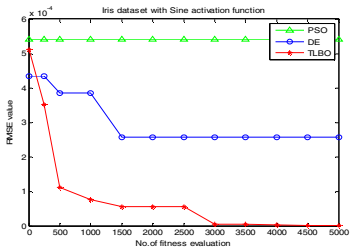


Fig. 2.

From table 1 and 2, we get that in all datasets the RMSE error for TLBONN is less than the RMSE error for both PSONN and DENN.

From table 3,4, 5 and 6 it is clear that convergence rate of TLBONN is slower in compared to PSONN and DENN but TLBONN is giving smaller error than PSONN and DENN in all the datasets. It is also marked that in how much no. of fitness evaluation other algorithms converges at the same no. of fitness evaluation also TLBONN is giving better result than other algorithm in almost all cases.

Due to limited space we have taken only two figure to show the convergence behavior of all algorithms.. Fig. 1 shows for iris dataset with cosine activation function and fig. 2 for iris dataset with sine activation function.

5 Conclusion

This paper investigated the application of a new optimization algorithm known as teaching learning based optimization (TLBO) to enhance the learning in neural network. The TLBONN approach was compared against PSONN and DENN. From

the simulation results it is observed that TLBONN has better training accuracy in compared to other two algorithms. As further study some modification of TLBO can be done to improve the convergence characteristics.

References

1. Rao, R.V., Savsani, V.J., Vakharia, D.P.: Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. Computer-Aided Design 43, 303–315 (2011)
2. Rao, R.V., Patel, V.: An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems. International Journal of Industrial Engineering Computations 3(4), 535–560 (2012)
3. Rao, R.V., Savsani, V.J., Vakharia, D.P.: Teaching-Learning-Based Optimization: A optimization method for continuous non-linear large scale problems. Information Sciences 183(1), 1–15 (2012)
4. Satapathy, S.C., Naik, A.: Data Clustering Based on Teaching-Learning-Based Optimization. In: Panigrahi, B.K., Suganthan, P.N., Das, S., Satapathy, S.C. (eds.) SEMCCO 2011, Part II. LNCS, vol. 7077, pp. 148–156. Springer, Heidelberg (2011)
5. Krishnanand, K.R., Panigrahi, B.K., Rout, P.K., Mohapatra, A.: Application of Multi-Objective Teaching-Learning-Based Algorithm to an Economic Load Dispatch Problem with Incommensurable Objectives. In: Panigrahi, B.K., Suganthan, P.N., Das, S., Satapathy, S.C. (eds.) SEMCCO 2011, Part I. LNCS, vol. 7076, pp. 697–705. Springer, Heidelberg (2011)
6. Rao, R.V., Savsani, V.J.: Mechanical design optimization using advanced optimization techniques. Springer, London (2012)
7. Satapathy, S.C., Naik, A., Parvathi, K.: High dimensional real parameter optimization with teaching learning based optimization. International Journal of Industrial Engineering Computations (2012), doi:10.5267/j.ijiec.2012.06.001
8. Naik, A., Parvathi, K., Satapathy, S.C., Nayak, R., Panda, B.S.: QoS Multicast Routing Using Teaching Learning Based Optimization. In: Aswatha Kumar, M., Selvarani, R., Suresh Kumar, T.V. (eds.) Proceedings of ICAdC. AISC, vol. 174, pp. 49–55. Springer, Heidelberg (2013)
9. Satapathy, S.C., Naik, A., Parvathi, K.: 0-1 Integer Programming for Generation Maintenance Scheduling in Power Systems Based on Teaching Learning Based Optimization (TLBO). In: Parashar, M., Kaushik, D., Rana, O.F., Samtaney, R., Yang, Y., Zomaya, A. (eds.) IC3 2012. CCIS, vol. 306, pp. 53–63. Springer, Heidelberg (2012)
10. Naik, A., Satapathy, S.C., Parvathi, K.: Improvement of initial cluster center of c-means using Teaching learning based optimization, Accepted and will be published in Procedia Technology, Elsevier and indexed by Scopus
11. Naik, A., Satapathy, S.C.: Rough set and Teaching learning based optimization technique for Optimal Features Selection. Ref.: Ms. No. CEJCS-D-12-00042, under minor review in Central European Journal of Computer Science
12. Kim, G.-H., Yoon, J.-E., An, S.-H., Cho, H.-H., Kang, K.-I.: Neural Network Model Incorporating A Genetic Algorithm in Estimating Construction Cost. Building and Environment 39(11), 1333–1340 (2004)
13. Shamsuddin, S.M.: Number of Hidden Neurons, Universiti Teknologi Malaysia. Lecture Note Advanced Artificial Intelligence (2004)

14. Charytoniuk, W., Chen, M.S.: Neural Network Design for Short-term Load Forecasting. In: International Conference on Electric Utility Deregulation and Restructuring and Power Technologies 2000, City University, London, April 4-7, pp. 554–561 (2000)
15. Clerc, M., Kennedy, J.: The particle swarm-explosion, stability and convergence in a multi-dimensional complex space. IEEE Trans. on Evolutionary Computation 6, 58–78 (2002)
16. Storn, R., Price, K.: Differential evolution-A simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Optim. 11(4), 341–359 (1997)

Large Scale Optimization Based on Co-ordinated Bacterial Dynamics and Opposite Numbers

Jaydeep Ghosh Chowdhury, Aritra Chowdhury, and Arghya Sur

Department of Electronics and Telecommunication Engg.,
Jadavpur University,
Kolkata-700 032, India
{jaydeep197,arghyasur1991}@gmail.com,
arit0001@yahoo.co.in

Abstract. This work, named Large Scale Optimization based on co-ordinated Bacterial Dynamics and Opposite Numbers (LSCBO) presents a very fast algorithm to solve large scale optimization problems. The computational simplicity of the algorithm allows it to achieve admirable results. There are only three searching agents in the population, one being the primary bacterium and the other two are secondary bacteria. The proposed algorithm is employed on 7 benchmark functions of CEC2008 and it gives better results compared to the other well known contemporary algorithms present in the literature. The main reason for this is that the computational burden of the algorithm is significantly reduced.

Keywords: Evolutionary algorithms, large scale optimization, bacterial dynamics, quorum sensing.

1 Introduction

The main idea behind solving optimization problems is to maximize or minimize an objective function by choosing the values of input variables properly within a definite boundary and computing objective functional value. The domain of large scale optimization problems exists where the number of input variables is very high. There have been many attempts to perform this special kind of optimization using Evolutionary Algorithms (EAs). These methods are used to solve such high dimensional complex large scale optimization problems when classical optimization methods fail or are not adequate enough to perform the task efficiently. Due to the evolutionary nature of such methods, there is however a drawback which comes into being in certain cases. These algorithms may become computationally complex [1, 2]. Furthermore, the performance of EAs decreases if the dimensionality of optimization problems is increased. The optimization techniques which employ smart sampling and meta-modeling are some commonly used as shown in [3, 4] to solve such problems. The main reason for this is that such algorithms require the initialization of an input population of random solutions. As a result the complexity increases as the fitness of

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 770–777, 2012.

each solution needs to be calculated repeatedly. In our algorithm, we completely do away with the concept of population as shown later.

Existing EAs like Differential Evolution (DE) and Particle Swarm Optimization (PSO) have a drawback of high computational complexity [5,6]. As mentioned earlier, this is due to the large size of population used in these algorithms and the unnecessary computation undertaken by the individuals with poor fitness in large scale optimization problems.

In this paper, Large Scale Optimization based on co-ordinated Bacterial Dynamics and Opposite Numbers (LSCBO) is derived from the bacterial foraging pattern. There exist only three bacteria in the entire population, one primary bacterium and two secondary bacteria. The primary bacterium forms the searching path which leads to the global optima by evolutionary process. There are two secondary bacteria: one known as the associated bacterium and the other known as the opposite associated bacterium. These bacteria search randomly around the primary bacterium. If the secondary bacterium achieves a better fitness than the primary one, the primary bacteria gets a velocity towards the secondary bacterium which gives the better fitness.

Experiments have been performed on the benchmark functions of CEC'08 to test the efficiency of the proposed method and the experimental results show that the LSCBO has ability to achieve much better results than some recently developed EAs which also specialize in large scale optimization.

2 Opposition-Based Bacterial Dynamics- An outline

2.1 Dynamic Random Search

The primary bacterium performs a random search to obtain the associated bacteria. This phenomenon is detailed in this section. In bacterial dynamics the primary bacterium occupies a position in the *n* dimensional space. The primary bacterium at the k^{th} iteration may be represented as,

$$X^k = (x_1^k, x_2^k, x_3^k, ..., x_n^k) \in R^n, \tag{1}$$

where x_i^k is the component of the X^k on the i^{th} dimension at k^{th} iteration. Now an associated bacterium $\tilde{X}$ is generated from the primary bacterium by dynamic random search. $\tilde{X}$ is associated to X as a mutation of the primary bacterium. A dimension $l \in \{1,2,3,.....,n\}$ is randomly chosen for the mutation where *n* is the number of dimensions of the primary as well as the associated bacterium. Then at k^{th} iteration the associated bacterium is generated as

$$\tilde{X}^k = X^k + \delta D^k, \tag{2}$$

Where $\delta D^k = (0,0,0,.....,d_l^k,0,0,.....,0)$ and d_l^k is randomly chosen between the lower and upper boundary on the l^{th} dimension which is defined by

$$d_l^k = c_1 r_1 (UB - LB) \tag{3}$$

where r_1 is a random number chosen in the range between -1 to 1. So, $r_1 \epsilon$ *[-1, 1]. UB* and *LB* denote the upper and lower boundary respectively. There is a point to be noted that in our proposed algorithm we use same *l* when the primary bacterium improves in objective functional value, but it does not improve we choose a new *l* randomly form the set $\{1,2,3,.....,n\}$. Searching Co-efficient c_1 is a constant which plays a role equivalent to maintaining the diversity of the population, as adopted in conventional Evolutionary Algorithms like DA, PSO etc. in our proposed algorithm, c_1 is set at 0.23.

Similarly, the opposite associated bacterium $\tilde{X}^*$ is expressed as,

$$\tilde{X}^* = X^k - \delta D^k \quad . \tag{4}$$

2.2 Simplified Quorum Sensing

The fittest bacterium among the present primary bacterium, the associated bacterium and opposite associated bacterium is sensed by the method of simplified quorum sensing. The primary bacterium is forced to move to the position of the fittest among the associated bacteria. We have implemented the jumping of the bacterium by introducing a velocity to the primary bacterium.

The velocity of the primary bacterium is denoted as $V^k = (v_1^k, v_2^k, v_3^k, ..., v_n^k)$ where v_i^k is a component of the V^k on the i^{th} dimension at k^{th} iteration.

Let FP^k be the position of the fittest bacterium between the associated and opposite associated bacterium. Then the velocity of primary bacterium is updated at the k^{th} iteration as,

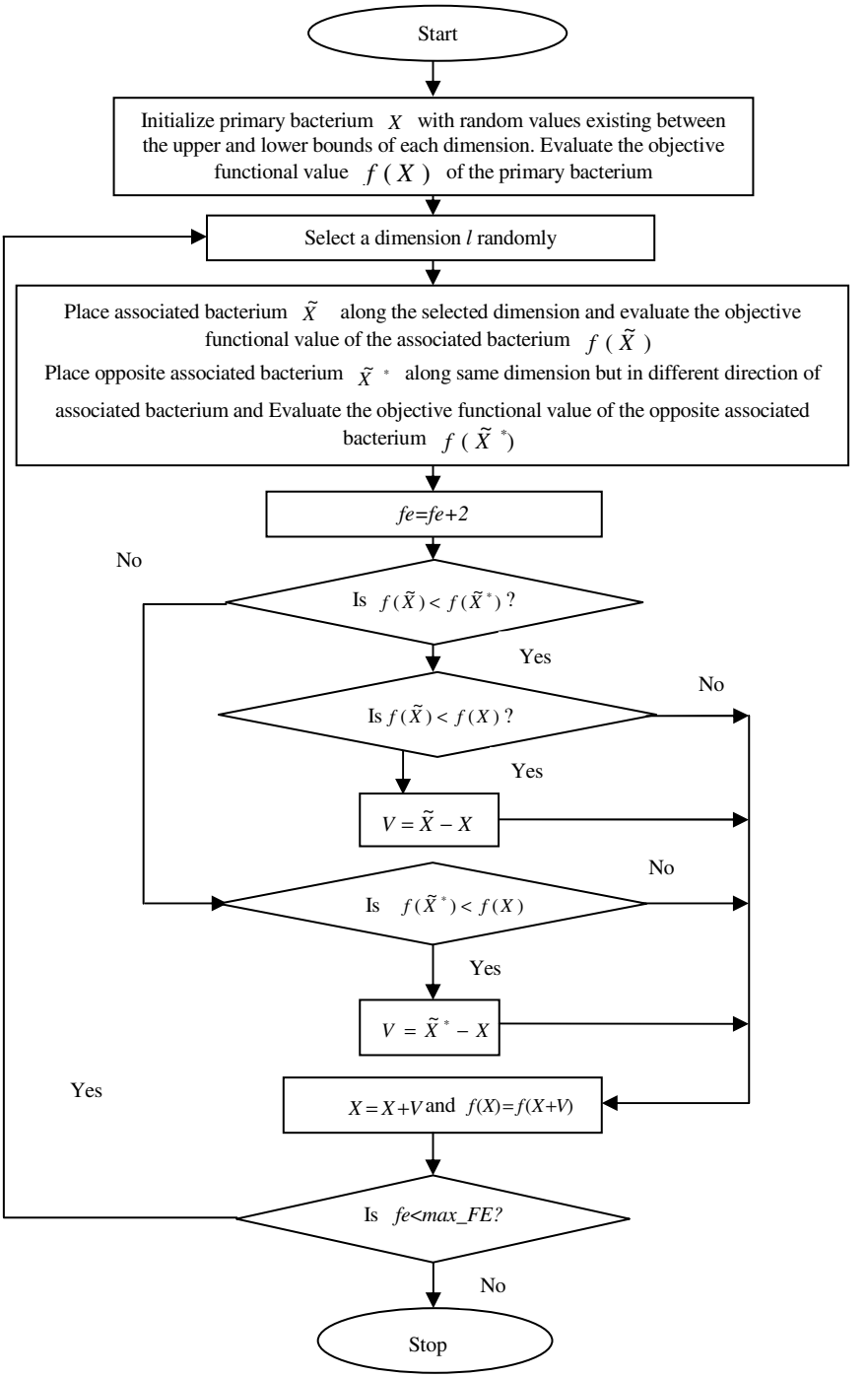
$$v_i^k = FP_i^k - x_i^k \tag{5}$$

where, FP_i^k and x_i^k are the components of the fittest position and primary bacterium on the i^{th} dimension at k^{th} iteration respectively. Thus, the position of the primary bacterium is updated as,

$$X^{k+1} = X^k + V^k \tag{6}$$

2.3 Flowchart of LSCBO

The flowchart of LSCBO is given as shown:


Start
Initialize primary bacterium X with random values existing between the upper and lower bounds of each dimension. Evaluate the objective functional value $f(X)$ of the primary bacterium
Select a dimension *l* randomly
Place associated bacterium $\tilde{X}$ along the selected dimension and evaluate the objective functional value of the associated bacterium $f(\tilde{X})$
Place opposite associated bacterium $\tilde{X}^*$ along same dimension but in different direction of associated bacterium and Evaluate the objective functional value of the opposite associated bacterium $f(\tilde{X}^*)$
fe=fe+2
No
Is $f(\tilde{X}) < f(\tilde{X}^*)$?
Yes
No
Is $f(\tilde{X}) < f(X)$?
Yes
$V = \tilde{X} - X$
No
Is $f(\tilde{X}^*) < f(X)$
Yes
$V = \tilde{X}^* - X$
Yes
$X = X + V$ and $f(X) = f(X+V)$
Is *fe<max_FE?*
No
Stop

3 Experimental Studies

In this section we describe test functions on which we have applied our proposed algorithm and then describe some existing EAs like CCPSO[7] and sep-CMA[8] which are used in our comparative studies.

3.1 Experimental Setup

We have applied our proposed OCBD algorithm in 7 benchmark functions provided at CEC'08 special session on large scale global optimization. The global optimum of each function is shifted with different values in each dimension.

In CEC'08 special session, proposed seven benchmark functions were

a) f_1*(Shifted Sphere)*
b) f_2*(Schwefel Problem)*
c) f_3*(Shifted Rosenbrock)*
d) f_4*(Shifted Rastrigin)*
e) f_5*(Shifted Griewank)*
f) f_6*(Shifted Ackley)*
g) f_7*(Fast Fractal)*

f_1, f_4 and f_7 are separable functions, and f_2, f_3, f_5 and f_7 *are* non-separable functions. These non-separable functions propose a formidable challenge to any EA. As these are large scale optimization problems, Shifted Griewank function f_5 becomes easier to optimize as the product components of this function becomes insignificant due to its high dimension and negligible variable interaction. We have compared the results of our proposed algorithm with other existing Evolutionary Algorithms like CCPSO sep-CMA-ES etc. by applying them on these benchmark functions.

We have tested our algorithm on the above seven benchmark functions of 100, 500 and 1000 dimensions. For each test functions, we have recorded the averaged results and standard deviations of 25 runs. For each run, the maximum number of fitness evaluations was set to 5000×n (where n is the number of dimension).

3.2 Results and Comparative Studies

Here, we how the opposition-based search and simplicity help to reach the global optimum. In the 3rd part of that section we compare the results of our proposed algorithm LSCBO with other EAs.

- **Effects of Opposition-based Search**:

The opposite associated bacterium is generated by moving the primary bacterium in the same dimension but in a direction opposite to the associated bacterium. The primary bacterium shifts to the position of whichever gives better fitness between the associated bacteria. Therefore, with the help of opposition-based search we can check the fitness value in both directions of a dimension of the primary bacterium. So if the associated bacterium does not give the better fitness, the opposite associated bacterium may give the better fitness in the opposite direction. Thus if we use the opposite associated bacterium along with associated bacterium we can achieve a better result.

- **Effect of simplicity**

The proposed method is very simple in sense of computational analysis. Due to its simplicity, the function is evaluated twice per iteration. So the number of function evaluation (FE) is 2 per iteration. As the number of MAX_FEs is fixed during a run, therefore our algorithm can be made to iterate more times than the other contemporary algorithms. The conventional evolutionary algorithms like DA, PSO, GA deal with population of searching agents. So they require a larger number of FEs per iteration. As a result their algorithm cannot be iterated so many times. Due to the large number of iterations as well as its simplicity, the primary bacterium can improve its fitness value and thus obtain a better and a faster result.

- **Comparison of OCBD with Other EAs**

The results CCPSO2 and sep-CMA-ES were examined on the seven benchmark functions provided at CEC'08. We have compared the results of LSCBO with these two algorithms on those 7 benchmark functions.

Table-1 shows the results of 7 test functions provided at CEC'08 on 100-D, 500-D and 1000-D. The results of CCPSO2 scaled very well on f_1 and f_6 compared to the results of LSCBO. But the LSCBO clearly outperforms the CCPSO2 on f_2, f_3, f_4 and f_7 from 100-D to 1000-D. The LSCBO performs well on f_5 at 100-D, but CCPSO2 outperforms it on 500-D and 1000-D. On the other hand, sep-CMA gives better results on f_1, f_3 and f_5 than LSCBO. But it is clear from the table LSCBO scaled much better on f_2, f_4, f_6 and f_7 compared to sep-CMA from 100-D to 1000-D.

Table 1. Results of CCPSO2, sep-CMA and LSCBO on test functions of 100-D, 500-D and 1000-D

Test function	Dimensions	CCPSO2	sep-CMA	LSCBO
F1	100	7.73e-14 (3.23e-14)	**9.02e-15(5.53e-15)**	1.64e-04(9.45e-05)
	500	3.00e-13 (7.96e-14)	**2.25e-14(6.10e-15)**	8.24e-04(1.93e-04)
	1000	5.18e-13 (9.16e-14)	**7.81e-15(1.52e-15)**	3.15e-03(3.65e-04)
F2	100	6.08e+00(7.83e+00)	2.31e+01(1.39e+01)	**4.64e-01(2.93e-01)**
	500	5.79e+01(4.21e+01)	2.12e+02(1.74e+01)	**2.91e+00(1.82e-01)**
	1000	7.82e+01(4.25e+01)	3.65e+02(9.02e+00)	**3.49e+00(3.26e-01)**
F3	100	4.23e+02(8.65e+02)	**4.31e+00(1.26e+01)**	2.49e+02(1.23e+01)
	500	7.24e+02(1.54e+02)	**2.93e+02(3.59e+01)**	4.23e+02(3.61e+01)
	1000	1.33e+03(2.63e+02)	**9.10e+02(4.54e+01)**	9.95e+02(7.23e+01)
F4	100	3.98e-02(1.99e-01)	2.78e+02(3.43e+01)	**7.92e-04(4.23e-05)**
	500	3.98e-02(1.99e-01)	2.18e+03(1.51e+02)	**2.95e-03(1.74e-04)**
	1000	1.99e-01(4.06e-01)	5.31e+03(2.48e+02)	**5.75e-03(3.48e-04)**
F5	100	3.45e-03(4.88e-03)	**2.96e-04(1.48e-03)**	3.12e-03(3.24e-04)
	500	1.18e-03(4.61e-03)	**7.88e-04(2.82e-03)**	3.98e-03(2.95e-04)
	1000	1.18e-03(3.27e-03)	**3.94e-04(1.97e-03)**	6.42e-03(4.23e-04)
F6	100	**1.44e-13(3.06e-14)**	2.12e+01(4.02e-01)	5.12e-03(4.17e-04)
	500	**5.34e-13(8.61e-14)**	2.15e+01(3.10e-01)	1.87e-02(2.41e-03)
	1000	**1.02e-12(1.68e-13)**	2.15e+01(3.19e-01)	3.42e-03(3.69e-04)
F7	100	-1.50e+03(1.04e+01)	-1.39e+03(2.64e+01)	**-1.57e+03(9.26e+00)**
	500	-7.23e+03(4.61e+01)	-6.37e+03(7.59e+01)	**-8.28e+03(1.23e+01)**
	1000	-1.43e+04(8.27e+01)	-1.25e+04(9.36e+01)	**-1.50e+04(7.29e+01)**

In f_3, the LSCBO converges prematurely as soon as run started and gives a worse result than sep-CMA, but performs well compared to CCPSO2. In f_4 and f_5 also, LSCBO converges very fast and is seen to scale very well. In f_2, LSCBO does not converge very fast, but gives better result gradually.

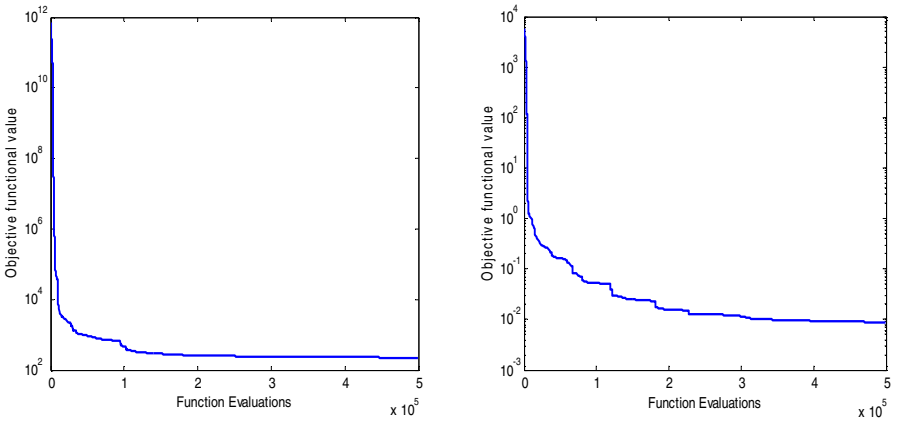


Fig. 1. Average objective functional values of OCBD on functions of CEC'08 of 100 dimensions. (a)*f_3(Shifted Rosenbrock)*, (b)*f_5(Shifted Griewank)*

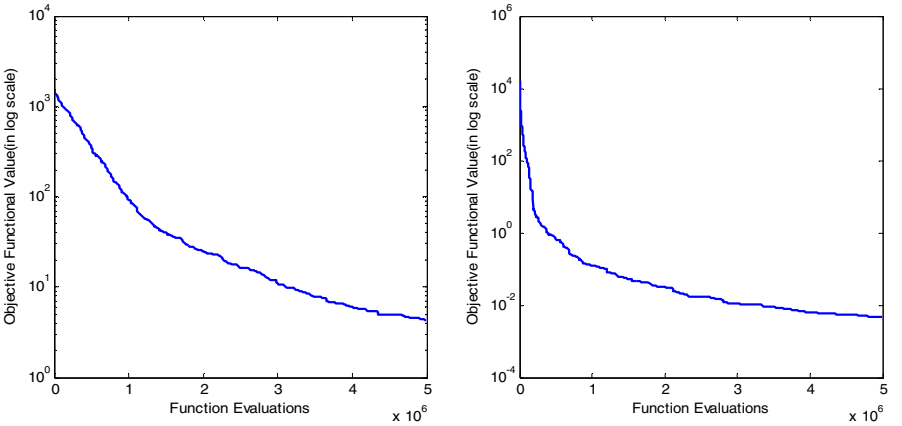


Fig. 2. Average objective functional values of LSCBO on functions of CEC'08 of 1000 dimensions. (a)*f_2(Schwefel Problem)*, (b)*f_4(Shifted Rastrigins)*

So according to the overall result, it is evident that LSCBO is able to give more accurate solution of the test functions than the recently developed EAs as well it performs very fast due to its simplicity.

4 Conclusion

A simple optimization algorithm LSCBO has been presented in this paper to solve large scale optimization problems. Due to the smaller number of searching agents in a population, it consumes much less time. Comparative studies of the performance of our algorithm LSCBO have been achieved by applying it on high dimensional uni-modal and multi-modal benchmark functions provided at CEC'08. From the experimental results we can conclude that the proposed algorithm can give better results with a higher convergence rate. Due to its simple dynamic opposition-based search, our proposed method becomes a highly competitive optimization algorithm for large scale optimization problems.

References

1. Rahnamayan, S., Tizhoosh, H.R., Salama, M.M.A.: Learning Robust Object Segmentation from User-Prepared Samples. WSEAS Transactions on Computers 4(9), 1163–1170 (2005)
2. Rahnamayan, S., Tizhoosh, H.R., Salama, M.M.A.: Towards Incomplete Object Recognition. WSEAS Transactions on Systems 4(10), 1725–1732 (2005)
3. Sharif, B., Wang, G.G., El-Mekkawy, T.: Mode Pursing Sampling Method for Discrete Variable Optimization on Expensive Blackbox Functions. ASME Transactions, Journal of Mechanical Design 130, 021402-1–021402-11 (2008)
4. Wang, L., Shan, S., Wang, G.G.: Mode-Pursuing Sampling Method for Global optimization on Expensive Black-box Functions. Journal of Engineering Optimization 36(4), 419–438 (2004)
5. Zhao, S.Z., Suganthan, P.N., Das, S.: Self-adaptive Differential Evolution with Multi-trajectory Search for Large Scale Optimization. Soft Computing 15(11), 2175–2185 (2011)
6. Qin, A.K., Huang, V.L., Suganthan, P.N.: Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans. on Evolutionary Computations, 398–417 (April 2009)
7. Li, X., Yao, X.: Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms. In: Proc. IEEE CEC, pp. 1546–1553 (May 2009)
8. Ros, R., Hansen, N.: A Simple Modification in CMA-ES Achieving Linear Time and Space Complexity. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN X. LNCS, vol. 5199, pp. 296–305. Springer, Heidelberg (2008)

A Strategy Adaptive Genetic Algorithm for Solving the Travelling Salesman Problem

Swahum Mukherjee[1], Srinjoy Ganguly[1], and Swagatam Das[2]

[1] Dept. of Electronics and Telecommunication Engg.,
Jadavpur University, Kolkata - 700 032, India
[2] Electronics and Communication Sciences Unit, Indian Statistical Institute,
Kolkata - 700 108, India
swahum.mukherjee@gmail.com, srinjoy_ganguly92@hotmail.com,
swagatam.das@ieee.org

Abstract. This paper presents a Strategy adaptive Genetic Algorithm to address a wide range of sequencing discrete optimization problems. As for the performance analysis, we have applied our algorithm on the Travelling Salesman Problem(TSP).Here we present an innovative crossover scheme which selects a crossover strategy from a consortium of three such crossover strategies, the choice being decided partly by the ability of the strategy to produce fitter off springs and partly by chance. We have maintained an account of each such strategy in producing fit off springs by adopting a model similar to The Ant Colony Optimization. We also propose a new variant of the Order Crossover which retains some of the best edges during the inheritance process. Along with conventional mutation methods we have developed a greedy inversion mutation scheme which is incorporated only if the operation leads to a more economical traversal. This algorithm provides better results compared to other heuristics, which is evident from the experimental results and their comparisons with those obtained using other algorithms.

Keywords: Genetic Algorithm, crossover, mutation, Travelling Salesman Problem; Ant Colony Optimization, pheromone, Roulette-Wheel Method, Minimum Spanning Tree.

1 Introduction

Evolutionary computation is a subfield of artificial intelligence (specifically computational intelligence) that involves combinatorial optimization problems. The evolutionary algorithms can be classified into i) Genetic Algorithms ii) Evolutionary Programming iii) Evolution Strategies and iv)Genetic Programming. In this paper we shall frame a new Strategy adaptive Genetic Algorithm (SaGA) that can efficiently solve the Travelling Salesman Problem (TSP) which is an NP-complete permutation problem.

The objective of TSP is to find the shortest route for a travelling salesman who, starting from his home city has to visit every city on a given list exactly once before

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 778–784, 2012.

returning to his starting city. Considering an n-city symmetric TSP, the search space will consist of (n-1)!/2 possible sequences of which we are to find the optimal sequence. For small values of n, the solution can be obtained easily but as the input size, n, increases, the run-time complexity increments exponentially.

In this paper in section 2 we provide an overview of genetic algorithms. A formal definition of the TSP is given in section 3. An account of the different components of genetic algorithms along with our modifications are given in section 4. In section 5 we present the results obtained for different TSP instances and compare them with other existing methods. Section 6 concludes the investigation.

2 Overview

Genetic algorithms are search algorithms inspired by evolutionary biology. Genetic algorithms are typically simulated as consisting of a population of abstract representations (chromosomes) of candidate solutions (individuals) to an optimization problem which evolves toward better solutions. The key steps of a standard genetic algorithm are:-

a) Initialization- Many individual candidate solutions are generated.
b) Selection- During each transition from a generation to the next, a portion of the population is selected to breed a new generation.
c) Reproduction- New individuals are created from pairs of individuals in the current population using genetic operators such as mutation and crossover.
d) Termination- The generational process is continued until a termination condition is satisfied.

3 The Travelling Salesman Problem

The Travelling Salesman Problem (TSP) is a well-known problem in the area of combinatorial optimization. The travelling salesman starts at one node, visits all other nodes successively only one time each, and finally returns to the starting node. i.e., given n cities, named $\{c_1, c_2,...,c_n\}$, and permutations, π_1, ..., π_n, the objective is to choose a permutation π_i of cities such that the sum of all Euclidean distances between each node and its successor is minimized. The Euclidean distance d, between any two cities with coordinate (x_1, y_1) and (x_2,y_2) is calculated by

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (1)$$

4 Key Components of the Algorithm

4.1 Fitness Function

The fitness of an n-city traversal sequence S is represented by g(x)= 1/f(x), where f(x)=c(S(1),S(2))+c(S(2),S(3))+…...+c(S(n-1),S(n))+c(S(n),S(1)), (2) Here c is the

cost matrix (or the adjacency matrix) of the problem. Our task is to maximize g(x) and hence minimize the distance function f(x).

4.2 Generation of the Initial Population

The initial population for the n-city TSP problem can be generated in two ways:-

1) Randomly generated.
2) Generated as per a certain rule so as to guarantee rapid convergence to the optimal solution.

In our algorithm we generate the initial population randomly to maintain a necessary diversity.

4.3 Selection for Crossover

Selection for crossover is done using the Roulette Wheel Method i.e. the members of the current population are crossed over with other members, whose fitness is adjacent to theirs.

4.4 Crossover

In this paper we propose a new, innovative crossover scheme which has the ability to select from a pool of three crossover operators maintaining an account of the success of the operators in producing fitter off springs. The majority of the current population converges to that crossover strategy which produces fitter off springs, as well as is probabilistically favoured. The operators competing are-

1. *Sequential Constructive Crossover (SCX)*
2. *Cyclic Crossover(CX)*
3. *Modified Branch Inherency Based Ordered Crossover (modified OX)*

Our algorithm aims to promote that crossover operator which produces fitter off springs for the members of the current population. To mathematically model this we have considered a model similar to the *Ant Colony Optimization Model* as proposed by Dorigo et.Al[1] where we have considered the three types of crossover as three distinct paths, one of which may be adopted by the pair undergoing crossover. This decision of choosing a path is made using two sets of continuously varying quantities- i) tau_{scx}, tau_{ox}, tau_{cx} ii) scx_{rand}, ox_{rand}, cx_{rand} $\epsilon(0,1)$. Every time a pair of parent chromosomes undergoes a certain crossover (say SCX), the *pheromone* on this path is updated as per the formula:

$$tau_{scx} = tau_{scx} + Q \times f_{scx}(x), \tag{3}$$

$f_{scx}(x)$ represents the fitness of the offspring obtained using the SCX operator. Likewise the off springs produced using the other two crossover schemes (OX and CX) also contribute to updating the pheromone on the respective paths using a similar formula. Now for every pair of members of the current population undergoing

crossover, the probability of a certain crossover scheme(say the modified OX) is given by $P_{ox}=(ox_{rand})^{A}*(tau_{ox})^{B}$, and likewise for SCX and CX.

The crossover strategy with the greatest probability is chosen to operate to produce the off springs. The values of the constants Q, A and B are user defined.

a) **Sequential Constructive Cross-Over**

We have adopted the Sequential Constructive Cross-Over(SCX), as proposed by Zakir H. Ahmed[2] , in order to ensure an increased rate of convergence. We have used the SCX because it exploits one major property of path representation, i.e. the distance between two successive cities. Hence, it selects the better edges from the two parents and forms the offspring(s).

b) **Cyclic Crossover**

The Cyclic Crossover Operator, as proposed by Oliver et al.[3], is unique because the off springs obtained are a result of recombination under the constraint that each city name comes from one parent or the other. The distance of the fitter of the two probable path traversals is employed in updating tau_{cx} as stated in the previous section.

c) **Modified branch inherency based order crossover(modified OX)**

In our approach we use a modified variant of the Order Crossover technique which ensures a high degree of convergence for large number of cities. The order crossover operator was proposed by Davis et al.[4]. In our algorithm, we follow OX5. The modifications are illustrated as follows. For example consider the following two parent tours which represent a permutation of 12 cities

p1= 1 4 7 3 11 6 9 10 5 2 8 12
p2= 3 2 10 1 4 9 7 12 6 8 11 5

Let the initial cut-points be randomly generated between positions 5 and 8. So the sub tours enclosed are copied into the corresponding off springs as shown below

o1= * * * * 11 6 9 10 * * * *
o2= * * * * 4 9 7 12 * * * *

Let the shortest tour subsequence of three consecutive city traversals in the remainder parent p1 be (4-> 7-> 3). The first pair of cut-points is generated enclosing this subsequence.

p1 = 1| 4 7 3| 11 6 9 10 5 2 8 12

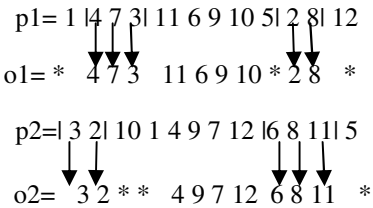
In the remaining subsequence let the shortest tour subsequence of two consecutive city traversals be (2->8).The second pair of cut-points is generated enclosing this subsequence.

p1= 1 4 7 3 11 6 9 10 5| 2 8| 12

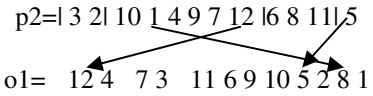
Let the shortest tour sub-sequences determined for parent2 be (6->8->11) and (3->2).

p2=| 3 2| 10 1 4 9 7 12 |6 8 11| 5

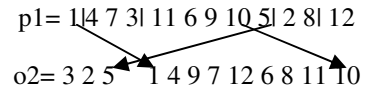
The shortest possible tour sub-sequences thus determined are copied into the corresponding off springs o1 and o2.



Starting from the end of the shortest tour sequence of three cities in p1,o1 is constructed preserving the order of cities in p2, starting from the end of the shortest subsequence of three cities in p2,omitting those which are already present in o1.When the end of the string p2 is reached we continue from its first position.



Similarly the second offspring o2 can be constructed as



4.5 Mutation

a) **Displacement mutation**
Displacement Mutation selects a subsequence at random and inserts it at a random position outside the subsequence.

b) **Exchange Mutation**
Exchange mutation selects two positions at random and swaps the cities on these positions.

c) **Greedy Inversion Mutation**
We have developed a novel inversion mutation scheme influenced by certain minimum spanning tree algorithms. One such is that proposed by Jarnik, rediscovered by Dijkstra and Prim[5]. Let us consider a graph with N vertices and M edges. Let S be a spanning forest (a candidate solution). Let us assume a sub-tour of the solution $s \in S$ where s contains n vertices. We consider this subsequence as a distinct ordered set in itself as inverting it would result in the same cost of traversal(valid for symmetric TSPs only). We now determine the cumulative cost c1(distance in this case) associated with the traversals of the edges across the cuts of the set s with the remainder of the spanning forest. We also determine the cumulative cost c2 associated with the traversals of the edges across the cuts of the inverted set s with the remainder of the spanning forest. If the algorithm detects c2 to be less than c1, it inverts the ordering in the set s, otherwise no inversion occurs.

5 Experimental Setup and Results

In our tests for TSP, the proposed SaGA algorithm is coded and complied in Matlab R2009b and the experiments are executed in a Core i3 3.0 GHz PC with 3 GB RAM.

Our results are compared with those obtained using S-CLPSO[6](Set-Based Particle Swarm Optimization), ACS[7](Ant Colony System), SWAP_GATSP[8], OX_SIM[9](standard GA with order crossover and simple inversion mutation), MOC_SIM(modified order crossover and SIM).

Table 1 summarizes the results obtained over 50 runs by running the SaGA, S-CLPSO, ACS, SWAP_GATSP, OX_SIM and MOC_SIM.

It is evident from the table that for majority of the different TSP instances, *the error percentages are lowest for SaGA*. Even the average results obtained using SaGA are more optimized than the ones obtained using S-CLPSO and ACS.

Table 1. Comparison of The Results Obtained over 50 Runs Using SaGA, S- CLPSO, ACS, SWAP_GATSP, OX_SIM and MOC_SIM for Different TSP Instances

Algorithm / Instances	Statistics	SaGA	ACS	S-CLPSO	SWAP_GATSP	OX_SIM	MOC_SIM
eil51	Best	**426**	429	426	439	493	444
51	Average	429	434.55	427.3	442	540	453
426	Error	0.7042	2.007	0.305	3.7559	26.7606	6.3380
KroA100	Best	**21282**	21355	21282	21397	21746	21514
100	Average	**21285**	21790.65	21352.5	21740	22120	21825
21282	Error	**0.0141**	2.39	0.3312	2.1521	3.9376	2.5515
eil76	Best	**538**	548	538	548	597	562
76	Average	544	557	541.7	555	620	580
538	Error	1.115	3.5315	0.6877	3.1599	15.2416	7.8067
St70	Best	**675**	677	675	685	823	698
70	Average	**677**	688.95	677.7	701	920	748
675	Error	**0.2962**	2.0667	0.4	3.8519	36.2963	10.8148
d198	Best	**15870**	———	———	15980	16542	16122
198	Average	**15914**			16106	17987	16348
15780	Error	**0.8491**			2.0659	13.9861	3.5995
pr299	Best	**48386**	48828	48478	———	———	———
299	Average	**48401**	49721.6	49222.5			
48191	Error	**0.4358**	3.1761	2.1404			
lin318	Best	**42492**	43050	42719	———	———	———
318	Average	**42604**	43624.4	43518.4			
42029	Error	**1.3681**	3.7959	3.5437			

Comparing SaGA with other COP solving algorithms such as S-CLPSO and ACS, the average of the error percentages for SaGA is 0.6556 which is less than 1.2346 for the S-CLPSO and 2.8278 for ACS. On comparing with other GAs, SaGA, with an average error percentage of 0.5957 also outperforms SWAP_GATSP(2.997), OX_SIM(19.24) and MOC_SIM(6.22).

6 Conclusion

In this paper we have used GA in a different approach and investigated it's suitability for the TSP problem. The algorithm has the liberty of choosing a crossover operator from a pool of three distinct such operators, the preference being determined by the ability of the chosen operator to produce fitter off springs. Such a competition among the operators is possible by using a model similar to the Ant Colony Optimization. We have also developed the Order Crossover operator so that the off springs inherit some of the best edges from their parents. At the same time to maintain better convergence speed we have introduced the greedy inversion mutation scheme which inverts a sequence in a candidate solution only if it results in a more economical traversal. This ensures a much faster convergence to the optimal solution. Thus using this algorithm helps in optimizing the speed and accuracy of the search process.

References

[1] Dorigo, M., Stutzle, T.: A Bradford Book. The MIT Press, ISBN 0-262-04219-3
[2] Ahmed, Z.H.: Genetic Algorithm for the Travelling Salesman Problem using sequential constructive crossover. IJBB 3(6) (2010)
[3] Oliver, I., et al.: A study of permutation crossover operators on the travelling salesman problem. In: Proc. of the Second Int. Conf. on Genetic Algorithms, pp. 224–230 (1987)
[4] Davis, L.: Applying Adaptive Algorithms to Epistatic Domains. In: Proceedings of the International Joint Conference on Artificial Intelligence, pp. 162–164 (1985)
[5] Prim, R.C.: Shortest connection networks and some generalizations. Bell System Technical Journal 36, 1389–1401 (1957)
[6] A Novel Set-Based Particle Swarm Optimization Method for Discrete Optimization Problems. IEEE Transactions on Evolutionary Computation 14(2) (April 2010)
[7] Solving Travelling Salesman Problem by Using Improved Ant Colony Optimization Algorithm. International Journal of Information and Education Technology 1(5) (December 2011)
[8] Ray, S.S., Bandyopadhyay, S., Pal, S.K.: Newoperators of genetic algorithms for traveling salesman problem. In: ICPR 2004, Cambridge, UK, vol. 2, pp. 497–500 (2004)
[9] Larranaga, P., Kuijpers, C., Murga, R., Inza, I., Dizdarevic, S.: Genetic algorithms for the travelling salesman problem: a review of representations and operators. Artificial Intell. Rev. 13, 129–170 (1999)

Efficient Design of Cosine-Modulated Filter Banks Using Evolutionary Multi-objective Optimization

Md. Nasir[1], Soumyadip Sengupta[1], and Swagatam Das[2]

[1] Dept. of Electronics and Telecomm. Engg. Jadavpur University, Kolkata - 700032, India
[2] Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata, India
{nasir795,roni14may}@gmail.com, swagatam.das@ieee.org

Abstract. We propose a novel and efficient way to design maximally decimated FIR cosine modulated filter banks, in which each analysis and synthesis filter has linear phase. We consider a class of near-perfect reconstruction CMFBs with the linear phase prototype filter, which structurally eliminates the amplitude overall distortion. The prototype filter design problem is then formulated into a multi-objective optimization problem (MOP), which aims at maximizing stop-band attenuation and minimizing reconstruction error simultaneously. We have modeled the design problem as a constrained multi-objective optimization problem which is efficiently solved by using a recently proposed algorithm MOEA/DFD. Experiment shows that the performance of MOEA/DFD exceeds that of MOEA/D and NSGA-II.

Keywords: Multi-rate filter, Cosine modulated filter bank, Prototype filter, Evolutionary multi-objective optimization.

1 Introduction

A substantial progress in the field of filter banks design and their applications have been made since the last two decade. Multirate filter banks are used in a variety of applications from data compression (speech, audio, image, and video) to communications (multicarrier modulation), and feature detection Among them, the cosine modulated filter banks(CMFB) have emerged as an attractive choice of filter banks with respect to implementation cost and design saving. In CMFBs, analysis and synthesis filter banks are derived by cosine modulating the low pass prototype filter. Hence the entire design of the filter banks is reduced to design the prototype filter. Therefore, during the design phase, it is required to optimize the coefficients of prototype filter only. This significantly reduces the complexities and computational overheads.

In the past, lot of work has been reported in developing filter banks [1–5]. Initially, the research effort was focused on design of the two-channel QMF banks [6–8] and later on it was extended to multi-channel QMF banks that can be designed using three different approaches [1,9]:

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 785–792, 2012.

- First approach is based on building the filter bank using a tree structure with two-channel filter banks as basic building blocks.
- In second approach, all the analysis and synthesis filters are considered to be independent, and are simultaneously optimized to satisfy the perfect reconstruction or near perfect reconstruction property.
- Last approach is based on cosine modulation or modified discrete Fourier transform (MDFT). In this technique, only the prototype filter needs to be designed, and all the analysis and synthesis filters are generated from this filter with the aid of cosine modulation or MDFT.

As compared to other two approaches mentioned above, cosine modulated filter banks are superior [1,9,10]:

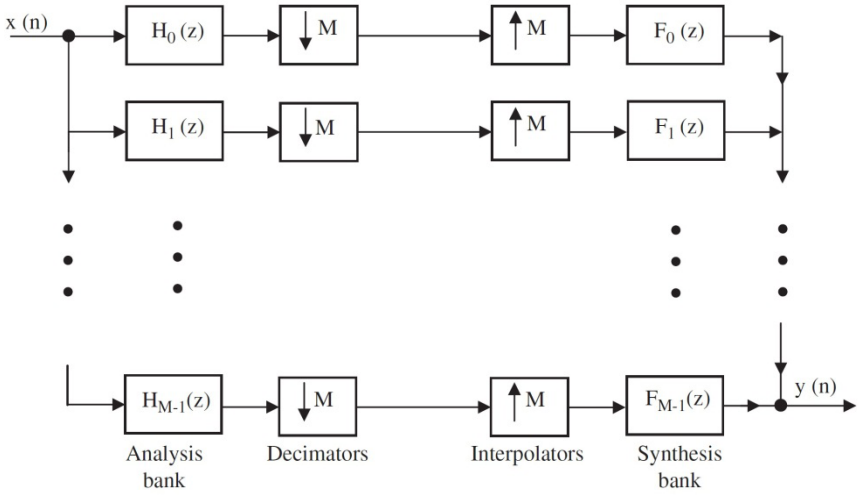


Fig. 1. M-channel maximally decimated filter bank

Cosine modulated filter banks are categorized in two ways: filter banks with approximate reconstruction and perfect reconstruction [1]. In CMFBs, first the analysis filters channelize the input signal to be processed. The extracted sub-band signals are then decimated and processed by a processing unit according to the application in hand. This is either stored or transmitted. In the receiver side, a synthesis filter bank reverses this process and reconstructs the original signal. The generalized parallel structure of cosine modulation is shown in Fig. 1.

In this paper, we show a class of cosine-modulated QMF banks with a prototype filter which structurally cancels the overall amplitude distortion. Then, the filter bank design problem essentially reduces to designing the prototype filter which has a frequency response satisfying given specifications. Different from the conventional (local) nonlinear optimization based CMFB design, our design approach is recast a multi-objective optimization problem (MOP) which considers both design objectives, namely, stop-band attenuation and reconstruction error.

We use an evolutionary multi-objective framework to achieve reconstruction error minimization and stop-band attenuation maximization with some constraints. A decision maker is implemented every time to choose the solution with maximum stop-band attenuation and with a fixed allowable reconstruction error, chosen according to the application requirement. We integrate the concept of a fuzzy dominance with original MOEA/D [12] and call the resulting algorithm MOEA/DFD (MOEA with Decomposition and Fuzzy Dominance), proposed by the authors in [13]. The performance of the algorithm in the present application context is compared with two state-of-the-art MOEAs, viz. NSGA - II (Non-dominated Sorting Genetic Algorithm) [14] and the original MOEA/D.

2 Problem Definition

In cosine-modulated QMF banks, all the analysis and synthesis filters can be generated by just modulating a linear phase low-pass prototype filter $H(z)$ with cutoff frequency $\frac{\pi}{2M}$ as follows:

$$h_k(n) = 2h(n)\cos\left(\frac{\pi}{M}(k+0.5)\left(n-\frac{N}{2}\right)+\theta_k\right)$$

$$f_k(n) = 2h(n)\cos\left(\frac{\pi}{M}(k+0.5)\left(n-\frac{N}{2}\right)-\theta_k\right)$$

for $0 \le n \le N, 0 \le k \le M-1.$

where $h_k(n)$ and $f_k(n)$ represent the impulse responses of the kth channel analysis and synthesis filters, respectively.

In the M-channel maximally decimated filter bank shown in Fig. 1, the reconstructed signal $Y(z)$ can be expressed in z-domain as:

$$Y(z) = X(z)T_0(z) + \sum_{l=1}^{M-1} X(ze^{-j2\pi l/M})T_l(z) \tag{1}$$

where $T_l(z) = \frac{1}{M}\sum_{k=0}^{M-1} H_k(ze^{-j2\pi l/M})F_k(z)$.

Here $T_0(z)$ is the overall distortion function and $T_l(z), l \neq 0$ is the aliasing transfer function. To cancel aliasing and achieve perfect reconstruction, it is required that

$$\begin{aligned} T_l(z) &= 0 \qquad \text{for } l = 1,2,\ldots,M-1 \\ T_0(z) &= cz^{-n_d} \end{aligned} \tag{2}$$

As the analysis and synthesis filters have narrow transition bands and high stop-band attenuation, the overlap between nonadjacent filters is negligible. Moreover, it was

shown in [11] that significant aliasing terms from the overlap of the adjacent filters are cancelled by choosing $\theta_k = (-1)^k \pi / 4$. Under these circumstances, the overall distortion function is given by

$$T_0(e^{j\omega}) = \frac{e^{-j\omega N}}{M} \sum_{k=0}^{2M-1} \left| H(e^{j(\omega - k\pi/M - \pi/2M)}) \right|^2 \tag{3}$$

It can be verified that $T_0(e^{j\omega})$ is periodic with period π/M, and so is approximated by

$$T_0(e^{j\omega}) \approx \frac{e^{-j\omega N}}{M} [\left| H(e^{j(\omega - k\pi/M - \pi/2M)}) \right|^2 + \left| H(e^{j(\omega - (k+1)\pi/M - \pi/2M)}) \right|^2], \tag{4}$$

for $\omega \in [(k+\frac{1}{2})\frac{\pi}{M}, (k+\frac{3}{2})\frac{\pi}{M}]$.

To eliminate amplitude distortion, $\left| T_0(e^{j\omega}) \right|$ must be constant for all frequencies, i.e.,

$$\left| H(e^{j\omega}) \right|^2 + \left| H(e^{j(\omega - \pi/M)}) \right|^2 = 1, \tag{5}$$

for $\omega \in [0, \frac{\pi}{M}]$.

Now, let $H(e^{j\omega})$ be a lowpass filter with the passband and stopband edges

$$\omega_p = (\frac{\pi}{2M} - \in), \qquad \omega_s = (\frac{\pi}{2M} + \in). \tag{6}$$

where $0 < \in < \pi/2M$ decides the transition bandwidth. Assuming that the filter has small ripples in the passband and high attenuation in the stopband, we have

$$\left| H(e^{j\omega}) \right|^2 + \left| H(e^{j(\omega - \pi/M)}) \right|^2 \approx 1, \tag{7}$$

for $\omega \in [0, \omega_p] \bigcup [\omega_s, \frac{\pi}{M}]$.

In the transition band, if

$$\left| H(e^{j\omega}) \right| = \cos(\frac{\pi}{4\in}(\omega - \omega_p)), \qquad \omega \in [\omega_p, \omega_s], \tag{8}$$

$$\text{then } \left| H(e^{j\omega}) \right|^2 + \left| H(e^{j(\omega - \pi/M)}) \right|^2 = \cos^2(\frac{\pi}{4\in}(\omega - \omega_p)) + \sin^2(\frac{\pi}{4\in}(\omega - \omega_p)) = 1, \tag{9}$$

In summary, the overall amplitude distortion will be cancelled if the prototype filter has the following magnitude response

$$\left|H(e^{j\omega})\right| = \begin{cases} 1 & \omega \in [0, \omega_p] \\ \cos(\frac{\pi}{4\epsilon}(\omega - \omega_p)) & \omega \in [\omega_p, \omega_s] \\ 0 & \omega \in [\omega_s, \pi] \end{cases} \tag{10}$$

For simplicity, we assume that the linear phase prototype filter have even order, i.e., $N = 2L$. Then

$$H(e^{j\omega}) = e^{-j\omega N/2} H_R(\omega) \tag{11}$$

and the amplitude response is

$$H_R(\omega) = \sum_{k=0}^{L} g_k \cos(k\omega) \tag{12}$$

Here the variable vector $\mathbf{g} = [g_0, g_1, g_2, ..., g_L]^{\mathrm{T}} = [h_L, 2h_{L-1}, 2h_{L-2}, ..., 2h_0]^{\mathrm{T}}$ represents the prototype filter coefficients.

Now we have to maximize the stopband attenuation which is given as

$$A_s = -20\log\delta_s \text{, where } \delta_s = \max_{\omega \in [\omega_s, \pi]} \left|H(e^{j\omega})\right|. \tag{13}$$

Also we have to minimize the maximum peak to peak reconstruction error

$$E_{pp} = \max_{\omega}(MT_0(e^{j\omega})) - \min_{\omega}(MT_0(e^{j\omega})); \tag{14}$$

Moreover, there are some constraints imposed on this optimization problem as follows:

$$\begin{aligned} &\left|H_R(\omega) - 1\right| \le \delta_p, && \omega \in [0, \omega_p], \\ &\left|H_R(\omega) - \cos(\frac{\pi}{4\epsilon}(\omega - \omega_p))\right| \le \delta_t, && \omega \in [\omega_p, \omega_s], \\ &\left|H_R(\omega)\right| \le \delta_s, && \omega \in [\omega_s, \pi]. \end{aligned} \tag{15}$$

3 Multi-objective Optimization, MOEA/DFD –Brief Background

A multi-objective optimization problem (MOP) can be formally stated as follows:

$$\text{Minimize } F(\vec{x}) = (f_1(\vec{x}), ..., f_m(\vec{x})) \text{ subject to } \vec{x} \in \Omega, \tag{16}$$

where Ω is the decision space, $F: \Omega \rightarrow R^m$ consists of m real-valued objective functions and R^m is called the objective space. If $\vec{x} \in R^n$, all the objectives are continuous and Ω is described by $\Omega = \{\vec{x} \in R^n \mid h_j(\vec{x}) \le 0, j = 1, ..., k\}$, where h_j are continuous functions, we call (16) a continuous MOP[16],[17].

3.1 MOEA/DFD: A Brief Discussion

We have already proposed MOEA/DFD in [13] and have clearly discussed the detailed algorithm and reasons behind the better performance of it. The main difference between MOEA/D and MOEA/DFD lies in the updation strategy which is described in form of a pseudo code is short and the symbols have their usual meanings.

Update of *z*: For each $j=1,\ldots,m$, if $z_j > f_j(\vec{y})$, then set $z_j = f_j(\vec{y})$.

Calculation of fuzzy dominance level:

For each $j \in P$,

For each $i \in \{1,\ldots,m\}$

if $f_i(\vec{y}) \le f_i(\vec{x}^j)$, $\mu_i^j = 1$;

else $\mu_i^j = e^{-A(f_i(\vec{y}) - f_i(\vec{x}^j))}$; end.

$\hat{\mu}^j = \cap \mu_i^j = \prod_{i=1}^{m} \mu_i^j$;end.

$\tau = 0.1 * gen / (\max\_gen)$;end.

Update of solutions: For each $j \in P$; If $\hat{\mu}^j > \tau$, then $x^j = y$, $FV^j = F(\vec{y})$

else, if $g^{te}(\vec{y}^j \mid \lambda^j, \vec{z}^*) \le g^{te}(\vec{x}^j \mid \lambda^j, \vec{z}^*)$, then $\vec{x}^j = \vec{y}$, $FV^j = F(\vec{y})$;

4 Experimental Results

4.1 Experimental Setup

Here a design example is considered to establish the efficiency of the proposed method experimentally. The result is compared to that of approaches based on two other evolutionary multi-objective optimization algorithms, namely, NSGA-II and MOEA/D. A 17-channel cosine modulated pseudo-QMF filter bank is to be designed with the prototype filter order of N=102. The other specifications are given as follows:

- $\omega_p = \frac{3.1176}{10000}\pi, \omega_s = 0.0585\pi,$
- peak passband ripple $\delta_p = 10^{-3}$
- transition band peak error $\delta_t = 10^{-3}$
- peak stopband ripple $\delta_s = 10^{-2}$

For MO algorithms, population size (N) is kept at 300 with 50,000 function evaluations. The parameters of MOEA/DFD, MOEA/D and NSGA-II are used as recommended. In all those cases we have used uniform crossover and one bit flip mutation scheme to maintain uniformity. Other parameters of those algorithms are kept as recommended in respective literatures.

4.2 Result

In a nutshell the objective is to maximize stopband attenuation while minimizing peak to peak reconstruction error. As the objectives are conflicting it is not possible to achieve their optimal values simultaneously. For this reason we have extracted the solution which attains some specific value of stopband attenuation and compared maximum peak to peak reconstruction error in our approach with that of two other algorithms in the set-up described in the previous sub-section. Results have taken from the values generated in 25 independent runs of each algorithm and the average values are reported. Figure 2 represents the Pareto optimal front of the multi-objective optimization problem of the design process.

Table 1. Experimental results for Maximum Peak to Peak Reconstruction Error for Specific Values of Stopband Attenuation

	Maximum Peak to Peak Reconstruction Error		
Stopband Attenuation	NSGAII	MOEA/D	MOEA/DFD
-20 dB	1.867e-3	9.446e-4	**8.793e-4**
-30 dB	3.315e-3	2.242e-3	**1.953e-3**
-40 dB	4.040e-3	3.565e-3	**2.993e-3**
-50 dB	6.063e-3	4.784e-3	**4.276e-3**

From the results listed in Tables 1 or as shown in the Figure 2, it is quite evident that MOEA/DFD outperforms NSGA-II and MOEA/D. In other words, the maximum peak to peak reconstruction error is minimum for all given values of stopband attenuation. NSGA - II performs worst in all those aspects.

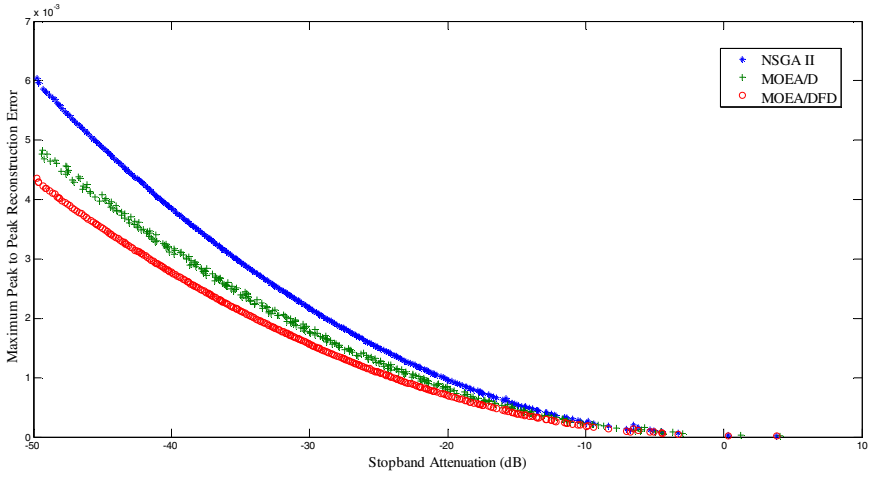


Fig. 2. Pareto optimal fronts generated by different algorithms

5 Conclusion

In this work, a multi-objective optimization based technique for design of prototype filter is presented. The simulation results show that the optimization algorithm effectively reduced the peak reconstruction error with comparable stopband attenuation. Not only that this multi-objective approach for cosine modulated filter bank is novel, but also the proposed framework based on MOEAD/DFD outperforms other evolutionary multi-objective algorithms. Much more simulation is under way to confirm the theoretical advantage of our method over other existing ones. This simple yet efficient approach can be used to many other filter design problems as well.

References

[1] Vaidyanathan, P.: Multirate and Filter Banks. Prentice-Hall, Englewood Cliffs (1993)
[2] Creusere, C.D., Mitra, S.K.: A simple method for designing high quality prototype filters for M-band pseudo-QMF banks. IEEE Trans. Signal Process. 43(4), 1005–1007 (1995)
[3] Nguyen, T.Q.: Near-perfect-reconstruction pseudo-QMF banks. IEEE Trans. Signal Processing 42(1), 65–76 (1994)
[4] Heller, P.N., Karp, T., Nguyen, T.Q.: A general formulation of modulated filter banks. IEEE Trans. Signal Process. 47(4), 986–1002 (1999)
[5] Datar, A., Jain, A., Sharma, P.C.: Design of Kaiser window based optimized prototype filter for cosine modulated filter banks. Signal Processing 90(5), 1742–1749 (2010)
[6] Johnston, J.D.: A filter family designed for use in quadrature mirror filter banks. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 291–294 (1980)
[7] Jou, Y.D.: Design of two channel linear phase QMF bank based on neural networks. Signal Processing 87, 1031–1044 (2007)
[8] Kok, C.W., Siu, W.C., Law, Y.M.: Peak constrained least QMF banks. Signal Processing 88, 2363–2371 (2008)
[9] Dolecek, G.J.: Multirate Systems: Design and Applications. Idea Group of Publishing (2002)
[10] Xu, H., Lu, W.-S., Antoniou, A.: Efficient iterative design method for cosine modulated QMF banks. IEEE Transactions on Signal Processing 44(7), 1657–1667 (1996)
[11] Kha, H.H., Tuan, H.D., Nguyen, T.Q.: An Efficient SDP Based Design for Prototype Filters of M-Channel Cosine-Modulated Filter Banks. In: International Conference on Acoustics, Speech, and Signal Processing (ICASSP), vol. 3, pp. III-893–III-896 (2007)
[12] Zhang, Q., Li, H.: MOEA/D: A multi-objective evolutionary algorithm based on decomposition. IEEE Transactions on Evolutionary Computation 11(6), 712–731 (2007)
[13] Nasir, M., Mondal, A.K., Sengupta, S., Das, S., Abraham, A.: An improved Multiobjective Evolutionary Algorithm based on decomposition with fuzzy dominance. In: Proceedings of the Conference on Congress on Evolutionary Computation, Trondheim, Norway, May 18-21, pp. 203–208. IEEE Press, Piscataway (2009)
[14] Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE Trans. on Evolutionary Computation 6(2), 182–197 (2002)
[15] Koduru, P., Das, S., Welch, S., Roe, J.L.: Fuzzy Dominance Based Multi-objective GA-Simplex Hybrid Algorithms Applied to Gene Network Models. In: Deb, K., Tari, Z. (eds.) GECCO 2004. LNCS, vol. 3102, pp. 356–367. Springer, Heidelberg (2004)
[16] Zhou, A., Qu, B.-Y., Li, H., Zhao, S.-Z., Suganthan, P.N., Zhang, Q.: Multiobjective Evolutionary Algorithms: A Survey of the State-of-the-art. Swarm and Evolutionary Computation 1(1), 32–49 (2011)
[17] Zhao, S.Z., Suganthan, P.N., Zhang, Q.: Decomposition Based Multiobjective Evolutionary Algorithm with an Ensemble of Neighborhood Sizes. IEEE Trans. on Evolutionary Computation 16(3), 442–446 (2012)

Voltage Stability Constrained Optimal Power Flow Using Non-dominated Sorting Genetic Algorithm-II (NSGA II)

C. Nithya[1], J. Preetha Roselyn[1], D. Devaraj[2], and Subhransu Sekhar Dash[3]

[1] SRM University,Kattankulathur-603203
{nith.satish,preetha.roselyn}@gmail.com
[2] Kalasalingam University, Srivilliputhur-626190
deva230@yahoo.com
[3] SRM University,Kattankulathur-603203
munu_dash_2k@yahoo.com

Abstract. Voltage stability has become an important issue in planning and operation of many power systems. Contingencies such as unexpected line outages in a stressed system may often result in voltage instability, which may lead to voltage collapse. This paper presents evolutionary algorithm techniques like Genetic Algorithm (GA) and Non-dominated Sorting Genetic Algorithm II (NSGA II) approach for solving the Voltage Stability Constrained Optimal Power Flow (VSC-OPF). Base-case generator power output, voltage magnitude of generator buses are taken as the control variables. Maximum L-index of load buses is used to specify the voltage stability level of the system. An improved GA which permits the control variables to be represented in their natural form is proposed to solve the Optimal Power Flow (OPF) problem and NSGA II is proposed to solve the VSC-OPF optimization problem. For effective genetic operation, crossover and mutation operators which can directly operate on floating point numbers and integers are used. The proposed approach has been evaluated on the IEEE 30-bus test system. Simulation results show the effectiveness of the proposed NSGA II approach than Multi-Objective Genetic Algorithm (MOGA) for improving the voltage security of the system.

1 Introduction

OPTIMAL Power Flow (OPF) [1] is a nonlinear programming problem, and is used to determine optimal outputs of generators and bus voltage in power system, with an objective to minimize total production cost, while the system is operating within its security limit. Since OPF was introduced in 1968, several methods have been employed to solve this problem, *e.g*. Gradient base [9], Linear programming method [3] and Interior point method [12]. However all of these methods suffer from three main problems. Firstly, they may not be able to provide optimal solution and usually getting stuck at local optima [17]. Secondly, all these methods are based on assumption of continuity and differentiability of objective function which is not actually allowed in a practical system. Finally, all these methods cannot be applied with discrete variables, which are transformer taps. It seems that GA is an appropriate

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 793–801, 2012.

method to solve this problem, which eliminates the above drawbacks [24, 22, and 11]. GA, invented by Holland in the early 1970s, is a stochastic global search method that mimics the metaphor of natural biological evaluation.

Genetic Algorithms (GA) [8] operates on a population of candidate solutions encoded to finite bit string called chromosome. In order to obtain optimality, each chromosome exchanges the information using operators borrowed from natural genetic to produce the better solution.

In the recent past, several voltage instability incidents have been reported all over the world. A system enters a state of voltage instability when a disturbance causes a progressive and uncontrollable decline in voltage. This change in voltage is so rapid that voltage control devices may not take corrective actions rapidly enough to prevent cascading outages. Voltage instability is typically associated with the presence of reactive power demands of loads not being met because of limitations on the production and transmission of reactive powers. Contingencies such as unexpected line outages in a stressed system may often result in voltage instability, which may lead to voltage collapse.

This problem can only be exacerbated by the application of open-market principles to the operation of power systems, as stability margins will be reduced even further to respond to market pressures, which demands greater attention on reduced operating costs. This has necessitated the importance of incorporating the voltage stability limit in the power system operation and planning stage. Canizares [5] have proposed two different formulations to include the voltage stability in the OPF formulation. In the first approach, the problem was formulated as a single-objective optimization problem with cost minimization [7].In the second approach, called the voltage stability constraint OPF method, the problem was formulated as a multi-objective optimization problem with cost minimization as the objective function and the voltage stability limit as the second objective function . In the present paper, the voltage stability indicator is incorporated in the OPF formulation through the L-index value.

In this paper three different cases are considered. In the first case base case OPF as a single objective optimization problem is solved using GA [7] . in the second case VSC-OPF problem is formulated in MOGA with minimization of fuel cost and L-index value. In the third case VSC-OPF problem is considered as a multi-objective problem and is solved using the NSGA II approach in an IEEE 30 bus system. NSGA [19] is a popular non-domination based genetic algorithm for multi-objective optimization. It is a very effective algorithm but has been generally criticized for its computational complexity, lack of elitism and for choosing the optimal parameter value for sharing parameter $\sigma share$. A modified version, NSGA- II [14] was developed, which has a better sorting algorithm, incorporates elitism and no sharing parameter needs to be chosen *a priori*.

2 Voltage Stability Index

The voltage stability analysis involves determination of an index known as voltage collapse proximity indicator. This index is an approximate measure of the closeness of the system to voltage collapse. There are various methods of determining the

voltage collapse proximity indicator. One such method is the L-index method proposed in Kessel and Glavitsch (1986) [16]. It is based on load flow analysis. Its value ranges from 0 (no load condition) to 1 (voltage collapse). The bus with the highest L-index value will be the most vulnerable bus in the system. The L-index has the advantage of indicating voltage instability proximity of the current operating point without calculation of the information about the maximum loading point.

3 Problem Formulation

List of symbols

N	Number of total buses
N_G	Number of generator buses
N_L	Number of load buses
N_l	Number of transmission lines
P_i,Q_i	Real and reactive power injected at bus i.
$G_{ij,}$ B_{ij}	Mutual conductance and susceptance between bus *i* and bus *j*
$G_{ii,}$ B_{ii}	Self-conductance and susceptance of bus *i*
$\|V_i\|$	Voltage magnitude at bus i.
P_{gi},Q_{gi}	Real and reactive power generation at bus.

In general, the OPF problem is formulated as an optimisation problem in which a specific objective function is minimised while satisfying a number of equality and inequality constraints. The objectives of the OPF problem considered here are minimisation of fuel cost in the normal state and the minimisation of the voltage stability index L^{max} in the emergency state. Power flow equations are the equality constraints of the problem, while the inequality constraints include the limits on real and reactive power generation and bus voltage magnitude as follows.

$$Min\ F(x) = \sum_{i=1}^{N_G}\left(a_i + b_i P_{Gi} + c_i P_{Gi}^2\right) \tag{1}$$

Where N_G is the number of generation including the slack bus. P_{gi} is the generated active power at bus *i*. a_i, b_i and c_i are the unit costs curve for i^{th}generator.

Subject to:

- Load flow constraints

$$\left.\begin{aligned} P_i - V_i \sum_{j=1}^{N_B} V_j \left(G_{ij}\cos\theta_{ij} + B_{ij}\sin\theta_{ij}\right) = 0 \\ Q_i - V_i \sum_{j=1}^{N_B} V_j \left(G_{ij}\sin\theta_{ij} + B_{ij}\cos\theta_{ij}\right) = 0 \end{aligned}\right\} \tag{2}$$

- Generator real power generation limit

$$P_{gi}^{min} \leq P_{gi} \leq P_{gi}^{max}, \qquad i=1.....,N_G \tag{3}$$

- Generator reactive power generation limit

$$Q_{gi}^{min} \leq Q_{gi} \leq Q_{gi}^{max}, \quad i=1.....,N_G \tag{4}$$

- Voltage constraint

$$V_i^{min} \leq V_i \leq V_i^{max}, \qquad i=1\ldots\ldots N_G \tag{5}$$

- Transmission line flow

$$S_l \leq S_l^{max} \qquad\qquad l \in N_l \tag{6}$$

The equality constraints given by the above equations are satisfied by running the power flow program. The active power generation (P_{gi}) (except the generator at the slack bus) and generator terminal bus voltages (V_{gi}) are the optimization variables and they are self-restricted by the optimization algorithm.

4 Non-dominated Sorting Genetic Algorithm II (NSGA II)

NSGA introduced by Srinivas and Deb [20], implements the idea of a selection method based on classes of dominance of all solutions. This algorithm identifies non-dominated solutions in the population, at each generation, to form non-dominated fronts, based on the concept of non-dominance of Pareto. After this, the usual selection, crossover, and mutation operators are performed.

5 Simulation Results

The proposed NSGA II approach has been applied to solve the VSC-OPF problem in an IEEE 30-bus test system. The system has six generator buses, 24 load buses and 41 transmission lines .The generator cost coefficients and the transmission line parameters are taken from Alsac and Scott (1974) [1]. Three different cases were considered for simulation, one without considering the voltage stability i.e, base case OPF using GA, the next one to solve the VSC-OPF problem using MOGA and the third one to solve VSC-OPF problem using NSGA II .These simulations were implemented using the MATLAB program. The results of these simulations are presented below.

5.1 Case (i): OPF with Cost Minimization Objective Using GA

In this case, the GA-based algorithm was applied to identify the optimal control variables of the system under base-load condition, with cost minimisation objective

and without considering the voltage stability of the system. The optimal values of the control variables along with the minimum fuel cost obtained are given in Table 1. Corresponding to this control variable setting, it was found that there are no limit violations in any of the state variables. The minimum cost solution obtained by the proposed approach and the minimum cost reported in the literature are presented in Table 2. From this table it is found that the minimum cost obtained by the proposed method is less than the values reported in many papers. Figure 1 shows the graphical representation of selecting the fitness value.

Table 1. Base Case OPF

Control Variables	Variable Setting
P1(MW)	177.3771
P2(MW)	48.4817
P5(MW)	21.3846
P8(MW)	21.7854
P11(MW)	21.3842
P13(MW)	12.3842
V1(pu)	1.043
V2(pu)	1.06
V5(pu)	1.043
V8(pu)	1.01
V11(pu)	1.072
V13(pu)	1.071
Cost($/hr)	801.9697
Loss(MW)	9.444

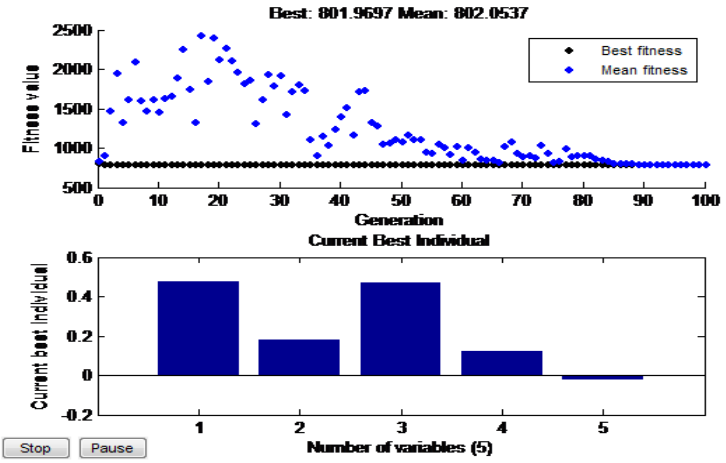


Fig. 1. Base case OPF using GA

Table 2. Comparison of fuel cost

Method	Minimum Cost($/hr)
Gradient approach [1]	802.43
Refined GA [21]	804.019
Hybrid evolutionary programming [25]	802.62
Evolutionary programming [23]	802.40
Proposed Method	801.9697

5.2 Case (iii): VSC-OPF Using NSGA II

The voltage stability index (L-index) is included as the second objective function of the VSC-OPF problem along with the base fuel cost. The NSGA II based algorithm was applied to solve this VSC-OPF problem. The optimal control variable setting obtained in this case is presented in Table 3 along with the L-index value. Figure 2 shows the pareto optimal front of generation cost and L-index .The solution is a set of non-dominated solutions. The comparison of the results obtained in multi-objective GA and NSGA II is shown in Table 4. From this table it is clear that the performance of NSGA II is better than MOGA in VSC-OPF problem.

Table 3. Results of VSC-OPF using NSGA II

Control Variables	**Variable Setting**
P1(MW)	170.5467
P2(MW)	50.6320
P5(MW)	22.3488
P8(MW)	20.3435
P11(MW)	14.0137
P13(MW)	16.0340
V1(pu)	1.0000
V2(pu)	0.9999
V5(pu)	1.0000
V8(pu)	1.0000
V11(pu)	1.0028
V13(pu)	1.0000
Cost($/hr)	808.1248
L_{max}	0.1104

Contingency analysis was conducted on the system by simulating the single line outage. In each case the maximum L-index value was evaluated. From the contingency analysis it was found that line outage 28-27 is the most severe one from the voltage security point of view during this contingency state. Table 5 gives the fuel cost, L_{max} and minimum voltage value of the contingency constrained VSC-OPF using NSGA II. This reduction in L_{max} is obtained at the expense of increased fuel cost. Figure 3 shows the pareto optimal front of contingency constrained VSC-OPF.

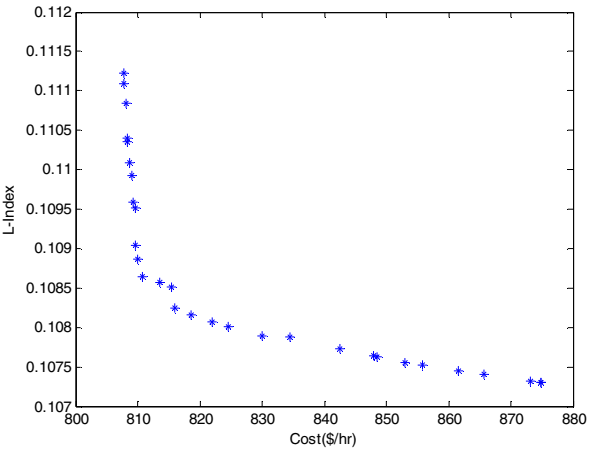


Fig. 2. Pareto optimal front for generation cost and L-index

Table 4. Comparison of results obtained in multi-objective GA and NSGA II

Method	MOGA	NSGA II
Cost($/hr)	802.5989	802.128
L_{max}	0.1904	0.1106

Table 5. Contingency constrained VSC-OPF using NSGA II Results

Cost ($/hr)	818.0289
L_{max}	0.2951
V_{min} (pu)	0.9779

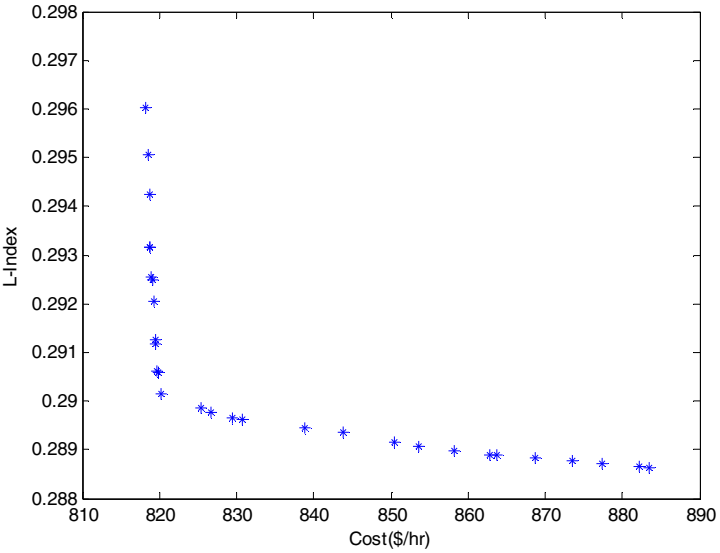


Fig. 3. Pareto optimal front of contingency constrained VSC-OPF

6 Conclusion

In this paper, the various aspects of single–objective optimal power flow and multi-objective voltage stability constrained optimal power flow are studied. An efficient and diversified approach using NSGA II algorithm is identified to solve the above multi-objective optimisation problems. Several case studies have been employed separately for single and multi-objective optimization problem. Firstly, the results are obtained for single objective OPF and contingency constrained VSC-OPF using genetic algorithm for the optimisation of Fuel cost which are then compared with the optimal power flow results of other papers. The multi-objective VSC-OPF problem is formulated using NSGA II algorithm. The proposed algorithm occupies less memory space and takes less CPU time than conventional GA approach. Simulation results of the IEEE 30-bus system have been presented to illustrate the effectiveness of the proposed approach to solve the VSC-OPF problem. Simulation were carried out using NSGA II and are found that voltage stability is improved in the proposed approach than other approaches, the algorithm has helped to improve the voltage security level of the system.

References

1. Alsac, O., Scott, B.: Optimal load flow with steady state security. IEEE Transactions on Power Systems PAS-93(3), 745–751 (1974)
2. Adel, A., Ela, A.E.L., Spea, S.R.: Optimal Correction Actions for Power Systems using Multi-Objective Genetic Algorithm. Electric Power Systems Research 79, 722–733 (2009)
3. Wood, A.J., Wollenberg, B.F.: Power Generation Operation and Control. John Wiley & Sons, Inc., NewYork (1996)

4. Baghaee, H.R., Jannati, M., Vahidi, B., Hosseinian, S.H., Rastegar, H.: Improment of Voltage Stability and Reduce Power System Losses by Optimal GA-Based Allocation of Multi–type FACTS Devices. In: IEEE Conference Publications, pp. 209–214 (2008)
5. Canizares, C., et al.: Comparison of voltage security constrained optimal power flow techniques. In: Proc. 2001, IEEE-PES Summer Meeting, Vancouver, BC (2001)
6. Devaraj, D., Yegnanarayana, B.: A combined genetic algorithm approach for optimal power flow. In: National Power Systems Conference, NPSC 2000, Bangalore, India, pp. 1866–1876 (2000)
7. Devaraj, D., Roselyn, J.P.: Improved genetic algorithm for voltage security constrained optimal power flow problem. Int. J. Energy Technology and Policy 5(4), 475–488 (2007)
8. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and MachineLearning. Addison Wesley Publishing Company, Inc., USA (1989)
9. Dommel, H.W., Tinney, W.F.: Optimal power flow solutions. IEEE Transactions on Power Apparatus and Systems PAS-87(10), 1866–1876 (1968)
10. Zitzler, E., Thiele, L.: Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. IEEE Transactions on Evolutionary Computation 3(4) (November 1999)
11. Sadat, H.: Power systems analysis. McGraw Hill Publication, New Delhi (1997)
12. Stott, B., Hobson, E.: Power system security control calculations using linear Programming. IEEE Transactions on Power Apparatus and Systems, PAS 97, 1713–1931 (1978)
13. Beyer, H.-G., Deb, K.: On Self-Adaptive Features in Real-Parameter Evolutionary Algorithm. IEEE Transactions on Evolutionary Computation 5(3), 250–270 (2001)
14. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A Fast Elitist Multi-objective Genetic Algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2), 182–197 (2002)
15. Deb, K., Agarwal, R.B.: Simulated Binary Crossover for Continuous Search Space. Complex Systems 9, 115–148 (1995)
16. Kessel, P., Glavitsch, H.: Estimating the voltage stability of power systems. IEEE Transactions on Power Delivery 1(3), 346–354 (1986)
17. Lee, K.Y., Park, Y.M., Ortiz, J.L.: A United Approach to Optimal Real and Reactive Power Dispatch. IEEE Transactions on Power Systems 3, 104 (1986)
18. Raghuwanshi, M.M., Kakde, O.G.: Survey on multiobjective evolutionary and real coded genetic algorithms. In: Proceedings of the 8th Asia Pacific Symposium on Intelligent and Evolutionary Systems, pp. 150–161 (2004)
19. Srinivas, N., Deb, K.: Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. Evolutionary Computation 2(3), 221–248 (1994)
20. Srinivas, N., Deb, K.: Multiobjective optimization using nondominated sorting in genetic algorithms. Technical report, Department of Mechanical Engineering, Indian Institute of Technology, Kanpur, India (1993)
21. Paranjothi, S.R., Anburaja, K.: Optimal power flow using refined genetic algorithm. Electric Power Components and Systems 30, 1055–1063 (2002)
22. Fletcher, R.: Practical Methods of Optimisation. John Willey & Sons (1986)
23. Somasundaram, P., Kuppusamy, K., Kumudini Devi, K.P.: Evolutionary programming based security constrained optimal power flow. Electric Power Systems Research 72, 137–145 (2004)
24. Bouktir, T., Belkacemi, M., Zehar, K.: Optimal power flow using modified gradient method. In: Proceeding ICEL 2000, U.S.T.Oran, Algeria, November 13-15, vol. 2, pp. 436–442 (2000)
25. Yuryevich, J., Wang, K.P.: Evolutionary programming based optimal power flow algorithms. IEEE Transactions on Power Systems 14(4), 1245–1250 (1999)
26. Zhou, A., Qu, B.-Y., Li, H., Zhao, S.-Z., Suganthan, P.N., Zhang, Q.: Multiobjective Evolutionary Algorithms: A Survey of the State-of-the-art. Swarm and Evolutionary Computation 1(1), 32–49 (2011)

Markov Random Field Model Based Graduated Penalty Function for Reinforcing Ill-Defined Edges in Color Image Segmentation

Panda Sucheta[1] and P.K. Nanda[2]

[1] Department of Computer Science and Engineering,
Padmanava College of Engineering Rourkela, Orissa, India
[2] Dept. of Electronics and Communication Engineering I.T.E.R,Siksha 'O'
Anusandhan University, Jagmohan Nagar,Khandagiri Bhubaneswar, Orissa, India
{pknanda.nitrkl,pandasucheta06}@gmail.com

Abstract. In the color image segmentation, the usual approach is to segment the image while attempting preserve strong edges. But often in real world, many weak edges or poorly defined edges need to be preserved in order to achieve meaningful segmentation. In this paper, we have proposed a color image segmentation scheme in statistical framework, where the weak edges have been reinforced together with the strong edges. In this regard we have proposed the notion of graduated edge penalty function based bilevel line field to take care of strong as well as ill defined edges in the *a priori* image modeling. We have also used our previously proposed Double Constrained Compound MRF model as the image prior. The image labels have been estimated using MAP estimation criterion and the model parameters estimation problem has been formulated in Maximum Conditional Pseudolikelihood (MCPL) framework. The Maximum a Posteriori(MAP) estimates of the labels have been obtained by a hybrid algorithm consisting of Simulated Annealing(SA) and Iterated Conditional Mode(ICM) algorithm. The proposed schemes, when compared with Yu 's method exhibited improved performance.

1 Introduction

The problem of object detection, shape analysis etc necessiates the first step of image segmentation. In real world scenario, color image segmentation provides adequate information for the above tasks. In such tasks, both image model and color model play a vital role. Color models such as RGB, HSV, YIQ,

$Ohta(I_1, I_2, I_3)$, $CIE(XYZ, Luv, Lab)$ are used as color models for different applications. Extensive use of HSV and I_1, I_2, I_3 models have been reported in literature. Specifically the linear model, i.e. Ohta model has been a very appropriate model for image analysis [2].

In model based approach, the accuracy of the segmented image greatly depends upon the use of appropriate image model. Stochastic models, particularly MRF models, have been successfully used as the image model for image restoration and segmentation [1,5,3,4]. Preservation of edges has been the important

B.K. Panigrahi et al. (Eds.): SEMCCO 2012, LNCS 7677, pp. 802–809, 2012.

criterion in image segmentation. A Graduated Edge Penalty function has been proposed [5] to preserve the weak edges together with the strong edges. Often, the segmentation problem is cast using MAP criterion and the MAP estimates are obtained by Simulated Annealing(SA) algorithm. The previously proposed Double Constrained MRF model by Panda *et al* [10] has been used as the image model. The model parameters can be estimated in both supervised and unsupervised framework. Homotopy continuation methods are globally convergent methods that have been used to trace the zeros of the map and hence to determine the solution of function [6,7]. In this work, attempts have been made to segment color images in unsupervised framework while preserving both strong as well as weak edges. The model parameters have been estimated by Homotopy Continuation method while the MAP estimates of the image labels have been estimated using hybrid algorithm. The hybrid algorithm consists of both SA and ICM algorithm. Initially SA is run to localize the solution and thereafter ICM is run to determine the solution. The hybrid algorithm is found to converge much faster than SA algorithm.

2 Image Model

2.1 Compound MRF Model

The compound MRF model has been proposed by Panda et al [10] where the model has been developed taking care of the (i) intra color-plane entities, and (ii)inter color plane interactions among the color planes. In case of Ohta (I_1, I_2, I_3) color model, each color plane is modeled as MRF model and the inter color plane interactions is also modeled as MRF thus resulting in Compound MRF model. The proposed compound MRF model is based on the following notion: (i)Intra-color-plane (I_1 or I_2 or I_3) enities of each color plane are modeled as MRF model (ii)Inter-color-plane interactions among color planes for e.g.(I_1 and I_2 and I_3), are also modeled as MRF.

We assume all images to be defined on discrete rectangular lattice M x N. In the following the Compound MRF model is developed. Let the observed image X be modeled as a random field and x is a realization which is the given image. Let Z denote the label process associated with the segmented image. We have considered the Ohta color space as our color model. Each color plane i.e.I_1,I_2 and I_3 is modeled by a MRF model. Let L denote the number of labels. For a given plane for example Z^1, if the spatial interactions are modeled by MRF, then the prior probability distribution $P(Z^1 = z^1)$ is Gibbs distributed and can be expressed as $P(Z^1 = z^1|\theta) = \frac{1}{Z'} e^{-U(z^1,\theta)}$, where, $Z' = \sum_{z^1} e^{-U(z^1,\theta)}$ is the partition function, $U(z^1, \theta)$ is called the energy function and is of the form $U(z^1, \theta) = \Sigma_{c \in C} V_c(z^1, \theta)$, with $V_c(z^1, \theta)$ being referred as the clique potential function and θ is the associated clique parameter vector. Analogously the spatial interactions of I_2 and I_3 planes can be defined. This prior MRF model taking care of all the three spatial planes would result in the energy function

$U(z,\theta) = \sum_{i,j} V_c(z,\theta)$, where, $V_c(z,\theta)$ denotes the clique potential function for the three spatial planes I_1,I_2 and I_3 respectively. However, the model is not complete for the color model. We model Z as a compound MRF, where the spatial interactions of individual color planes are taken care together with the inter color plane interactions of pixels. If this inter color plane interactions need to be modeled with the MRF prior, we can express $P(Z^2_{i,j} = z^2_{i,j} | Z^1_{k,l} = z^1_{k,l}, (k,l) \neq (i,j), \forall (k,l) \in M \times N) = P(Z^2_{i,j} = z^2_{i,j} | Z^1_{k,l} = z^1_{k,l}, (k,l) \neq (i,j), (k,l) \in \eta^1_{i,j})$.

Let z denote the labels of pixels taking care of all three color planes. In otherwords, z denotes the labels for pixels of the color image. For example, $z_{i,j}$ corresponds to the $(i,j)^{th}$ pixel label consisting of three color components. The prior probability of z has been contributed by the intra color plane interactions and inter color plane interactions of pixels. Hence, the prior model of z consists of the clique potential functions $V_{c_s}(z)$ and $V_{c_t}(z)$ corresponding to intra color plane interactions and inter color plane interactions respectively. The vertical and horizontal line fields for different color planes ($k = 1, 2, 3$) are denoted as v^k and h^k respectively.The model parameter θ has been estimated using Homotopy Continuation method proposed by Panda *et al* [10].

2.2 Double Constrained Compound Markov Random Field (DCCMRF) Model

The notions of the Constrained MRF model has been fused with the notion of Compound Model to develop a new model known as Constrained Compound Model [11]. The Constrained MRF model has been proposed by reinforcing on the local neighbourhood. The energy function for the Constrained model is given by

$$U_{sc}(z) = \sum_{i,j} U_s(z_{s_{i,j}}) + \lambda_c \left\{ z_{i,j_{avg}} - \sum_{z_{i,j} \in L} z_{i,j} \frac{e^{-U(z_{i,j})}}{\sum_{z_{i,j} \in L} e^{-U(z_{i,j})}} \right\}^2 \quad (1)$$

Where

$$U_{sc}(z_{i,j}) = V_{c_s}(z_{i,j}) + \lambda_c \left\{ z_{i,j_{avg}} - \sum_{z_{i,j} \in L} z_{i,j} \frac{e^{-U(z_{i,j})}}{\sum_{z_{i,j} \in L} e^{-U(z_{i,j})}} \right\}^2 \quad (2)$$

Where U_{sc} denote the energy function corresponding to intra color plane interactions. Where, $z_{i,j_{avg}}$ is the average of the neighouring pixels and λ_c is the constrained model parameter. The energy function taking care of both intra-color-plane and inter-color-plane interactions with intra plane constraints is given by

$$U(z) = U_{sc}(z_{i,j}, \theta) + U_{tc}(z_{i,j}, \theta) \quad (3)$$

3 Graduated Edge Penalty Function with Bi-level Line Field

3.1 MRF Model with Bi-level Line Field and Varying Edge Penalty Function

Often, in practice, a given scene may have edges which are weak enough to be detected by the single line field model. In such situations, it is required to reinforce the weak edges together with the strong edges. Qiyao Yu and D.A.Clausi have introduced the notion of Graduated Increased Edge Penalty (GIEP) to penalize the strong edges more than the weak edges in the MRF modeling [5,9]. They have proposed the edge penalty function to be any monotonically decreasing function. In order to preserve the weak edges together with the strong edges, we introduce a notion of multi-stage line fields in the *a priori* image modeling. We model the weak edges as one binary random field and the strong edges as another binary random field. Thus the image is modeled as MRF with two line fields in it. The corresponding MRF model is given by equation (4). The Line fields $h_{i,j}$,$v_{i,j}$ in equation (4) corresponds to the strong edges of the image and $h^{'}_{i,j}$,$v^{'}_{i,j}$ corresponds to the weak edges of the image. The last term corresponds to the Clique Potential function resulting due to intra-color-plane interactions. The **Two-stage Edge Penalty** function is given as follows

$$U(z,h,v) = \sum_{i,j} \alpha \left[\| z_{i,j} - z_{i,j-1} \|^2 (1 - v_{i,j}) \right.$$
$$\left. (1 - v^{'}_{i,j}) + \| z_{i,j} - z_{i-1,j} \|^2 (1 - h_{i,j})(1 - h^{'}_{i,j}) \right]$$
$$+ \beta_1 \left[v_{i,j} + h_{i,j} \right] + \beta_2 \left[v^{'}_{i,j} + h^{'}_{i,j} \right] \quad (4)$$

The notion of Graduated Edge Penalty is introduced where the strong edges are more penalized than the weak edges. The edge penalty function is Gaussian Weighted as given by the following equation, where the strong edges are more penalized than the weaker ones. In the above equation, $h_{i,j}$,$v_{i,j}$ corresponds to the strong edges of the image and $h^{'}_{i,j}$,$v^{'}_{i,j}$ corresponds to the weak edges. The threshold for the strong and weak edges are selected on trial and error basis. This Gaussian Weighted penalty function could reinforce the weaker edges while preserving the strong edges as well. The a posteriori energy function for **Bi-level Binary Field with Gaussian Weighted penalty function** is expressed as follows :

$$U(z,h,v) = \sum_{i,j} \alpha \left[\| z_{i,j} - z_{i,j-1} \|^2 \right.$$
$$(1 - v_{i,j})(1 - v^{'}_{i,j}) + \| z_{i,j} - z_{i-1,j} \|^2$$
$$\left. (1 - h_{i,j})(1 - h^{'}_{i,j}) \right] + \beta_1 / \sqrt{2\pi\sigma_1^2} \{ h_{i,j}$$
$$exp^{-(\Delta z_1)^2} / 2\sigma_1^2 +$$

$$v_{i,j} exp^{-(\Delta z_2)^2}/2\sigma_2^2\} + \beta_2/\sqrt{2\pi\sigma_2^2}\{h'_{i,j}$$
$$exp^{-(\Delta z_1)^2}/2\sigma_1^2 + v'_{i,j} exp^{-(\Delta z_2)^2}/2\sigma_2^2\} \quad (5)$$

4 Color Image Segmentation

Given a color image, let Z denote the label process and z is a realization. Z is modeled as the Double Constrained Compound MRF (CMRF) model and θ denotes the associated model parameters. In supervised segmentation scheme, by and large, the labels are the MAP estimates assuming the estimates of the associated model parameter $\hat{\theta}$ to be available. In this scheme, the MAP estimates of the labels and the estimates of the model parameters are carried out concurrently in unsupervised framework. In this regard, an estimation strategy need to be developed, which using the observed image, X,will yield an optimal pair (Z^{opt}, θ^{opt}). We consider the following joint optimality criterion,

$$(Z^{opt}, \theta^{opt}) = arg \max_{z,\theta} P(Z = z|X = x, \theta) \quad (6)$$

The estimated pair (Z^{opt}, θ^{opt}) satisfying (6) is the global optima of $P(Z = z/X = x, \theta)$ with respect to Z and θ. Since both Z and θ are unknown and interdependent, the problem is a hard problem. Therefore, sub-optimal solution could be an alternative. it is necessary to opt for strategies for suboptimal solution. In (6), z, θ could be viewed as a set of parameter of the given function $P(Z = z/X = x, \theta)$. For such kind of problems in deterministic framework, Wendell and Horter[8] have proposed an alternate approach that would yield suboptimal solutions instead of optimal solution. In [8] the variables have been split and recursively estimated. The final estimate in this process is called as the *partial optimal solution*. In our case, in stochastic framework, we in the same spirit have split the original problem into estimation of labels(z) and estimation of parameters θ to obtain the partial optimal solutions The splitting of the variables is expressed as follows

$$(Z^*) = arg \max_{z} P(Z = z|X = x, \theta^*) \quad (7)$$

$$(\theta^*) = arg \max_{\theta} P(Z = z^*|X = x, \theta^*) \quad (8)$$

Where $Z^* and \theta^*$ are the partial optimal solutions and are not global maxima, rather they are almost always local optimal solutions [8]. But with $\theta = \theta^*$, the estimate z^* is global optimal satisfying equation (7) and analogously for $z = z^*$, θ^* is global optimal satisfying equation(8). Since neither θ^* nor z^* is known, this is also hard to be solved. Towards this end, a recursive scheme is adopted where the model parameter estimation and segmentation is alternated. Let at the k^{th} iteration $\theta^k = [\alpha^k, \beta^k]^T$ be the estimate of model parameters and z^k be the estimate of the labels of the observed image. We adopt the following recursion

$$(Z^{k+1}) = arg \max_{z} P(Z = z|X = x, \theta^k) \quad (9)$$

$$(\theta^{k+1}) = arg \max_{\theta} P(Z = z^{k+1} | X = x, \theta) \tag{10}$$

The first problem of equation (9) is solved using Bayesian approach [3] and the DCCMRF model. The optimal value of θ^k is obtained by the proposed Homotopy Continuation method. The hybrid algorithm has been used to obtain the MAP estimates. One estimate of z^k and θ^k constitute one combined iteration. This recursion is continued for finite number of steps to obtain z^* and θ^*, the partial optimal solutions.

5 Hybrid Algorithm

It is observed that SA algorithm takes substantial amount of time to converge to the global optimum solution. SA algorithm has the attribute of coming out of the local minima and converging to the global optimal solution. This feature could be attributed to the acceptance criterion(acceptance with a probability). We have exploited this feature, that is the proposed hybrid algorithm uses the notion of acceptance criterion to come out of the local minima and to be near the global optimal solution. Thus, in the hybrid algorithm, SA algorithm produces an intermediate solution that can be local to the optimal solution. In order to obtain the optimal solution, a local convergence based strategy is adopted for quick convergence. Towards this end, we have used Iterated Conditional Mode (ICM) algorithm as the locally convergent algorithm. Thus, the proposed algorithm is a hybrid of both SA algorithm and ICM algorithm. The hybrid algorithm's working principle is as follows. Initially, a specific number of time steps of SA algorithm, fixed by trial and error, are executed to achieve the near optimal solution. Thereafter, ICM is run to converge to the desired optimal solution.

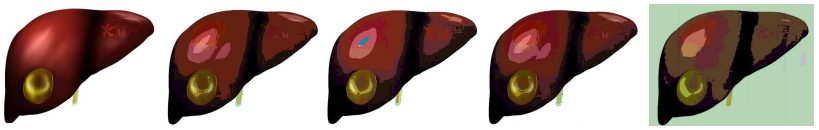
6 Simulation

Two images having both weak and strong edges have been considered and are shown in Fig. 1(a) and Fig. 2(a). The first image , the crow image, has been taken from the Berkeley image Data Base for segmentation, where nonuniform lighting and weak edges are prevalent. The corresponding ground truth images have been constructed manually and are shown in Fig. 1(b). The estimated MRF model parameters are $\alpha = 0.1023$ and $\beta = 2.25$. However σ has been choosen by trial and error and is fixed at 0.5. Fig. 1(c) corresponds to the MRF model and it can be seen that the weak edges could not be preserved. As observed from Fig. 1(d), the weak edges could be extracted properly. DCCMRF model could segment the image and preserve the weak edges as well. As shown in Fig. 1 Yu 's method has resulted in preservation of some strong edges but not all the weak edges. The misclassification error in case of MRF and DCCMRF are 16.26 and 3.11 respectively. The misclassification error for Yu 's method is 10.12. The second image considered is a biomedical image i.e Liver Abscisa image as shown in Fig. 2(a). In this image there are many weak edges together eith strong edges.

As observed from Fig. 2 (c) and (e), the MRF model and Yu 's method produced results where the weak edges have been merged together with the strong edges. But in case of DCCMRF model, the edges could be extracted, thus a small area has been identified within an oval shaped portion. This observation has also been reflected in the percentage of misclassification error. The MRF, DCCMRF and Yu 's method have misclassification error 15.27, 1.97 and 4.56 respectively. The proposed DCCMRF based scheme outperformed the MRF based and Yu's based schemes.

(a) (b) (c) (d) (e)

Fig. 1. (a)*Berkley Crow (670 x 448)* (b)Ground Truth (c)MRF optimized using Hybrid (d)DCCMRF optimized using Hybrid (e)Yu 's method



(a) (b) (c) (d) (e)

Fig. 2. (a)*Liver Abscess image(468 x 345)* (b)Ground Truth (c)MRF optimized using Hybrid (d)DCCMRF optimized using Hybrid (e)Yu 's method

7 Conclusions

A color image segmentation scheme has been proposed to preserve the weak edges together with the strong edges. The DCCMRF model with bi-level lne field could produce segmentation preserving the ill defined edges. The model parameters could also be estimated using unsupervised algorithm. The hybrid algorithm has been used to speed up the process of MAP estimation. The only limitation of the scheme is to choose a proper value of σ for the degradation process. Current attempts are made for multilevel linefield based DCCMRF model and estimation of the degradation parameter σ.

References

1. Cheng, H.D., Jiang, X.H., Sun, Y., Wang, J.: Color Image Segmentation: Advances and prospects. Pattern Recog. 34, 2259–2281 (2001)
2. Ohta, Y.I., Kanade, T., Sakai, T.: Color information for region segmentation. Comp. Grap. Image. Process. 62, 222–241 (1980)
3. Geman, S., Geman, D.: Stochastic relaxation,Gibbs distributions and the Bayesian restoration of images. IEEE. Trans. on PAMI 6, 721–741 (1984)

4. Kato, Z., Pong, T.C., Lee, J.C.M.: Color image segmentation and parameter estimation in a markovian framework. Pattern Recognition Letters 22, 309–321 (2001)
5. Yu, Q., Clausi, D.A.: IRGS: Image Segmentation Using Edge Penalties and Region Merging. IEEE Transaction on Pattern Analysis and Machine Intelligence PAMI 30(12), 2126–2139 (2008)
6. Chow, N., Mallet-Paret, J., Yorke, J.A.: Finding zeros of maps: homotopy methods that are constructive with probability one. Math. Computation 32(143), 887–899 (1978)
7. Stonick, V.L., Alexander, S.T.: A Relationship between recursive least square update and homotopy continuation methods. IEEE Trans. Signal Processing 39(2), 530–532 (1991)
8. Wendell, R.E., Horter Jr., A.P.: Minimization of a non-separable objective function subject to disjoint constraints. Operations Research 24(4), 643–657 (1976)
9. Mishra, A.K., Fieguth, P.W., Clausi, D.A.: Decoupled Active Contour (DAC) for Boundary Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 33(5), 917–930 (2011)
10. Sucheta, P., Nanda, P.K.: Color Image using Constrained Compound Markov Random Field Model and Homotopy Continuation Method. In: Proc. of the First International Conference on Distributed Frameworks and Applications, Universiti Sains Malaysia, Penang, Malaysia, pp. 151–158 (2008)
11. Sucheta, P., Nanda, P.K.: Constrained compound Markov Random field model with graduated penalty function for color Image Segmentation. In: IEEE International Conference on Control, Robotics and Cybernetics (ICCRC 2011), pp. VI-126–VI-132 (2011)

Author Index

Abdelaziz, Almoataz Y. 406, 548
Acharyya, Arnav 484
Agrawal, Sanjay 82
Albert Singh, N. 41
Ali, Layak 66
Anavatti, Sreenatha 356
Attia, M.A. 548

Badr, M.L.A. 406
Banerjee, Dhrubojyoti 248
Bansal, Ramesh 398
Basu, Debabrota 687
Behera, Santi 388
Benala, Tirimula Rao 124
Bhattacharya, Bidishna 106, 215
Biswal, B.B. 90
Biswas, Animesh 442
Biswas, Subhodip 151, 160, 189, 459, 467, 611
Bobby, Thomas Christy 594
Bose, Digbalay 151, 160, 189, 459, 467, 611

Card, Stuart W. 585
Cervone, Guido 721
Chakraborty, Niladri 106, 116, 215
Chakraborty, Nilanjan 662, 713
Chaudhuri, Sheli Sinha 17, 181
Chowdhury, Aritra 746, 770
Chowdhury, Jaydeep Ghosh 746, 770
Chua, Tay Jin 9
Coletti, Mark 721

da Cunha, Danilo Souza 628
Dan, Mainak 636
Das, Manideepto 577
Das, Asit Kumar 294, 602, 705
Das, Deblina 433
Das, Sudipta 33
Das, Swagatam 17, 160, 232, 276, 331, 425, 433, 459, 467, 611, 644, 687, 746, 778, 785
Dash, Subhransu Sekhar 259, 793
Dash, Subrat K. 133
De, Arnab Kumar 442
Deb, Debayan 1
Deb, Kalyanmoy 1, 484
Debchoudhury, Shantanab 267, 620
de Castro, Leandro Nunes 143, 628
Dehuri, Satchidananda 124, 492
Devaraj, D. 793
Diwakar, P.G. 49

El-Sharkawy, M.A. 548

Fazendeiro, Paulo 670, 729
Ferrari, Daniel G. 143

Ganguly, Srinjoy 687, 778
Gao, Kaizhou Z. 9
Garratt, Matthew 356
Geetha, Mohanasuram 169
Ghosh, Ashish 492
Ghosh, Shameek 364
Ghoshal, Sakti Prasad 33, 451
Guardieiro, Paulo R. 98
Gulammohien, G.M. Kadhar Nawaz 169
Gupta, Rohan 738

Halder, Udit 25, 181, 425
Hanmandlu, M. 738
Hardel, Gopi Ram 451
Hazra, Ranjay 372

Jana, Nanda Dulal 417
Janarthanan, R. 248
Jayaraman, V.K. 364
Jegatheesan, R. 679
Jerin Leno, I. 500

Kalia, Harihar 492
Kalyankar, Vivek D. 540
Kar, Rajib 33, 451
Kar, Sanjeeb Kumar 566
Kavitha, Ganesan 380
Khattab, H.M. 406
Kolling, Andreas 662, 713
Konar, Amit 248
Krishnanand, K.R. 476

Kuber, Karthik 585
Kumar, Ajay 372, 754
Kumar, Amioy 738
Kumar, Piyush 372
Kumar, Sajjan 116
Kumari, R. Sheela 41
Kundu, Rupam 267, 276, 620, 644
Kundu, Souvik 151, 160, 189, 459, 467, 611

Laha, Dipak 198
Latha, K. 519
Lewis, Michael 713
Lewis, Mike 662

Mahanand, B.S. 348
Mahapatra, Rosalin 285
Maity, Dipankar 25, 181, 425
Majhi, Babita 285
Mall, Rajib 124
Mallick, Manas Kumar 232, 476
Mandal, Durbadal 33, 451
Mandal, Kamal Krishna 106, 116, 215
Mani, V. 49
Manupati, V.K. 754
Mathuranath, P.S. 41
Maulik, Ujjwal 323
Mazumder, Sahisnu 509, 528
Mehrotra, Kishan G. 585
Mekhamer, S.F. 406
Mishra, Debadutta 90
Mishra, Nandan 566
Mitra, Pallavi 636
Mitra, Rahul 509, 528
Mohan, Chilukuri K. 585
Mohanty, Mihir Narayan 697
Mohanty, Pradeep Kumar 566
Mohanty, Prases K. 240
Mohapatra, Ankita 398
Mohapatra, Priyabrata 754
Mohapatra, Usha Manasi 285
MohitGolchha, S. 224
Mukherjee, Rohan 267, 276, 620, 644
Mukherjee, Swahum 687, 778

Naik, Anima 761
Nanda, P.K. 802
Nasir, Md. 785
Naveen, Nekuri 206
Nayak, Manas R. 133
Nekuri, Naveen 577
Neogi, Satyajit 433
Nithya, C. 793
Nunnally, Steven 662, 713

Omkar, S.N. 49

Padmavathi, S. 224
Padole, Chandrashekhar 670, 729
Pal, Diptendu 116
Panda, Rutuparna 82
Panda, Sidhartha 566
Panda, Sumanta 90
Panigrahi, Bijaya Ketan 232, 398, 406, 425, 476, 548, 738
Pant, Millie 339
Parhi, Dayal R. 240
Parvathi, K. 761
Pati, Soumen Kumar 294, 602
Phukan, Ripunjoy 57, 314
Ponnambalam, S.G. 500
Prasad, Shitala 372
Prasanthi, V.L. 124
Prata, Paula 670, 729

Raja, Nadaradjane Sri Madhava 380
Rajasekhar, Anguluri 232, 331
Rajendra, Ritwik 49
Rajinikanth, V. 519
Rakshit, Pratyusha 248
Ramachandran, Nayana 364
Ramakrishnan, Swaminathan 380, 594
Ramanand, K.R. 476
Ramaswamy, Savitha 348
Rao, C. Raghavendra 206
Rathinam, Ananthanaryanan 57, 314
Ravi, Vadlamani 74, 206, 577
Reddy, Kalam Narendar 74
Roselyn, J. Preetha 793
Rout, Minakhi 285
Rout, Pravat Kumar 133
Routray, Aurobinda 697
Roy, Preetam 116
Roy, Sayan Basu 636

Sabat, Samrat L. 66
Saha, Indrajit 323
Sahu, Binod Kumar 566
Samal, Mahendra Kumar 356
Saravanan, S. 169

Saravana Sankar, S. 500
Sarkar, Jnanendra Prasad 323
Sarkar, Soham 17
Satapathy, Suresh Chandra 761
Sattianadan, D. 259
SeeniMohamed, A. 224
Sengupta, Nandita 509, 528
Sengupta, Shampa 705
Sengupta, Soumyadip 785
Senthilnath, J. 49
Sequeira, Pedro 670, 729
Sevakula, Rahul K. 303
Sharma, Akshay 738
Sharma, Tarun Kumar 339
Sharma, Tuhin 509, 528
Shreyas, P.B. 49
Si, Tapas 417
Sil, Jaya 417, 509, 528
Silva, Éderson R. 98
Singh, Bhim 398
Singh, Raj Mohan 558
Singh, V.P. 339
Subramanian, Kartick 348
Sucheta, Panda 802
Sudhakaran, M. 259
Suganthan, Ponnuthurai Nagaratnam 9
Sundaram, Suresh 348
Sur, Arghya 746, 770
Sycara, Katia 662, 713

Tamura, Kenichi 653
Thakkar, J.J. 754
Tiwari, M.K. 754
Tripathy, Manish 388
Tudu, Bhimsen 116, 215

Udgata, Siba K. 66

Varghese, Tinu 41
Venkata Rao, Ravipudi 540
Venkateshwari, C. 364
Verma, Nishchal K. 303
Vijayakumar, K. 259, 679

Walker, Phillip 662, 713

Yasuda, Keiichiro 653