Ying Tan
Yuhui Shi
Zhen Ji (Eds.)

LNCS 7331

Advances in Swarm Intelligence

Third International Conference, ICSI 2012
Shenzhen, China, June 2012
Proceedings, Part I

Springer

Lecture Notes in Computer Science 7331

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

Ying Tan Yuhui Shi Zhen Ji (Eds.)

Advances in Swarm Intelligence

Third International Conference, ICSI 2012
Shenzhen, China, June 17-20, 2012
Proceedings, Part I

Springer

Volume Editors

Ying Tan
Peking University
Key Laboratory of Machine Perception (MOE)
Department of Machine Intelligence
School of Electronics Engineering and Computer Science
Beijing 100871, China
E-mail: ytan@pku.edu.cn

Yuhui Shi
Xi'an Jiaotong-Liverpool University
Department of Electrical and Electronic Engineering
Suzhou 215123, China
E-mail: yuhui.shi@xjtlu.edu.cn

Zhen Ji
Shenzhen University
College of Computer Science and Software Engineering
Shenzhen 518060, China
E-mail: jizhen@szu.edu.cn

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-30975-5 e-ISBN 978-3-642-30976-2
DOI 10.1007/978-3-642-30976-2
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012939129

CR Subject Classification (1998): F.1, H.3, I.2, H.4, H.2.8, I.4-5

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 book and its companion volume, LNCS vols. 7331 and 7332, constitute the proceedings of the Third International Conference on Swarm Intelligence (ICSI 2012) held during June 17–20, 2012, in Shenzhen, China. ICSI 2012 was the third international gathering in the world for researchers working on all aspects of swarm intelligence, following the successful and fruitful Beijing event (ICSI 2010) and Chongqing event (ICSI 2011), which provided a high-level academic forum for the participants to disseminate their new research findings and discuss emerging areas of research. It also created a stimulating environment for the participants to interact and exchange information on future challenges and opportunities in the field of swarm intelligence research.

ICSI 2012 received 247 submissions and 10 invited papers from about 591 authors in 24 countries and regions (Algeria, Australia, Brazil, China, France, Hong Kong, India, Islamic Republic of Iran, Japan, Republic of Korea, Kuwait, Macau, Malaysia, Mexico, Russian Federation, Saudi Arabia, Singapore, South Africa, South Sudan, Chinese Taiwan, Tunisia, Turkey, UK, USA) across six continents (Asia, Europe, North America, South America, Africa, and Oceania). Each submission was reviewed by at least two reviewers, and on average 2.6 reviewers. Based on rigorous reviews by the Program Committee members and reviewers, 145 high-quality papers were selected for publication in this proceedings volume with an acceptance rate of 56.4%. The papers are organized in 27 cohesive sections covering all major topics of swarm intelligence research and development.

In addition to the contributed papers, the ICSI 2012 technical program included three plenary speeches by Xin Yao (The University of Birmingham, UK, IEEE Fellow, Vice President of IEEE Computational Intelligence), Carlos Artemio Coello Coello (NCINVESTAV-IPNl, Mexico, IEEE Fellow), and Guang-Bin Huang (Nanyang Technological University, Singapore, inventor of Extreme Learning Machine). Besides the regular oral sessions, ICSI 2012 had one special session on "Data Fusion and Computational Intelligence" and several poster sessions focusing on diverse areas.

As organizers of ICSI 2012, we would like to express sincere thanks to Shenzhen University, Peking University, and Xi'an Jiaotong-Liverpool University for their sponsorship, as well as to the IEEE Computational Intelligence Society, World Federation on Soft Computing, and International Neural Network Society for their technical co-sponsorship. We appreciate the Natural Science Foundation of China for its financial and logistic support.

We would also like to thank the members of the Advisory Committee for their guidance, the members of the International Program Committee and additional reviewers for reviewing the papers, and the members of the Publications Committee for checking the accepted papers in a short period of time. Particularly,

we are grateful to the proceedings publisher Springer for publishing the proceedings in the prestigious series of *Lecture Notes in Computer Science*. Moreover, we wish to express our heartfelt appreciation to the plenary speakers, session chairs, and student helpers. In addition, there are still many more colleagues, associates, friends, and supporters who helped us in immeasurable ways; we express our sincere gratitude to them all. Last but not the least, we would like to thank all the speakers, authors, and participants for their great contributions that made ICSI 2012 successful and all the hard work worthwhile.

June 2012

Ying Tan
Yuhui Shi
Zhen Ji

Organization

General Chairs

Ying Tan	Peking University, China
Zhen Ji	Shenzhen University, China

Program Committee Chair

Yuhui Shi	Xi'an Jiaotong-Liverpool University, China

Advisory Committee Chairs

Guoliang Chen	Shenzhen University, Shenzhen, China
Russell C. Eberhart	Indiana University-Purdue University, USA

Technical Committee Chairs

Zexuan Zhu	Shenzhen University, China
Qing-hua Wu	University of Liverpool, UK
Kalyanmoy Deb	Indian Institute of Technology, India
Andries Engelbrecht	University of Pretoria, South Africa
Ram Akella	University of California, USA
Jose Alfredo Ferreira Costa	Federal University, Brazil

Plenary Sessions Chairs

Martin Middendorf	University of Leipzig, Germany
Jun Zhang	Sun Yat-Sen University, China

Special Sessions Chairs

Shan He	University of Birmingham, UK
Xiaodong Li	RMIT Unversity, Australia

Publications Chairs

Radu-Emil Precup	Politehnica University of Timisoara, Romania
Zhishun Wang	Columbia University, USA

Publicity Chairs

Eugene Santos Jr.	Thayer School of Engineering at Dartmouth, USA
Yew-Soon Ong	Nanyang Technological University, Singapore
Juan Luis Fernandez Martinez	University of Oviedo, Spain
Fernando Buarque	Universidade of Pernambuco, Brazil
Zhuhong You	Shenzhen University, China

Finance Chairs

Chao Deng	Peking University, China
Andreas Janecek	University of Vienna, Austria

Local Arrangements Chair

Jiarui Zhou	Shenzhen University, Shenzhen, China

Program Committee

Ram Akella	University of California, USA
Payman Arabshahi	University of Washington, USA
Sabri Arik	Istanbul University, Turkey
Carmelo J. A. Bastos Filho	University of Pernambuco, Brazil
Christian Blum	Universitat Politecnica de Catalunya, Spain
Salim Bouzerdoum	University of Wollongong, Australia
Walter W. Chen	National Taipei University of Technology, Taiwan
Manuel Chica	European Centre for Soft Computing, Spain
Leandro Coelho	Pontifícia Universidade Católica do Parana, Brazil
Carlos A. Coello Coello	CINVESTAV-IPN, Mexico
Jose Alfredo Ferreira Costa	UFRN Universidade Federal do Rio Grande do Norte, Brazil
Prithviraj Dasgupta	University of Nebraska, USA
Kalyanmoy Deb	Indian Institute of Technology, India
Kusum Deep	Indian Institute of Technology Roorkee, India
Mingcong Deng	Tokyo University of Agriculture and Technology, Japan
Yongsheng Ding	Donghua University, China
Haibin Duan	Beijing University of Aeronautics and Astronautics, China
Mark Embrechts	RPI, USA
Andries Engelbrecht	University of Pretoria, South Africa
Pan Feng	Beijing University of Technology, China
Yoshikazu Fukuyama	Fuji Electric Systems Co., Ltd. Japan
Wai-Keung Fung	University of Manitoba, Canada

Beatriz Aurora Garro Licon	CIC-IPN, Mexico
Dunwei Gong	China University of Mining and Technology, China
Ping Guo	Beijing Normal University, China
Walter Gutjahr	University of Vienna, Austria
Mo Hongwei	Harbin Engineering University, China
Jun Hu	Chinese Academy of Sciences, China
Guangbin Huang	Nanyang Technological University, Singapore
Hisao Ishibuchi	Osaka Prefecture University, Japan
Andreas Janecek	University of Vienna, Austria
Zhen Ji	Shenzhen University, China
Changan Jiang	Kagawa University, Japan
Colin Johnson	University of Kent, USA
Farrukh Khan	FAST-NUCES Islamabad, Pakistan
Arun Khosla	National Institute of Technology Jalandhar, India
David King	Nottingham Trent University, UK
Thanatchai Kulworawanichpong	Suranaree University of Technology, Thailand
Germano Lambert-Torres	Itajuba Federal University, Brazil
Xia Li	Shenzhen University, China
Xiaodong Li	RMIT University, Australia
Yangmin Li	University of Macau, Macao, China
Jane Liang	Zhengzhou University, China
Andrei Lihu	"Politehnica" University, Timisoara
Fernando B. De Lima Neto	University of Pernambuco, Brazil
Ju Liu	Shandong University, China
Qun Liu	Chongqing University of Posts and Communications, China
Wenlian Lu	Fudan University, China
Xiaoqiang Lu	Dalian University of Technology, China
Wenjian Luo	University of Science and Technology of China
Jinwen Ma	Peking university, China
Xiujun Ma	Peking University, China
Juan Luis Fernandez Martinez	University of California Berkeley, USA
Bernd Meyer	Monash University, Australia
Martin Middendorf	University of Leipzig, Germany
Mahamed G.H. Omran	Gulf University for Science and Technology, Kuwait
Thomas Potok	Oak Ridge National Laboratory, USA
Radu-Emil Precup	Politehnica University of Timisoara, Romania
Yuhui Shi	Xi'an Jiaotong-Liverpool University, China
Michael Small	Hong Kong Polytechnic University, Hong Kong, China
Ponnuthurai Suganthan	Nanyang Technological University, Singapore
Norikazu Takahashi	Kyushu University, Japan

Kay Chen Tan	National University of Singapore, Singapore
Ying Tan	Peking University, China
Peter Tino	University of Birmingham, UK
Christos Tjortjis	The University of Manchester, UK
Frans Van Den Bergh	CSIR SAC (Pretoria), South Africa
Bing Wang	University of Hull, UK
Guoyin Wang	Chongqing University of Posts and Telecommunications, China
Jiahai Wang	Sun Yat-sen University, China
Lei Wang	Tongji University, China
Ling Wang	Tsinghua University, China
Lipo Wang	Nanyang Technological University, Singapore
Qi Wang	Xi'an Institute of Optics and Precision Mechanics of CAS, China
Hongxing Wei	Beihang University, China
Shunren Xia	Zhejiang University, China
Zuo Xingquan	Beijing University of Posts and Telecommunications, China
Ning Xiong	Mälardalen University, Sweden
Benlian Xu	Changshu Institute of Technology, China
Xin-She Yang	National Physical Laboratory
Yingjie Yang	De Montfort University, UK
Gary Yen	Oklahoma State University, USA
Hoengpeng Yin	Chongqing University, China
Peng-Yeng Yin	National Chi Nan University, Taiwan, China
Jie Zhang	Newcastle University, UK
Jun Zhang	Waseda University, Japan
Jun Zhang	Sun Yat University, China
Junqi Zhang	Tongji University, China
Lifeng Zhang	Renmin University, China
Qieshi Zhang	Waseda University, Japan
Qingfu Zhang	University of Essex, UK
Yanqing Zhang	Georgia State University, USA
Dongbin Zhao	Chinese Academy of Science, China

Additional Reviewers

Barajas, Joel
Chen, Xinyu
Day, Rong-Fuh
Filipczuk, Paweł
Guo, Gege
Guruprasad, K.R.
Jhuo, I-Hong
Jinno, Kenya
Jumadinova, Janyl
Lenagh, William
Li, Fuhai
Murata, Junichi
Nakano, Hidehiro
Sun, Yang
Tong, Can
Wang, Chunye
Yu, Jian

Table of Contents – Part I

Swarm Intelligence Based Algorithms

Particle Swarm Optimization

Applications of PSO Algorithms

Ant Colony Optimization Algorithms

Biogeography-Based Optimization Algorithms

Novel Swarm-Based Optimization Algorithms

Artificial Immune System

Bee Colony Algorithms

Differential Evolution

Genetic Algorithms

Neural Networks and Fuzzy Methods

Hybrid Algorithms

Multi-Objective Optimization Algorithms

Multi-robot, Swarm-Robot and Multi-agent Systems

Table of Contents – Part II

Machine Learning Methods

Feature Extraction and Selection Algorithms

Data Mining Methods

Biometrics and Information Security

Pattern Recognition Methods

Intelligent Control

Wireless Sensor Network

Scheduling and Path Planning

Signal Processing

Visual Simulation and Parallel Implementation

Mathematics

Other Applications

Special Session on Data Fusion and Computational Intelligence

The Biological Interaction Stability Problem

Zvi Retchkiman Konigsberg

Instituto Politecnico Nacional
mzvi@cic.ipn.mx

Abstract. This paper addresses the biological interaction stability problem among organisms of the same or different species associated with the need for a common resource that occurs in a limited supply relative to demand by considering it as a discrete event dynamical system. Timed Petri nets are a graphical and mathematical modeling tool applicable to discrete event dynamical systems in order to represent its states evolution. Lyapunov stability theory provides the required tools needed to aboard the stability problem for discrete event dynamical systems modeled with timed Petri nets. By proving boundedness one confirms a dominant oscillating behavior of both organisms dynamics performance. However, the oscillating frequency results to be unknown. This inconvenience is overcome by considering a specific recurrence equation, in the max-plus algebra.

Keywords: Biological interaction, Discrete Event Dynamical Systems, Max-Plus Algebra, Lyapunov Method, Timed Petri Nets.

1 Introduction

Consider the biological interaction stability problem among organisms of the same or different species associated with the need for a common resource that occurs in a limited supply relative to demand. In other words, competition occurs when the capability of the environment to supply resources is smaller than the potential biological requirement so that organisms interfere with each other. Plants, for example, often compete for access to a limited supply of nutrients, water, sunlight, and space. In the study of this type of problems Lotka-Volterra models as well as evolutionary game theory concepts have been used [1,2]. This paper proposes a new modeling and analysis methodology which consists in considering the competition as a discrete event dynamical system. Timed Petri nets are a graphical and mathematical modeling tool applicable to discrete event systems in order to represent its states evolution where the timing at which the state changes is taken into consideration. Lyapunov stability theory provides the required tools needed to aboard the stability problem for discrete event system modeled with timed Petri nets whose mathematical model is given in terms of difference equations [5]. Employing Lyapunov methods, a sufficient condition for the stabilization problem is also obtained. It is shown that it is possible to restrict the discrete event systems state space in such a way that boundedness is

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 1–10, 2012.

guaranteed. By proving boundedness one confirms a dominant oscillating behavior of both organisms dynamics performance. However, the oscillating frequency results to be unknown. This inconvenience is overcome by considering a specific recurrence equation, in the max-plus algebra, which is assigned to the the timed Petri net graphical model. The main contribution of the paper consists in combining Lyapunov theory with max-plus algebra to study the biological interaction stability problem among organisms treated as discrete event dynamical systems modeled with timed Petri nets. The paper is organized as follows. In section 2, Lyapunov theory for discrete event modeled with Petri nets is addressed. Section 3, presents Max-Plus algebra. In section 4, the stability for discrete event dynamical systems modeled with timed Petri nets is given. Section 5, discusses the biological interaction stability problem. Finally, the paper ends with some conclusions.

2 Lyapunov Stability and Stabilization of Discrete Event Systems modeled with Petri Nets

NOTATION: $N = \{0, 1, 2, ...\}$, $R_+ = [0, \infty)$, $N_{n_0}^+ = \{n_0, n_0 + 1, ..., n_0 + k, ...\}$, $n_0 \geq 0$. Given $x, y \in R^n$, $x \leq y$ is equivalent to $x_i \leq y_i, \forall i$. A function $f(n, x)$, $f : N_{n_0}^+ \times R^n \to R^n$ is called nondecreasing in x if given $x, y \in R^n$ such that $x \geq y$ and $n \in N_{n_0}^+$ then, $f(n, x) \geq f(n, y)$. Consider systems of first ordinary difference equations given by

$$x(n+1) = f[n, x(n)], x(n_o) = x_0, n \in N_{n_0}^+ \tag{1}$$

where $n \in N_{n_0}^+$, $x(n) \in R^n$ and $f : N_{n_0}^+ \times R^n \to R^n$ is continuous in $x(n)$.

Definition 1. *The n vector valued function $\Phi(n, n_0, x_0)$ is said to be a solution of (1) if $\Phi(n_0, n_0, x_0) = x_0$ and $\Phi(n+1, n_0, x_0) = f(n, \Phi(n, n_0, x_0))$ for all $n \in N_{n_0}^+$.*

Definition 2. *The system (1) is said to be i). Practically stable, if given (λ, A) with $0 < \lambda < A$, then*

$$|x_0| < \lambda \Rightarrow |x(n, n_0, x_0)| < A, \forall n \in N_{n_0}^+, n_0 \geq 0;$$

ii). Uniformly practically stable, if it is practically stable for every $n_0 \geq 0$.

Definition 3. *A continuous function $\alpha : [0, \infty) \to [0, \infty)$ is said to belong to class $\mathcal{K}$ if $\alpha(0) = 0$ and it is strictly increasing.*

Consider a vector Lyapunov function $v(n, x(n))$, $v : N_{n_0}^+ \times R^n \to R_+^p$ and define the variation of v relative to (1) by

$$\Delta v = v(n+1, x(n+1)) - v(n, x(n)) \tag{2}$$

Then, the following result concerns the practical stability of (1).

Theorem 1. *Let* $v : N_{n_0}^{+} \times R^n \to R_{+}^{p}$ *be a continuous function in* x*, define the function* $v_0(n, x(n)) = \sum_{i=1}^{p} v_i(n, x(n))$ *such that satisfies the estimates*

$$b(|x|) \leq v_0\,(n, x\,(n)) \leq a(|x|); a, b \in \mathcal{K}, \Delta v(n, x(n)) \leq w(n, v(n, x(n))) \tag{3}$$

for $n \in N_{n_0}^{+}$, $x(n) \in R^n$ *, where* $w : N_{n_0}^{+} \times R_{+}^{p} \to R^p$ *is a continuous function in the second argument. Assume that :* $g(n, e) \triangleq e + w(n, e)$ *is nondecreasing in* e*,* $0 < \lambda < A$ *are given and finally that* $a(\lambda) < b(A)$ *is satisfied. Then, the practical stability properties of*

$$e(n+1) = g(n, e(n)), e(n_0) = e_0 \geq 0. \tag{4}$$

imply the practical stability properties of system (1).

Corollary 1. *In Theorem (1): i). If* $w(n, e) \equiv 0$ *we get uniform practical stability of (1) which implies structural stability. ii). If* $w(n, e) = -c(e)$*, for* $c \in \mathcal{K}$*, we get uniform practical asymptotic stability of (1).*

Definition 4. *A Petri net is a 5-tuple,* $PN = \{P, T, F, W, M_0\}$ *where:* $P = \{p_1, p_2, ..., p_m\}$*is a finite set of places,* $T = \{t_1, t_2, ..., t_n\}$ *is a finite set of transitions,* $F \subset (P \times T) \cup (T \times P)$ *is a set of arcs,* $W : F \to N_1^{+}$ *is a weight function,* M_0*:* $P \to N$ *is the initial marking,* $P \cap T = \varnothing$ *and* $P \cup T \neq \varnothing$*.*

Definition 5. *The clock structure associated with a place* $p_i \in P$ *is a set* $\mathbf{V} = \{V_i : p_i \in P\ \}$ *of clock sequences* $V_i = \{v_{i,1}, v_{i,2}, ...\}$*,* $v_{i,k} \in R^{+}$*,* $k = 1, 2, ...$

The positive number $v_{i,k}$, associated to $p_i \in P$, called holding time, represents the time that a token must spend in this place until its outputs enabled transitions $t_{i,1}, t_{i,2}, ...$, fire. We partition P into subsets P_0 and P_h, where P_0 is the set of places with zero holding time, and P_h is the set of places that have some holding time.

Definition 6. *A timed Petri net is a 6-tuple* $TPN = \{P, T, F, W, M_0, \mathbf{V}\}$ *where* $\{P, T, F, W, M_0\}$ *are as before, and* $\mathbf{V} = \{V_i : p_i \in P\ \}$ *is a clock structure. A timed Petri net is a timed event petri net when every* $p_i \in P$ *has one input and one output transition, in which case the associated clock structure set of a place* $p_i \in P$ *reduces to one element* $V_i = \{v_i\}$

Notice that if $W(p, t) = \alpha$ (or $W(t, p) = \beta$) then, this is often represented graphically by α, (β) arcs from p to t (t to p) each with no numeric label.

Let $M_k(p_i)$ denote the marking (i.e., the number of tokens) at place $p_i \in P$ at time k and let $M_k = [M_k(p_1), ..., M_k(p_m)]^T$ denote the marking (state) of PN at time k. A transition $t_j \in T$ is said to be enabled at time k if $M_k(p_i) \geq W(p_i, t_j)$ for all $p_i \in P$ such that $(p_i, t_j) \in F$. It is assumed that at each time k there exists at least one transition to fire. If a transition is enabled then, it can fire. If an enabled transition $t_j \in T$ fires at time k then, the next marking for $p_i \in P$ is given by

$$M_{k+1}(p_i) = M_k(p_i) + W(t_j, p_i) - W(p_i, t_j). \tag{5}$$

Let $A = [a_{ij}]$ denote an $n \times m$ matrix of integers (the incidence matrix) where $a_{ij} = a_{ij}^{+} - a_{ij}^{-}$ with $a_{ij}^{+} = W(t_i, p_j)$ and $a_{ij}^{-} = W(p_j, t_i)$. Let $u_k \in \{0,1\}^n$ denote a firing vector where if $t_j \in T$ is fired then, its corresponding firing vector is $u_k = [0, ..., 0, 1, 0, ..., 0]^T$ with the one in the j^{th} position in the vector and zeros everywhere else. The nonlinear difference matrix equation describing the dynamical behavior represented by a PN is:

$$M_{k+1} = M_k + A^T u_k \tag{6}$$

where if at step k, $a_{ij}^{-} < M_k(p_j)$ for all $p_i \in P$ then, $t_i \in T$ is enabled and if this $t_i \in T$ fires then, its corresponding firing vector u_k is utilized in the difference equation to generate the next step. Notice that if M' can be reached from some other marking M and, if we fire some sequence of d transitions with corresponding firing vectors $u_0, u_1, ..., u_{d-1}$ we obtain that

$$M' = M + A^T u, u = \sum_{k=0}^{d-1} u_k. \tag{7}$$

Let $(N_{n_0}^m, d)$ be a metric space where $d : N_{n_0}^m \times N_{n_0}^m \to R_+$ is defined by

$$d(M_1, M_2) = \sum_{i=1}^{m} \zeta_i \mid M_1(p_i) - M_2(p_i) \mid; \zeta_i > 0$$

and consider the matrix difference equation which describes the dynamical behavior of the discrete event system modeled by a PN, see (7).

Proposition 1. *Let PN be a Petri net. PN is uniform practical stable if there exists a Φ strictly positive m vector such that*

$$\Delta v = u^T A \Phi \leq 0 \tag{8}$$

Moreover, PN is uniform practical asymptotic stable if the following equation holds

$$\Delta v = u^T A \Phi \leq -c(e), c \in \mathcal{K} \tag{9}$$

Lemma 1. *Let suppose that Proposition (1) holds then,*

$$\Delta v = u^T A \Phi \leq 0 \Leftrightarrow A\Phi \leq 0 \tag{10}$$

Remark 1. Notice that since the state space of a TPN is contained in the state space of the same now not timed PN, stability of PN implies stability of the TPN.

2.1 Lyapunov Stabilization

Definition 7. *Let PN be a Petri net. PN is said to be stabilizable if there exists a firing transition sequence with transition count vector u such that system (7) remains bounded.*

Proposition 2. *Let PN be a Petri net. PN is stabilizable if there exists a firing transition sequence with transition count vector u such that the following equation holds*

$$\Delta v = A^T u \leq 0 \tag{11}$$

Remark 2. By fixing a particular u, which satisfies (11), the state space is restricted to those markings that are finite.

3 Max-Plus Algebra

3.1 Basic Definitions

NOTATION: $\epsilon = -\infty$, $e = 0$, $\mathbb{R}_{max} = \mathbb{R} \cup \{\epsilon\}$, $\underline{n} = 1, 2, ..., n$. Let $a, b \in \mathbb{R}_{max}$ and define the operations $\oplus$ and $\otimes$ by: $a \oplus b = \max(a, b)$ and $a \otimes b = a + b$.

Definition 8. *The set $\mathbb{R}_{max}$ with the two operations $\oplus$ and $\otimes$ is called a max-plus algebra and is denoted by $\Re_{\max} = (\mathbb{R}_{max}, \oplus, \otimes, \epsilon, e)$.*

Definition 9. *A semiring is a nonempty set R endowed with two operations $\oplus_R$, $\otimes_R$, and two elements ϵ_R and e_R such that: $\oplus_R$ is associative and commutative with zero element ϵ_R, $\otimes_R$ is associative, distributes over $\oplus_R$, and has unit element e_R, $\in_R$is absorbing for $\otimes_R$ i.e., $a \otimes_R \epsilon = \epsilon_R \otimes a = a$, $\forall a \in R$..*

In addition if $\otimes_R$ is commutative then R is called a commutative semiring , and if $\oplus_R$ is such that $a \oplus_R a = a$, $\forall a \in R$ then it is called idempotent.

Theorem 2. *The max-plus algebra $\Re_{\max} = (\mathbb{R}_{max}, \oplus, \otimes, \epsilon, e)$ has the algebraic structure of a commutative and idempotent semiring.*

3.2 Matrices and Graphs

Let $\mathbb{R}_{max}^{n \times n}$ be the set of $n \times n$ matrices with coefficients in $\mathbb{R}_{max}$ with the following operations: The sum of matrices $A, B \in \mathbb{R}_{max}^{n \times n}$, denoted $A \oplus B$ is defined by: $(A \oplus B)_{ij} = a_{ij} \oplus b_{ij} = max(a_{ij}, b_{ij})$ for i and $j \in \underline{n}$. The product of matrices $A \in \mathbb{R}_{max}^{n \times l}, B \in \mathbb{R}_{max}^{l \times n}$, denoted $A \otimes B$ is defined by: $(A \otimes B)_{ik} = \bigotimes_{j=1}^{l} (a_{ij} \otimes b_{jk})$ for i and $k \in \underline{n}$. Let $\mathcal{E} \in \mathbb{R}_{max}^{n \times n}$ denote the matrix with all its elements equal to ϵ and denote by $E \in \mathbb{R}_{max}^{n \times n}$ the matrix which has its diagonal elements equal to e and all the other elements equal to ϵ. Then, the following result can be stated.

Theorem 3. *The 5-tuple $\Re_{\max}^{n \times n} = (\mathbb{R}_{max}^{n \times n}, \oplus, \otimes, \mathcal{E}, E)$ has the algebraic structure of a noncommutative idempotent semiring.*

Definition 10. *Let $A \in \mathbb{R}_{max}^{n \times n}$ and $k \in \mathbb{N}$ then the k-th power of A denoted by $A^{\otimes k}$ is defined by: $A^{\otimes k} = A \otimes A \otimes \cdots \otimes A, (k$ times$)$, where $A^{\otimes 0}$ is set equal to E.*

Definition 11. *A matrix $A \in \mathbb{R}_{max}^{n\times n}$ is said to be regular if A contains at least one element distinct from ϵ in each row.*

Definition 12. *Let $\mathcal{N}$ be a finite and non-empty set and consider $\mathcal{D} \subseteq \mathcal{N} \times \mathcal{N}$. The pair $G = (\mathcal{N}, \mathcal{D})$ is called a directed graph, where $\mathcal{N}$ is the set of elements called nodes and $\mathcal{D}$ is the set of ordered pairs of nodes called arcs. A directed graph $G = (\mathcal{N}, \mathcal{D})$ is called a weighted graph if a weight $w(i, j) \in \mathbb{R}$ is associated with any arc $(i, j) \in \mathcal{D}$.*

Let $A \in \mathbb{R}_{max}^{n\times n}$ be any matrix, a graph $\mathcal{G}(A)$, called the communication graph of A, can be associated as follows. Define $\mathcal{N}(A) = \underline{n}$ and a pair $(i, j) \in \underline{n} \times \underline{n}$ will be a member of $\mathcal{D}(A) \Leftrightarrow a_{ji} \neq \epsilon$, where $\mathcal{D}(A)$ denotes the set of arcs of $\mathcal{G}(A)$.

Definition 13. *A path from node i to node j is a sequence of arcs $p = \{(i_k, j_k) \in \mathcal{D}(A)\}_{k\in\underline{m}}$ such that $i = i_1, j_k = i_{k+1}$, for $k < m$ and $j_m = j$. The path p consists of the nodes $i = i_1, i_2, ..., i_m, j_m = j$ with length m denoted by $\mid p \mid_1 = m$. In the case when $i = j$ the path is said to be a circuit. A circuit is said to be elementary if nodes i_k and i_l are different for $k \neq l$. A circuit consisting of one arc is called a self-loop.*

Let us denote by $P(i, j; m)$ the set of all paths from node i to node j of length $m \geq 1$ and for any arc $(i, j) \in \mathcal{D}(A)$ let its weight be given by a_{ij} then the weight of a path $p \in P(i, j; m)$ denoted by $\mid p \mid_w$ is defined to be the sum of the weights of all the arcs that belong to the path. The average weight of a path p is given by $\mid p \mid_w \; / \mid p \mid_1$. Given two paths, as for example, $p = ((i_1, i_2), (i_2, i_3))$ and $q = ((i_3, i_4), ((i_4, i_5)$ in $\mathcal{G}(A)$ the concatenation of paths $\circ : \mathcal{G}(A) \times \mathcal{G}(A) \to \mathcal{G}(A)$ is defined as $p \circ q = ((i_1, i_2), (i_2, i_3), (i_3, i_4), (i_4, i_5))$. The communication graph $\mathcal{G}(A)$ and powers of matrix A are closely related as it is shown in the next theorem.

Theorem 4. *Let $A \in \mathbb{R}_{max}^{n\times n}$, then $\forall k \geq 1$: $[A^{\otimes k}]_{ji} = max\{\mid p \mid_w : p \in P(i, j; k)\}$, where $[A^{\otimes k}]_{ji} = \epsilon$ in the case when $P(i, j; k)$ is empty i.e., no path of length k from node i to node j exists in $\mathcal{G}(A)$.*

Definition 14. *Let $A \in \mathbb{R}_{max}^{n\times n}$ then define the matrix $A^+ \in \mathbb{R}_{max}^{n\times n}$ as: $A^+ = \bigoplus_{k=1}^{\infty} A^{\otimes k}$. Where the element $[A^+]_{ji}$ gives the maximal weight of any path from j to i. If in addition one wants to add the possibility of staying at a node then one must include matrix E in the definition of matrix A^+ giving rise to its Kleene star representation defined by: $A^* = \bigoplus_{k=0}^{\infty} A^{\otimes k}$.*

Lemma 2. *Let $A \in \mathbb{R}_{max}^{n\times n}$ be such that any circuit in $\mathcal{G}(A)$ has average circuit weight less than or equal to ϵ. Then it holds that: $A^* = \bigoplus_{k=0}^{n-1} A^{\otimes k}$.*

Definition 15. *Let $G = (\mathcal{N}, \mathcal{D})$ be a graph and $i, j \in \mathcal{N}$, node j is reachable from node i, denoted as $i\mathcal{R}j$, if there exists a path from i to j. A graph G is said to be strongly connected if $\forall i, j \in \mathcal{N}, j\mathcal{R}i$. A matrix $A \in \mathbb{R}_{max}^{n\times n}$ is called*

irreducible if its communication graph is strongly connected, when this is not the case matrix A is called reducible.

Definition 16. *Let $G = (\mathcal{N}, \mathcal{D})$ be a not strongly connected graph and $i, j \in \mathcal{N}$, node j communicates with node i, denoted as $i\mathcal{C}j$, if either $i = j$ or $i\mathcal{R}j$ and $j\mathcal{R}i$.*

The relation $i\mathcal{C}j$ defines an equivalence relation in the set of nodes, and therefore a partition of $\mathcal{N}$ into a disjoint union of subsets, the equivalence classes, $\mathcal{N}_1, \mathcal{N}_2, ..., \mathcal{N}_q$ such that $\mathcal{N} = \mathcal{N}_1 \cup \mathcal{N}_2 \cup ... \cup \mathcal{N}_q$ or $\mathcal{N} = \bigcup_{i \in \mathcal{N}} [i]; [i] = \{j \in \mathcal{N} : iCj\}$. Given the above partition, it is possible to focus on subgraphs of G denoted by $G_r = (\mathcal{N}_r, \mathcal{D}_r); r \in \underline{q}$ where $\mathcal{D}_r$ denotes the subset of arcs, which belong to $\mathcal{D}$, that have both the begin node and end node in $\mathcal{N}_r$. If $\mathcal{D}_r \neq \varnothing$, the subgraph $G_r = (\mathcal{N}_r, \mathcal{D}_r)$ is known as a maximal strongly connected subgraph of G.

Definition 17. *The reduced graph $\widetilde{G} = (\widetilde{\mathcal{N}}, \widetilde{\mathcal{D}})$ of G is defined by setting $\widetilde{\mathcal{N}} = \{[i_1], [i_2], ... [i_q]\}$ and $([i_r], [i_s]) \in \widetilde{\mathcal{D}}$ if $r \neq s$ and there exists an arc $(k, l) \in \mathcal{D}$ for some $k \in [i_r]$ and $l \in [i_s]$.*

Let A_{rr} denote the matrix by restricting A to the nodes in $[i_r]$ $\forall r \in \underline{q}$ i.e., $[A_{rr}]_{kl} = a_{kl}$ $\forall k, l \in [i_r]$. Then $\forall r \in \underline{q}$ either A_{rr} is irreducible or is equal to ϵ. Therefore since by construction the reduced graph does not contain any circuits, the original reducible matrix A after a possible relabeling of the nodes in $G(A)$, can be written as:

$$A = \begin{pmatrix} A_{11} & A_{12} & \cdots & \cdots & A_{1q} \\ \mathcal{E} & A_{22} & \cdots & \cdots & A_{2q} \\ \mathcal{E} & \mathcal{E} & A_{33} & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \mathcal{E} & \mathcal{E} & \cdots & \mathcal{E} & A_{qq} \end{pmatrix} \tag{12}$$

with matrices A_{sr} $1 \leq s < r \leq q$, where each finite entry in A_{sr} corresponds to an arc from a node in $[i_r]$ to a node in $[i_s]$.

Definition 18. *Let $A \in \mathbb{R}_{max}^{n \times n}$ be a reducible matrix then, the block upper triangular given by (12) is said to be a normal form of matrix A.*

Spectral Theory and Linear Equations

Definition 19. *Let $A \in \mathbb{R}_{max}^{n \times n}$ be a matrix. If $\mu \in R_{max}$ is a scalar and $v \in R_{max}^n$ is a vector that contains at least one finite element such that: $A \otimes v = \mu \otimes v$ then, μ is called an eigenvalue and v an eigenvector.*

Let $\mathcal{C}(A)$ denote the set of all elementary circuits in $\mathcal{G}(A)$ and write: $\lambda = \max_{p \in \mathcal{C}(A)} \frac{|p|_w}{|p|_1}$ for the maximal average circuit weight. Notice that since $\mathcal{C}(A)$ is a finite set, the maximum is attained (which is always the case when matrix A is irreducible). In case $\mathcal{C}(A) = \emptyset$ define $\lambda = \epsilon$.

Definition 20. *A circuit $p \in G(A)$ is said to be critical if its average weight is maximal. The critical graph of A, denoted by $G^c(A) = (\mathcal{N}^c(A), \mathcal{D}^c(A))$, is the graph consisting of those nodes and arcs that belong to critical circuits in $G(A)$.*

Theorem 5. *If $A \in \mathbb{R}_{max}^{n\times n}$ is irreducible, then there exists one and only one finite eigenvalue (with possible several eigenvectors). This eigenvalue is equal to the maximal average weight of circuits in $G(A)$ $\lambda(A) = \max_{p\in\mathcal{C}(A)} \frac{|p|_w}{|p|_1}$.*

Theorem 6. *Let $A \in \mathbb{R}_{max}^{n\times n}$ and $b \in \mathbb{R}_{max}^{n}$. If the communication graph $G(A)$ has maximal average circuit weight less than or equal to e, then $x = A^* \otimes b$ solves the equation $x = (A \otimes x) \oplus b$. Moreover, if the circuit weights in $G(a)$ are negative then, the solution is unique.*

3.3 Max-Plus Recurrence Equations for Timed Event Petri Nets

Definition 21. *Let $A_m \in \mathbb{R}_{max}^{n\times n}$ for $0 \le m \le M$ and $x(m) \in \mathbb{R}_{max}^{n}$ for $-M \le m \le -1$; $M \ge 0$. Then, the recurrence equation: $x(k) = \bigoplus_{m=0}^{M} A_m \otimes x(k-m)$; $k \ge 0$ is called an Mth order recurrence equation.*

Theorem 7. *The Mth order recurrence equation, given by equation $x(k) = \bigoplus_{m=0}^{M} A_m \otimes x(k-m)$; $k \ge 0$, can be transformed into a first order recurrence equation $x(k+1) = A \otimes x(k)$; $k \ge 0$ provided that A_0 has circuit weights less than or equal to zero.*

With any timed event Petri net, matrices $A_0, A_1, ..., A_M \in \mathbb{N}^n \times \mathbb{N}^n$ can be defined by setting $[A_m]_{jl} = a_{jl}$, where a_{jl} is the largest of the holding times with respect to all places between transitions t_l and t_j with m tokens, for $m = 0, 1, ..., M$, with M equal to the maximum number of tokens with respect to all places. Let $x_i(k)$ denote the kth time that transition t_i fires, then the vector $x(k) = (x_1(k), x_2(k), ...x_m(k))^T$, called the state of the system, satisfies the Mth order recurrence equation: $x(k) = \bigoplus_{m=0}^{M} A_m \otimes x(k-m)$; $k \ge 0$ Now, assuming that all the hypothesis of theorem (7) are satisfied, and setting $\hat{x}(k) = (x^T(k), x^T(k-1), ..., x^T(k-M+1))^T$, equation $x(k) = \bigoplus_{m=0}^{M} A_m \otimes x(k-m)$; $k \ge 0$ can be expressed as: $\hat{x}(k+1) = \hat{A} \otimes \hat{x}(k)$; $k \ge 0$, which is known as the standard autonomous equation.

4 The Solution to the Stability Problem for Discrete Event Dynamical Systems Modeled with Timed Petri Nets

Definition 22. *A TPN is said to be stable if all the transitions fire with the same proportion i.e., if there exists $q \in \mathbb{N}$ such that*

$$\lim_{k\to\infty} \frac{x_i(k)}{k} = q, \forall i = 1, ..., n \tag{13}$$

This means that in order to obtain a stable TPN all the transitions have to be fired q times. It will be desirable to be more precise and know exactly how many times. The answer to this question is given next.

Lemma 3. *Consider the recurrence relation* $x(k+1) = A \otimes x(k), k \geq 0$, $x(0) = x_0 \in \mathbb{R}^n$ *arbitrary.* A *an irreducible matrix and* $\lambda \in \mathbb{R}$ *its eigenvalue then,*

$$\lim_{k\to\infty} \frac{x_i(k)}{k} = \lambda, \forall i = 1, ..., n \tag{14}$$

Proof. Let v be an eigenvector of A such that $x_0 = v$ then,

$$x(k) = \lambda^{\otimes k} \otimes v \Rightarrow x(k) = k\lambda + v \Rightarrow \frac{x(k)}{k} = \lambda + \frac{v}{k} \Rightarrow \lim_{k\to\infty} \frac{x_i(k)}{k} = \lambda$$

Now starting with an unstable TPN, collecting the results given by: proposition (2), what has just been discussed about recurrence equations for TPN at the end of subsection (3.3) and the previous lemma (3) plus theorem (5), the solution to the problem is obtained.

5 The Biological Interaction Stability Problem

Consider a biological interaction system whose TPN model is depicted in Fig 1.

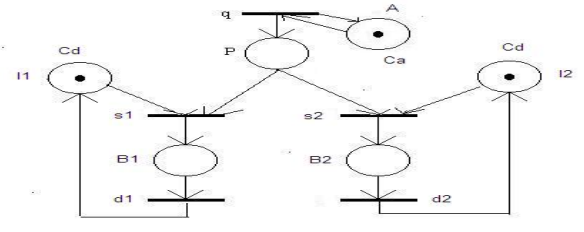


Fig. 1. Timed Petri net model

Where the events (transitions) that drive the system are: q: resource to be consumed, s1, s2: consuming starts, d1,d2: the resource has been consumed. The places that represent the states are: A: the resource is active, P: the resource is available to be consumed, B1, B2: the resource is being consumed, I1, I2: the organisms are idle. The holding times associated to the places A and I1, I2 are Ca and Cd respectively, (with $Ca > Cd$) i.e., limited resource supply. The incidence matrix that represents the PN model is

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$

Therefore since there does not exists a Φ strictly positive m vector such that $A\Phi \leq 0$ the sufficient condition for stability is not satisfied, (moreover, the PN (TPN) is unbounded since by the repeated firing of q, the marking in P grows indefinitely). However, by taking $u = [k, k/2, k/2, k/2, k/2]; k > 0$ (but unknown) we get that $A^T u \leq 0$. Therefore, the PN is stabilizable which implies that the TPN is stable. Now, let us proceed to determine the exact value of k. From the TPN model we obtain that:

$$A_0 = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 0 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 0 & \varepsilon & \varepsilon \end{pmatrix} \text{ and} A_1 = \begin{pmatrix} Ca & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & Cd & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & Cd \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{pmatrix} \text{ which implies } A_0^* = \begin{pmatrix} 0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 0 & 0 & \varepsilon & \varepsilon & \varepsilon \\ 0 & \varepsilon & 0 & \varepsilon & \varepsilon \\ 0 & 0 & \varepsilon & 0 & \varepsilon \\ 0 & \varepsilon & 0 & \varepsilon & 0 \end{pmatrix},$$

leading to:

$$\hat{A} = A_0^* \otimes A_1 = \begin{pmatrix} Ca & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ Ca & \varepsilon & \varepsilon & Cd & \varepsilon \\ Ca & \varepsilon & \varepsilon & \varepsilon & Cd \\ Ca & 0 & \varepsilon & Cd & \varepsilon \\ Ca & \varepsilon & \varepsilon & \varepsilon & Cd \end{pmatrix}$$

Therefore, $\lambda(A) = \max\limits_{p \in \mathcal{C}(A)} \frac{|p|_w}{|p|_1} = \max\{Ca, Cd\} = Ca$. This means that in order for the TPN to be stable and work properly the speed at which the two organisms consume has to be equal to Ca which is attained by taking $k = Ca$, i.e., the resource has to be equally shared between the two organisms.

6 Conclusions

The main contribution of the paper consists in combining Lyapunov theory with max-plus algebra to study the biological interaction stability problem treated as a discrete event dynamical system modeled with timed Petri nets.

References

1. Haberman, R.: Mathematical Models in mechanical vibrations, population dynamics, and traffic flow. Prentice Hall (1977)
2. Weibul, J.: Evolutionary Game Theory. The MIT Press (1977)
3. Baccell, G., Cohen, G., Olsder, G.J., Quadrat, J.P.: Synchronization and Linearity, Web-edition (2001)
4. Heidergott, B., Olsder, G.J., van der Woude, J.: Max Plus at Work. Princeton University Press (2006)
5. Retchkiman, Z.: Stability theory for a class of dynamical systems modeled with Petri nets. International Journal of Hybrid Systems 4(1) (2005)
6. Retchkiman, Z.: Modeling and Analysis of the Metro-bus Public Transport System in Mexico City using Timed Event Petri Nets and Max-Plus Algebra. Nonlinear Analysis: Hybrid Systems (2008)

Population-Based Incremental with Adaptive Learning Rate Strategy

Komla A. Folly

Department of Electrical Engineering, University of Cape Town,
Private Bag. Rondebosch 7701 Cape Town, South Africa
Komla.Folly@uct.ac.za

Abstract. Population-Based Incremental Learning (PBIL) is a relatively new class of Evolutionary Algorithms (EA) that has been recently applied to a range of optimization problems in engineering with promising results. PBIL combines aspects of Genetic Algorithm with competitive learning. The learning rate in the standard PBIL is generally fixed which makes it difficult for the algorithm to explore the search space effectively. In this paper, a PBIL with Adapting learning rate is proposed. The Adaptive PBIL (APBIL) is able to thoroughly explore the search space at the start of the run and maintain the diversity longer than the standard PBIL. To show its effectiveness, the proposed algorithm is applied to the problem of optimizing the parameters of a power system controller. Simulation results show that APBIL based controller performs better than the standard PBIL based controller.

Keywords: Adaptive learning rate, low frequency oscillations, population-based incremental learning, power system stabilizer.

1 Introduction

In the last three decades or so, there has been a growing interest in applying Evolutionary Algorithm (EA) to engineering optimization problems. The most widely used EA is Genetic Algorithms (GAs) [1]. Although GAs provide robust and powerful adaptive search mechanism, they have several drawbacks such as "genetic drift" which prevents GAs from maintaining diversity in the population. Other drawbacks include the difficulty to optimally select the genetic operators (e.g., population size, crossover and mutation rates), and the slow convergence of the algorithm when solving complex problems [2]-[3].

In the last few years, Particle Swarm Optimization (PSO) which belongs to the family of swarm intelligence has also been proposed as an alternative to GAs [4]-[6]. Recently, a novel type of Evolutionary Algorithm called Population-Based Incremental Learning (PBIL) [7]-[8] has received increasing attention [9]-[12]. PBIL is simpler and more effective than GAs. In PBIL, the crossover operator of GAs is abstracted away and the role of population is redefined [7]. PBIL works with a probability vector (PV) which controls the random bit strings generated by PBIL and is used to create other individuals through learning. Learning in PBIL consists of

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 11–20, 2012.

using the current probability vector (PV) to create *N* individuals. The best individual is used to update the probability vector, increasing the probability of producing solutions similar to the current best individuals [8]-[9]. It has been shown that PBIL outperforms standard GAs approaches on a variety of optimization problems including commonly used benchmark problems [7], [8]**.** In [10], PBIL based power system stabilizers (PSSs) were compared with GA based PSSs and were found to give better results than GA based PSSs**.** In [11], it was shown that PBIL-PSS performed as effectively as BGA-PSS. In [12], PBIL based PSSs, were compared with several other population-based algorithms such as Differential Evolution based particle Swarm Optimization (DEPSO), Modified Clonal Selection Algorithm (MCSA), Small Population based Particle Swarm Optimization (SPPSO) and were found to give adequate performance. However, there are still some issues related to PBIL. The learning rate in the standard PBIL is generally fixed to a certain value. Therefore, it becomes difficult for the algorithm to explore and/or exploit the search space in an effective manner. It has been reported in [14] that PBIL suffers from diversity loss making the algorithm to converge to local optima. To cope with this problem, an adaptive learning rate first proposed in [15] is used in this paper to design power system controller for a simple power system. The Adaptive PBIL (APBIL) is able to thoroughly explore the search space at the start of the run and maintain the diversity of solutions longer during the search than the standard PBIL. In formualting the Adaptive PBIL algorithm, we have tried to use simple equations so as to keep the simplicity of the algorithm. To show the effectiveness of the Adaptive PBIL, the algorithm was applied to the problem of optimizing the parameters of a power system stabilizer (PSS). Simulation results show that the PSS designed based on the Adaptive PBIL performs better than those based on standard PBIL and the Conventional PSS (CPSS).

2 Overview of the Standard PBIL

PBIL is a technique that combines aspects of Genetic Algorithms and simple competitive learning derived from Artificial Neural Networks [7], [8]. PBIL belongs to the family of Estimation of Distribution Algorithms (EDAs), which use the probability (or prototype) vector to generate sample solutions. There is no crossover operator in PBIL; instead the probability vector is updated using solution with the highest fitness values [9]. Initially, the values of the probability vector are set to 0.5 to ensure that the probability of generating 0 or 1 is equal. As the search progresses, these values are moved away from 0.5, towards either 0.0 or 1.0.

Like in GA, mutation is also used in PBIL to maintain diversity. In this paper, the mutation is performed on the probability vector; that is, a forgetting factor is used to relax the probability vector toward a neutral value of 0.5 [9], [10].

A summary of the PBIL used in this paper is given below [8]-[12]:

Step 1. Initialize element of the probability vector (PV) to 0.5 to ensure uniformly-random bit strings.

Step2. Generate a population of uniformly-random bit strings and comparing it element-by-element with the PV. Wherever an element of the PV is greater than the corresponding random element, a "1' is generated, otherwise a '0' is generated.

Step 3. Interpret each bit string as a solution to the problem and evaluate its merit in order to identify the "Best".

Step 4. Adjust *PV* by slightly increasing *PV* (*i*) to favor the generation of bit strings which resemble "Best", if Best (*i*) = 1 and decrease PV(*i*) if Best(*i*) = 0.

Step 5. Apply the mutation and generate a new population reflecting the modified distribution. Stop if satisfactory solution is found. Otherwise, go to step 2.

It should be mentioned that the probability vector guides the search, which produces the next sample point from which learning takes place. The learning rate determines the speed at which the probability vector is shifted to resemble the best (fittest) solution vector. If the learning rate is fixed during the run, it cannot provide the flexibility needed to achieve a trade-off between exploration and exploitation. The effect of the learning rate on the performance of the PBIL is still an active research topic [7]-[13]. In the next section we propose an adaptive learning rate to cope with this shortcoming.

3 Overview of the PBIL with Adaptive Learning Rate

As discussed previously, the learning rate in the standard PBIL is usually fixed to a specific value. This means that the user has to spend a lot of time and try several values of the learning rate before deciding on the "best" value to use. In addition, a fixed learning rate may not be adequate if the search space environment is dynamic and changes often as is the case in power systems.

If the learning rate value is too high, this may lead to premature convergence and the algorithm could converge to local optima. If on the other hand, the learning rate is too small, the algorithm may be slow to converge and will require more time to find the optimal solution. This is computationally costly. It is therefore critical that the learning rate be chosen such that a trade-off between exploration and exploitation is achieved.

To develop the adaptive learning algorithm, we assume that at the start of the run, diversity will be needed for the algorithm to be able to explore thoroughly the search space. Therefore, at the start, a very small value of learning rate (LR≈ 0) is selected. Therefore, the emphasis at the beginning of the run is on the exploration of the search space as opposed to exploitation. As the run progresses and good individuals start to emerge, the emphasis is shifted gradually from exploration to exploitation of the search space. In the algorithm discussed in this paper, we increase the learning rate slowly (and linearly) according to the change in generation as given in the following equation:

$$LR(i) = LR\frac{G(i)}{G\,max}\ . \tag{1}$$

where

LR(i) is the learning rate at the i^{th} generation

LR is the final learning rate

G(i) is the *ith* generation

Gmax: is the maximum generation allowed

The pseudocode of the APBIL is similar to that of standard PBIL except that the learning rate is not anymore fixed, but varies according to the generation.

4 Problem Description and System Model

The APBIL is now applied to a problem of controller design in power systems. The controller to be designed is also known as Power System Stabilizer (PSS) and is needed to damp low frequency oscillations ranging from 0.1 Hz to 3 Hz which occur in overly stressed power systems or when power is transmitted over weak transmission lines [16]-[17]. These oscillations are highly undesirable because they can lead to fatigue of machine shafts and limit the ability of the system to transfer the maximum power. It is therefore important that low-frequency oscillations are damped quickly if the security of the system is to be maintained.

In this paper, the power system model used is a single machine connected to an infinite bus (SMIB) system [11], [16] as shown in Fig. 1. The generator is modelled using a six order differential equations. To improve the transient stability, an Automatic Voltage Regulator (AVR) is used. However, high gain fast acting AVR used in this paper also has a negative effect on the damping. This AVR was modelled by a first order differential equation. On the other hand, the speed governor was neglected. The non-linear differential equations of the system are linearized around the nominal operating condition as given below:

$$\begin{aligned} \dot{x} &= A_o x + B_o u \\ y &= C_o x + D_o u \end{aligned} \quad (2)$$

where

A is the system state matrix; *B* is the system input matrix; *C* is the system output matrix; *D* is the feed forward matrix; *x* is the vector of the system states; *u* is the vector of the system inputs; and *y* is the vector of the system outputs (i.e., speed variations).

A, *B*, *C* and *D* are constant matrices of appropriate dimensions. The sizes and contents of these matrices are given in the Appendix A.

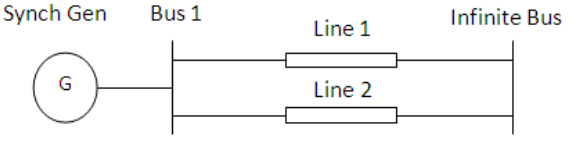


Fig. 1. Power system model

5 Problem Formulation

5.1 Selected Operating Conditions

Table 1 shows the eigenvalues and damping ratios in brackets of the four operating conditions that are considered in this paper. In Table 1, P_e represents the electrical output, X_e is the system's total transmission reactance and ζ the damping ratio. It can be seen that the nominal operating condition (case 1) is stable. However, all the other three cases are unstable as shown by the negative damping ratios in Table 1. Without PSS, the system is unstable and will not be able to operate adequately under these operating conditions.

5.2 PSS Structure and Objective Functions

PBIL with fixed and adaptive learning rate is applied to optimize the parameters of a fixed structure ($\Delta\omega$ input) PSS of the form given in Eq. (3)

$$K(s) = K_p \left(\frac{T_w s}{1 + T_w s} \right) \left(\frac{1 + T_1 s}{1 + T_2 s} \right) \left(\frac{1 + T_3 s}{1 + T_4 s} \right). \tag{3}$$

where, K_p is the gain, T_1-T_4 represent suitable time constants. T_w is the washout time constant needed to prevent steady-state offset of the voltage. The value of T_w is not critical for the PSS and has been set to 5sec.

Therefore, five parameters are required for the optimization.

Since most of the operating conditions considered in this paper are unstable and dominate the time domain responses of the system, it is expected that by maximizing the minimum damping ratio, one could simultaneously stabilize a set of system models over a wide range of operating conditions [10]-[12]. The following objective function was used to design the PSSs.

$$J = \max\left(\min\left(\zeta_{i,j} \right) \right). \tag{4}$$

where $i = 1, 2 \ldots n$, and $j = 1, 2, \ldots m$

and $\zeta_{i,j} = \dfrac{-\sigma_{i,j}}{\sqrt{\sigma_{i,j}^{\ 2} + \omega_{i,j}^{\ 2}}}$ is the damping ratio of the i-th eigenvalue in the j-th operating condition. σ_{ij} is the real part of the eigenvalue and the ω_{ij} is the frequency. The total number of eigenvalues is n and m denotes the number of operating conditions.

Table 1. Selected open-loop operating conditions including eigenvalues and damping ratios

Case	P_e [p.u]	X_e [pu]	Eigenvalue (ζ)
1	0.40	0.25	-0.8432 ± 8.5529i (0.0981)
2	0.80	0.80	0.3144 ± 7.9100i (-0.0397)
3	0.80	1.00	0.3601 ± 7.4183i (-0.0485)
4	1.04	0.80	0.7880 ± 7.9306i (-0.0989)

5.3 Application of Standard PBIL to Controller Design

The configuration of the standard PBIL is as follows:

Length of chromosome: 15 bits
Trial solutions (population): 50
Generations: 600
Learning rate (LR): 0.1
Mutation (Forgetting factor-FF): 0.005

5.4 Application of APBIL to Controller Design

The configuration of the APBIL is as follows:

Length of chromosome: 15 bits
Trial solutions (population): 50
Generations: 800
Initial Learning rate (LR_0) = 0.00025
Final Learning rate (LR_{max}): 0.2
Mutation (Forgetting factor-FF): 0.005

5.5 Design of the Conventional PSS

The conventional PSS and has been designed based on the nominal operating condition using phase compensation technique [16]. Therefore, it is anticipated that the controller will not perform optimally at off nominal operating conditions.

6 Simulation Results

6.1 Eigenvalue Analysis

Table 2 shows the eigenvalues and the damping ratios (in brackets) of the closed-loop system equipped with the CPSS, the PBIL-PSS and the Adaptive PBIL-PSSs.

It can be seen from Table 2 that all the PSSs have improved the damping ratio of the system under all the cases. In particular cases 2-4 which were unstable without the controllers have been stabilized with the introduction of the PSSs. The APBIL-PSS

provides the best damping ratio for all the operating conditions considered, except for case 1 where PBIL-PSS's damping ratio is about 0.8% higher than that of APBIL-PSS. This difference is practically insignificant.

6.2 Time Domain Simulations

Time domain simulations for small-signal stability study were also performed by applying a 10% step disturbance in Vref. The rotor speed responses of the system under the various cases are shown in Figs. 2-5. It can be seen from these figures that overall, the APBIL-PSS gives the best performance for all the operating conditions. It has the lowest overshoots and undershoots than the standard PBIL-PSS and the CPSS. The standard PBIL-PSS in turn performs better than the CPSS.

In terms of settling time, both the APBIL-PSS and PBIL-PSS have almost the same settling time. On the other hand, the responses with the CPSS have the largest overshoots and undershoots and the longest setting time. The performance of the CPSS seems to deteriorate as the system moves further from the nominal operating condition. This is expected as it has been designed based on a single operating condition using the classical control approach.

Table 2. Closed-system eigenvalues and damping ratio

Case	CPSS	PBIL	APBIL
1	-2.78±5.52i (0.45)	-4.31 ±4.41i (0.70)	-2.94 ±3.05i (0.69)
2	-1.54 ±5.79i (0.26)	-3.02 ±4.96i (0.52)	-2.34 ±3.67i (0.54)
3	-1.22 ±5.67i (0.21)	-2.57 ±5.03i (0.46)	-2.15 ±3.85i (0.49)
4	-1.27 ±5.92i (0.21)	-2.81 ±5.09i (0.48)	-2.19 ±3.80i (0.50)

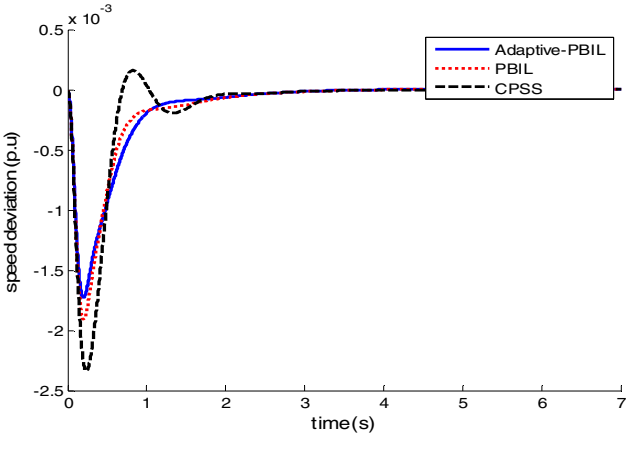


Fig. 2. Rotor speed responses for a 10% step disturbance (case 1)

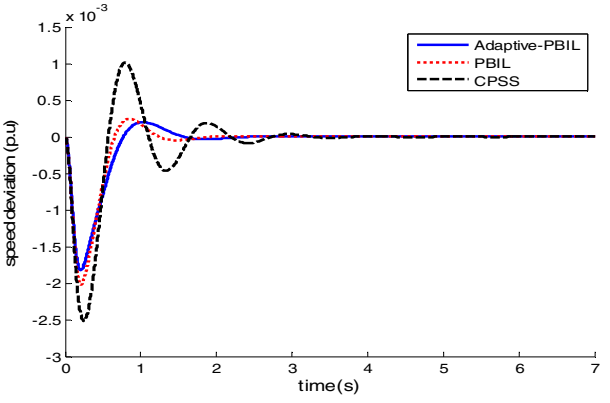


Fig. 3. Rotor speed responses for a 10% step disturbance (case 2)

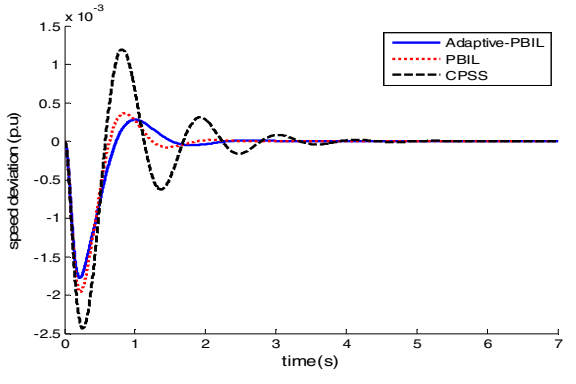


Fig. 4. Rotor speed responses for a 10% step disturbance (case 3)

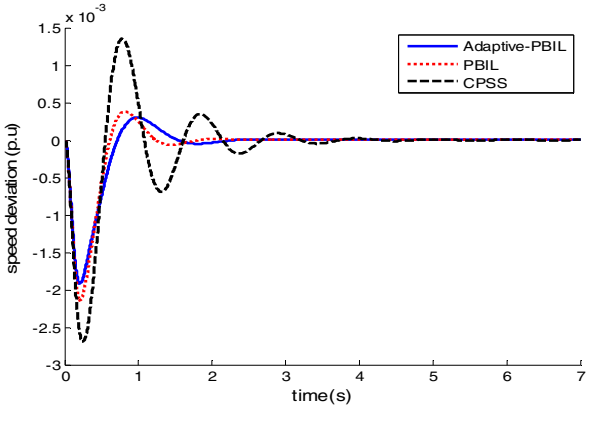


Fig. 5. Rotor speed responses for a 10% step disturbance (case 4)

7 Conclusions

By using adaptive learning rate we are able to achieve a better trade-off between exploration and exploitation as can be seen by the performance of the Adaptive-PBIL. Eigenvalue analysis shows that the Adaptive-PBIL based PSS provides a better damping to the system than the standard PBIL based PSS and the CPSS. These results have been confirmed by time domain simulations based on a small disturbance. Due to the low CPU requirements and the robustness of the representation of PBIL, this method is very attractive for online implementation. It is expected that the Adaptive-PBIL will be improved in the future by introducing some feedback mechanism.

Acknowledgments. The financial support of THRIP-uid 71967 and TESP is acknowledged.

References

1. Goldberg, D.E.: Genetic Algorithms in Search, Optimization & Machine Learning. Addison-Wesley (1989)
2. Davis, L.: Handbook of Genetic Algorithms. International Thomson Computer Press (1996)
3. Yao, J., Kharma, N., Grogono, P.: Bi-objective Multipopulation Genetic Algorithm for Multimodal Function Optimization. IEEE Trans. Evol. Comput. 14(1), 80–102 (2010)
4. Kennedy, J.F., Kennedy, J., Eberhart, R.C., Shi, Y.: Swarm Intelligence. Morgan Kaufmann (2001)
5. Abido, A.A.: Particle swarm Optimization for Multimachine Power System Stabilizer Design. IEEE Trans. Power Syst. 3(3), 1346–1351 (2001)
6. Venayagamoorthy, G.K.: Improving the Performance of Particle Swarm Optimization using Adaptive Critics Designs. In: IEEE Proceedings on Swarm Intelligence Symposium, pp. 393–396 (2005)
7. Baluja, S.: Population-Based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning. Technical Report, CMU-CS-94-163, Carnegie Mellon University (1994)
8. Baluja, S., Caruana, R.: Removing the Genetics from the Standard Genetic Algorithm. Technical Report CMU-CS-95-141, Carnegie Mellon University (1995)
9. Greene, J.R.: Population-Based Incremental Learning as a Simple, Versatile Tool for Engineering Optimization. In: Proceedings of the First International Conf. on EC and Applications, pp. 258–269 (1996)
10. Folly, K.A.: Design of Power System Stabilizer: A Comparison Between Genetic Algorithms (GAs) and Population-Based Incremental Learning (PBIL). In: Proc. of the IEEE PES 2006 General Meeting, Montreal, Canada (2006)
11. Sheetekela, S., Folly, K.A.: Power System Controller Design: A Comparison Between Breeder Genetic Algorithm (BGA) and Population-Based Incremental Learning (PBIL). In: Proc. of the Int. Joint Conference on Neural Networks IJCNN (2010)
12. Mitra, P., Yan, C., Grant, L., Venayagamoorthy, G.K., Folly, K. : Comparative Study of Population-Based Techniques for Power System Stabilizer Design. In: 15th Int. Conf. on Intelligent System Applications to Power Systems (ISAP 2009), Curitiba, Brazil (2009)

13. Gosling, T., Jin, N., Tsang, E.: Population-Based Incremental Learning Versus Genetic Algorithms: Iterated Prisoners Dilemma. Technical Report CSM-40, University of Essex, England (2004)
14. Rastegar, R., Hariri, A., Mazoochi, M.: The Population-Based Incremental Learning Algorithm Converges to Local Optima. Neurocomputing 69, 1772–1775 (2006)
15. Folly, K.A., Venayagamoorthy, G.K.: Effect of Learning Rate on the Performance of the Population-Based Incremental Learning Algorithm. In: Proc. of the International Joint Conf. on Neural Network (IJCNN), Atlanta Georgia, USA (2009)
16. Kundur, P.: Power System Stability and Control. McGraw – Hill, Inc. (1994)
17. Gibbard, M.J.: Application of Power System Stabilizer for Enhancement of Overall System Stability. IEEE Trans. on Power Systems 4(2), 614–626 (1989)

A SI-Based Algorithm for Structural Damage Detection

Ling Yu[1,2], Peng Xu[1], and Xi Chen[1]

[1] Department of Mechanics and Civil Engineering, Jinan University, Guangzhou 510632, China
[2] MOE Key Lab of Disaster Forecast and Control in Engineering, Jinan University, Guangzhou 510632, China
lyu1997@163.com

Abstract. As a challenging task in the structural health monitoring (SHM) field, structural damage detection, one of most important issues of the SHM system, is mathematically converted into a constrained optimization problem, which is then hopefully solved by a swarm intelligence (SI) based algorithm proposed in this paper. The performance of the proposed algorithm is experimentally evaluated by the measured data of four damage patterns of a building model of 3-storey steel frame structure made in laboratory. Some illustrated results show that the proposed method is very suitable for the structural multi-damage identification, which also show that the SI-based algorithm for structural damage detection can provide an effective and robust tool in the SHM field.

Keywords: Structural health monitoring, damage detection, Swarm intelligence, optimization problem, ant colony optimization, particle swarm optimization.

1 Introduction

In the last few decades, the structural damage detection (SDD), one of the most critical components of the structural health monitoring (SHM) system, has been most commonly investigated in many ways and algorithms [1]. Traditionally, structural system identification techniques have been commonly used by relating the damage to the change in the vibration characteristics of the structure [2-3]. However, there are a number of challenges to be overcome before routine applications of SHM. In essence, the most effective strategy should treat the SDD as a constrained optimization problem [4]. However, there is no universal agreement as to the optimum method for using measured vibration data for damage detection, location or quantification. One of difficulties is that the traditional gradient-based methods are easily led to local rather than global minimum, i.e. so called the premature convergence. Therefore, it is necessary to explore some new approaches for the SDD problem.

Swarm intelligence (SI) provides a new framework for the design and implementation of systems made of many agents that are capable of cooperation for the solution of complex problems. The potential advantages of the swarm intelligence approach are manifold: collective robustness, individual simplicity and scalability [5]. The majority of

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 21–28, 2012.

research to date has focused on demonstrating the cooperative problem solving capabilities of swarm intelligent systems. Very encouraging results have been obtained, especially in optimization applications, such as ant colony optimization (ACO) and particle swarm optimization (PSO), where swarms of software agents cooperate to search for good solutions to difficult optimization problems.

In order to achieve a promising solution to the constrained optimization problem on SDD, an improved PSO is proposed to tackle the premature convergence at the last phase of iterations for a conventional PSO. Extending ACO to the continuous optimization problem on SDD is also conducted in this paper. Feasibility of the proposed SI-based algorithm is compared and assessed by using some measured data of 3-storey steel frame structure made in laboratory. Experimental verifications show satisfied results as well as good performance, effectiveness and robustness of the proposed method.

2 Theoretical Background

2.1 Improved Particle Swarm Optimization

Particle Swarm Optimization (PSO). As PSO is implemented, each particle is determined by its position and velocity. It moves towards its best previous position, *pbest*, and towards the best position of the whole swarm, *gbest*, respectively.

Supposing a swarm of p particles, the search space is d-dimensional, and then the position of i-th particle can be represented by a d-dimensional vector, $x_i = (x_{i1}, x_{i2} \cdots x_{id})$, $i = 1,2,\cdots, p$. The velocity of the particle can be represented by another vector, $v_i = (v_{i1}, v_{i2} \cdots v_{id})$. The best previous visited position of the particle is denoted as $pbest_i = (pbest_{i1}, pbest_{i2}, \cdots, pbest_{id})$. The velocity and the position of the particle are updated according to the following two equations:

$$v_{ij}^{k+1} = w \times v_{ij}^{k} + c_1 \times r_1 \times (pbest_{ij} - x_{ij}^{k}) + c_2 \times r_2 \times (gbest_j - x_{ij}^{k}) \tag{1}$$

$$x_{ij}^{k+1} = x_{ij}^{k} + v_{ij}^{k+1} \tag{2}$$

Where $j = 1,2,\cdots d$. Both r_1 and r_2 are random numbers, which uniformly distributed between [0, 1]. Both c_1 and c_2 are the positive constant, $c_1 = c_2 = 2$ [5]. w is inertial weight calculated in the following form for a quick convergence and good result,

$$w = w_{\max} - iter \times \frac{w_{\max} - w_{\min}}{iter_{\max}} \tag{3}$$

Where, $w_{\max}$ and $w_{\min}$ are the inertial and final weights respectively. $iter$ and $iter_{\max}$ are the current and maximum iteration numbers respectively. A large inertial weight facilitates global exploration while a small one tends to facilitates local

exploration. Suitable selection of the inertial weight can balance the global and local search. Here, PSO refers to the PSO with inertial weight w and $w_{\max}$=0.9, $w_{\min}$=0.4.

Improved PSO (IPSO). At last phase of PSO calculation, as all particles move towards *pbest* and *gbest*, positions of all particles would be near to the *pbest* and *gbest*, the last two items of the Eq. (1) are also close to zero. Therefore, a so-called premature convergence problem will easily be caused.

An improved PSO (IPSO) algorithm is proposed for easily exploring a global solution at the last stage of the PSO. When the iteration numbers of the algorithm reaches to some target value, e.g., $iter = 0.3iter_{\max}$ the particle position is changed to the following one,

$$x_{ij}^{k+1} = x_{ij}^{k} + \alpha v_{ij}^{k+1} - (C3 + \alpha) \cdot (f_i - f_{iter\min}) \tag{4}$$

$$\alpha = 1/(1 + \exp(-(f_i - f_{iter\min}))) \tag{5}$$

Where, α is a sigmoid function, a typical neuronal non-linear transfer function that helps make the outputs reachable. f_i is the fitness value of the *iter* number for the *i-th* set of particle, $f_{iter\min}$ is the minimum fitness value. From Eq. (5), the adjustment coefficient α ranges between [0, 1]. When $C3 = -\alpha$, the third term in Eq. (4) is equal to zero. The second term in Equation is less than v_{ij}^{k+1}, if α is not equal to one. Therefore, the new position of particle, x_{ij}^{k+1}, is forced to decrease. Meanwhile, the x is limited to a range between $[C4 \cdot x_{\min}, x_{\max}]$, where C4 is less than one. C4=0.9, C3=1 are accepted in this paper [6].

2.2 Continuous Ant Colony Optimization

Ant Colony Optimization (ACO). In general, the ACO approach attempts to solve an optimization problem by iterating the following two steps:

i) Candidate solutions are constructed in a probabilistic way by using a probability distribution over the search space.
ii) The candidate solutions are used to modify the probability distribution in a way that is deemed to bias future sampling toward high quality solutions.

The central component of ACO algorithms is the pheromone model which is a set of so-called pheromone trail parameters. The numerical values of these pheromone trail parameters reflect the search experience of the algorithm. They are used to bias the solution construction over time to regions of the search space containing high quality solutions. They are required to update in order to increase the pheromone values associated with good or promising solutions and to decrease those that are associated with bad ones. Usually, this is achieved by increasing the pheromone levels associated with chosen good solution and by decreasing all the pheromone values through pheromone evaporation. In general, good solutions found earlier by the ants are used to update the pheromone in order to increase the probability of the search by subsequent ants in the promising regions of the search space.

Continuous Ant Colony Optimization (CACO). In ACO applied to combinatorial optimization problems, the set of available solution components is defined by the problem formulation. For a continuous optimization problem, the fundamental idea underlying ACO is the shift from using a discrete probability distribution to using a continuous one, that is, a probability density function (PDF). For CACO, the general approach to sampling PDF P(x) is to use the inverse of its cumulative distribution function (CDF) D(x). However, it is important to note that for an arbitrarily chosen PDF P(x), it is not always straightforward to find the inverse of D(x). One of the most popular functions that is used as a PDF is the Gaussian function. By defining a Gaussian kernel $G^i(x)$ as a weighted sum of several one-dimensional Gaussian functions $g_l^i(x)$ as below:

$$\boldsymbol{G}^i(\boldsymbol{x}) = \sum_{l=1}^{k} \omega_l \boldsymbol{g}_l^i(\boldsymbol{x}) = \sum_{l=1}^{k} \omega_l \frac{1}{\sigma_l^i \sqrt{2\pi}} \boldsymbol{e}^{\frac{(x-\mu_l^i)^2}{2(\sigma_l^i)^2}} \tag{6}$$

Where, i=1,2,…,n is the number of dimensions of problem, which identifies a single such PDF. The $G^i(x)$ is parameterized with three vectors of parameters: ω is the vector of weights associated with the individual Gaussian functions, μ^i is the vector of means, and σ^i is the vector of standard deviations. The cardinality of all these vectors is equal to the number of Gaussian functions constituting the Gaussian kernel. Their definition follows the metaheuristic framework in reference [7].

The whole process is repeated for each dimension and each time the average distance σ_l^i is calculated only with the use of the single dimension. This ensures that the algorithm is able to adapt to linear transformations of the considered problem.

3 SI-Based Optimization Problem on Structural Damage Detection

The motion equation of a system with n degrees of freedom (DOFs) can be expressed as follows

$$\boldsymbol{K}\boldsymbol{\varphi}_j = \lambda_j \boldsymbol{M}_0 \boldsymbol{\varphi}_j \tag{7}$$

$$\boldsymbol{K} = \boldsymbol{K}(\boldsymbol{\alpha}) = \sum_{i=1}^{N_e} \alpha_i \boldsymbol{K}_i \tag{8}$$

Where $\boldsymbol{K}$ is the global stiffness matrix of a damaged structure. Since the change in mass matrix before and after damage is very small and always assumed to be unchanged in most cases, the global mass matrix of health structure $\boldsymbol{M}_0$ is used here instead of its corresponding damaged one $\boldsymbol{M}$. λ_j and φ_j are the j-th eigenvalue and eigenvector respectively, $j=1,2,...,N_m$ and N_m is the number of measured mode shapes. $\lambda_j=(2\pi f_j)^2$, and f_j is the j-th natural frequency. $\boldsymbol{K}_i$ is the i-th element stiffness matrix, and $\boldsymbol{\alpha}=[\alpha_1,\alpha_2,...,\alpha_{N_e}]^T$ is the stiffness reduction extent (SRE), which is specified to the elemental bending stiffness (EI) in this paper. N_e is the element number of the finite element model (FEM) of a structure.

Objective Function. The objective function is defined as a problem minimizing differences between the experimental and analytical modal results. The minimization of objective function is expressed as a bound-constrained nonlinear least-squares (BCNLS) problem as follows,

$$\min_{\boldsymbol{\alpha}\in R^{N_e}} f(\boldsymbol{\alpha}) = \frac{1}{2}\left\|\boldsymbol{r}(\boldsymbol{\alpha})\right\|^2 = \frac{1}{2}\sum_{i=1}^{N} r_i^2(\boldsymbol{\alpha}) \tag{9}$$

$$\boldsymbol{l} \le \boldsymbol{\alpha} \le \boldsymbol{u}, \boldsymbol{r} : R^{N_e} \to R^N$$

$$\boldsymbol{r}(\boldsymbol{\alpha})_{N\times 1} = \left[\boldsymbol{r}_f(\boldsymbol{\alpha})^T_{N_m\times 1},\ \boldsymbol{r}_\varphi(\boldsymbol{\alpha})^T_{N_m\times 1}\right]^T \tag{10}$$

Where $\boldsymbol{r}(\boldsymbol{\alpha}) = (r_1(\boldsymbol{\alpha}), r_2(\boldsymbol{\alpha}), \ldots, r_N(\boldsymbol{\alpha}))^T$, is a *N*-dimensional vector-valued function, $N = 2N_m \ge N_e$. $\boldsymbol{l}$, $\boldsymbol{u}$ are vectors of lower and upper bound, respectively. In this paper, all components of $\boldsymbol{l}$ and $\boldsymbol{u}$ are set to be 0 and 1 respectively for the purpose of damage detection. $\boldsymbol{r}_f(\boldsymbol{\alpha})$ is a N_m-dimensional vector denoting the difference of frequency ratio before and after damage. $\boldsymbol{r}_\varphi(\boldsymbol{\alpha})$ is also a vector containing the modal assurance criterion (MAC) of each tested mode shape with N_m dimension, where mode shapes with only measured DOFs are used and mode shape expansion is not required here. The formulas of $\boldsymbol{r}_f(\boldsymbol{\alpha})$ and $\boldsymbol{r}_\varphi(\boldsymbol{\alpha})$ are given as, respectively,

$$r_f^i(\boldsymbol{\alpha}) = \left|1 - \frac{f_a^i(\boldsymbol{\alpha})}{f_t^i}\right|, \qquad r_\varphi^i(\boldsymbol{\alpha}) = \frac{(\boldsymbol{\varphi}_t^{iT}\boldsymbol{\varphi}_a^i(\boldsymbol{\alpha}))^2}{(\boldsymbol{\varphi}_t^{iT}\boldsymbol{\varphi}_t^i)(\boldsymbol{\varphi}_a^i(\boldsymbol{\alpha})^T\boldsymbol{\varphi}_a^i(\boldsymbol{\alpha}))} \tag{11}$$

Here f^i and $\boldsymbol{\varphi}^i$ are the *i*-th natural frequency and mode shape, $i = 1, 2, \ldots, N_m$, subscripts *t* and *a* denote tested and analytical data respectively.

4 Experimental Verification

In order to assess the performance of the proposed SI-based algorithm for SDD optimal problem, some measured data of a simple 3-storey building steel frame are adopted [8]. The traditional PSO and the proposed IPSO and CACO here respectively are used to solve the optimal problem on SDD for evaluation on their validity.

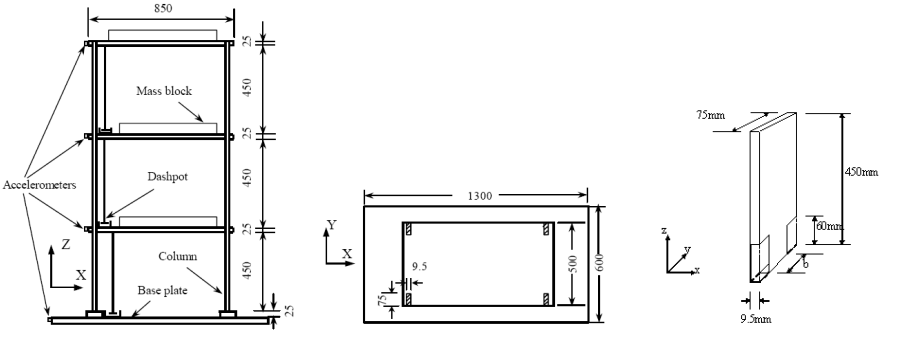
4.1 Configuration of 3-Storey Steel Frame

The 3-storey building frame was fabricated using three steel plates of 850×500×25 mm^3 with four equally sized rectangular columns of 9.5×75 mm^2 as shown in Fig. 1 a-b). The plates and columns were properly welded to form rigid connections. The building model was then welded on a steel base plate of 20 mm thickness. The steel base plate was in turn bolted firmly on a shaking table using a total of eight bolts of high tensile strength. The overall dimensions of the building were 1450×850×500 mm^3. All the columns were made of high strength steel of 435 MPa yield stress and 200 GPa modulus of elasticity. The 9.5×75 mm^2 cross-section of the column was

arranged in such a way that the first natural frequency of the building was much lower in the *x*-direction than in the *y*-direction. This arrangement restricted the building motion in the *x*-direction and thus the building was effectively reduced to planar building in the *x*-*z* plane. The thickness of each steel floor was 25 mm so that the floor can be regarded as a rigid plated in the horizontal direction, leading to a shearing type of deformation. The geometric scale of the building model was assumed to be 1/5. To do a better simulation, an additional mass block of 135 kg was placed on each floor of the building model.

4.2 Structural Multi-damage Detection

Four damage patterns considered in the experimental studies are shown in Table 1. Here the stiffness reduction extent (SRE) is computed based on shear build assumptions. They were implemented step by step by cutting the width of the columns, i.e., *b* as shown in Fig. 1 c), in the first storey to 51.30 mm (pattern i) and then to 37.46 mm (pattern ii) within a height of 60 mm from the bottom, followed by cutting the width of the columns in the second storey to 51.30 mm (pattern iii) and then to 37.46 mm (pattern iv) within a height of 60 mm from the second floor. The measured natural frequencies before and after damage for each damage pattern are listed in Table 2. Values in parentheses denote error (%) between measured natural frequencies from undamaged and damaged frames.



a) Elevation b) Plan c) Details of damaged column

Fig. 1. Configuration of 3-storey frame model (all dimensions in mm)

Table 1. Damage patterns of 3-storey frame model

Storey No.	undamage		damage patterns							
			(i)		(ii)		(iii)		(iv)	
	b/mm	SRE	*b*/mm	SRE	*b*/mm	SRE	*b*/mm	SRE	*b*/mm	SRE
1	75	0	51.30	11.6%	37.46	21.1%	37.46	21.1%	37.46	21.1%
2	75	0	75	0	75	0	51.30	11.6%	37.46	21.1%
3	75	0	75	0	75	0	75	0	75	0

Table 2. Measured natural frequencies (Hz) before and after damage

Modal No.	undamage	damage patterns			
		(i)	(ii)	(iii)	(iv)
1	3.369	3.259 (3.27%)	3.113 (7.60%)	3.076 (8.7%)	3.003 (10.86%)
2	9.704	9.485 (2.26%)	9.302 (4.14%)	9.192 (5.28%)	9.082 (6.40%)
3	14.282	14.209 (0.51%)	14.136 (1.02%)	13.660 (4.36%)	13.330 (6.67%)

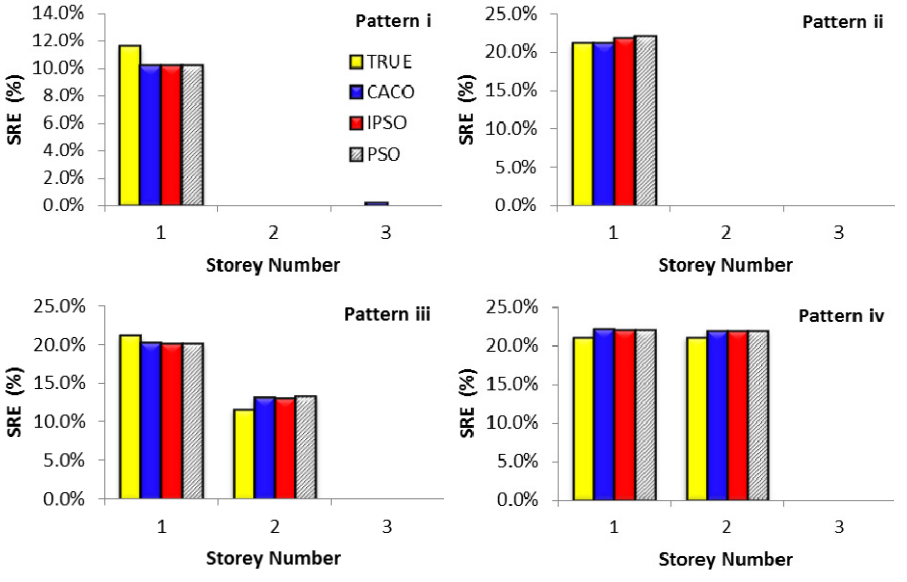


Fig. 2. Comparison on multi-damage detetion results

Since the identification model of the 3-storey frame structure (i.e., a 3-storey shear-building model) is simple, the computational cost is very small by using the proposed SI-based SDD algorithms, i.e. traditional PSO, proposed IPSO and CACO, and the convergence can be quickly achieved for all damage patterns. The damage identification results for all four damage patterns are compared in Fig. 2. It is very clear that both the identified damage location and extent are very close to the true ones, which shows that the proposed method is very effective.

5 Conclusions

A swarm intelligence (SI) based algorithm is proposed for optimal problems on structural damage detection (SDD) in structural health monitoring (SHM) field in this paper, which includes an improved particle swarm optimization (IPSO) and continuous ant colony optimization (CACO) techniques having been applied to the multi-damage identification of a 3-storey building model in laboratory. Further, assessment on the effectiveness and robustness of the proposed algorithm has also

been carried out. The illustrated results show that the identified damages are consistent with the true damages either for single damage or for multiple damage patterns. Moreover, the proposed SI-based algorithm can not only locate the structural damages but also quantify the severity of damages, which shows that the SI-based algorithm is feasible and effective for the SDD optimal problem in the SHM field.

Acknowledgments. The project is jointly supported by the National Natural Science Foundation of China (50978123 and 11032005), the Guangdong Natural Science Foundation (10151063201000022) and the Fundamental Research Funds for the Central Universities (21609601).

References

1. Farrar, C.R., Worden, K.: An introduction to structural health monitoring. Phil. Trans. R. Soc. A 365, 303–315 (2007)
2. Fan, W., Qiao, P.Z.: Vibration-based Damage Identification Methods: A Review and Comparative Study. Struct. Health Monit. 10(1), 83–111 (2011)
3. Yan, Y., Cheng, L., Wu, Z., Yam, L.: Development in vibration-based structural damage detection technique. Mech. Syst. Signal Process. 21, 2198–2211 (2007)
4. Yu, L., Xu, P.: Structural health monitoring based on continuous ACO method. Microelectron. Reliab. 51, 270–278 (2011)
5. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm intelligence—from natural to artificial systems. Oxford University Press, Oxford (1999)
6. Wan, Z.Y., Zhu, H.P., Yu, L.: Structural damage detection based on an improved PSO algorithm. Gongcheng Lixue/Engineering Mechanics 23 (sup.I), 73–78 (2006) (in Chinese)
7. Socha, K., Dorigo, M.: Ant colony optimization for continuous domains. Eur. J. Oper. Res. 185, 1155–1173 (2008)
8. Zhu, H.P., Xu, Y.L.: Damage detection of mono-coupled periodic structures based on sensitivity analysis of modal parameters. J. Sound Vib. 285(1), 365–390 (2005)

A Quantum-inspired Bacterial Swarming Optimization Algorithm for Discrete Optimization Problems

Jinlong Cao[1] and Hongyuan Gao[2]

[1] School of Information and Communication Engineering,
Beijing University of Posts and Telecommunications, Beijing, China
[2] College of Information and Communication Engineering,
Harbin Engineering University, Harbin, China
lansebolang2008@163.com, gaohongyuan@hrbeu.edu.cn

Abstract. In order to solve discrete optimization problem, this paper proposes a quantum-inspired bacterial swarming optimization (QBSO) algorithm based on bacterial foraging optimization (BFO). The proposed QBSO algorithm applies the quantum computing theory to bacterial foraging optimization, and thus has the advantages of both quantum computing theory and bacterial foraging optimization. Also, we use the swarming pattern of birds in block introduced in particle swarm optimization (PSO). Then we evaluate the efficiency of the proposed QBSO algorithm through four classical benchmark functions. Simulation results show that the designed algorithm is superior to some previous intelligence algorithms in both convergence rate and convergence accuracy.

Keywords: quantum-inspired bacterial swarming optimization, bacterial foraging optimization, particle swarm optimization.

1 Introduction

The natural system that has developed so long is one of the rich sources of inspiration for inventing new intelligence algorithms. Some intelligence algorithms are widely studied for application, such as particle swarm optimization (PSO) [1]. Particle swarm optimization were successfully applied to solve engineering problem of discrete optimization [2]. Quantum information science is a result of merging physical science into information science. Quantum-inspired genetic algorithm (QGA) is the product of quantum computing theory and genetic algorithm. In QGA, qubit encoding is used to represent the chromosome, and evolutionary process is implemented by using quantum logic gate operation on the chromosomes. Now, much attention is paid to QGA because it has the characteristics of strong searching capability, rapid convergence, short computing time and small population size [3–4].Quantum particle swarm optimization (QPSO) is an effective swarm intelligence method for multi-user detection [5]. So the quantum theory is very efficient in the intelligence algorithm domain.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 29–36, 2012.

In recent years, a new and rapidly growing subject - bacterial foraging optimization (BFO) attracts more and more attentions. BFO has been applied to many kinds of real world optimization problems, such as harmonic signal estimation [6]. Although BFO has prominent and encouraging performance for global optimization problems as reported in [6], simulation results had proved that the algorithm is time-consuming. The BFO in literature can't solve discrete problem. In order to design the algorithm that can solve discrete problem, we introduce the concept of quantum theory to the BFO, and adapt the process of the BFO to accelerate the convergence rate, thus we propose the quantum-inspired bacterial swarming optimization (QBSO) algorithm.

The rest of the paper is organized as follows. In Section 2, we describe the process of the QBSO algorithm. In Section 3, we evaluate the performance of the proposed algorithm through several Benchmark functions. In Section 4, conclusions and future work are summarized.

2 Quantum-inspired Bacterial Swarming Optimization

The classical BFO algorithm comprises the following processes: chemotaxis, swarming, reproduction, elimination and dispersal. The process of the QBSO algorithm mainly comprises three steps: chemotaxis, reproduction and elimination-dispersal. For simplicity, we eliminate the process of cell-to-cell communications. From the process of the BFO algorithm, we can see it is very complex, and simulation results show that it has low convergence rate and inaccurate convergence value. In order to solve discrete optimization problem, we design the QBSO algorithm which is based on the conventional BFO algorithm to get a better performance.

First we introduce the process of the chemotaxis. In the QBSO algorithm, a number of different representations can be used to encode the solutions onto the quantum bacterium. The QBSO algorithm uses quantum coding, called a quantum bit or Q-bit, for the probabilistic representation that is based on the concept of quantum bit, and a quantum bit position is defined as a string of quantum bits. One quantum bit is defined as the smallest unit of information in the QBSO, which is defined as a pair of composite numbers $(\alpha,\beta)^T$, where $|\alpha|^2+|\beta|^2=1$. $|\alpha|^2$ gives the probability that the quantum bit will be found in the '0' state and $|\beta|^2$ gives the probability that the quantum bit will be found in the '1' state. The quantum bit position of the *i*-th quantum bacterium at the *m*th chemotactic step of the *n*th reproduction loop in the *p*th elimination-dispersal event in the population of the *S* bacteria is defined as

$$\boldsymbol{\psi}_i(m,n,p)=\begin{bmatrix}\alpha_{i1}(m,n,p) & \alpha_{i2}(m,n,p) & \cdots & \alpha_{il}(m,n,p)\\ \beta_{i1}(m,n,p) & \beta_{i2}(m,n,p) & \cdots & \beta_{il}(m,n,p)\end{bmatrix} \tag{1}$$

where $|\alpha_{ij}(m,n,p)|^2+|\beta_{ij}(m,n,p)|^2=1$, $(j=1,2,\cdots,l)$, l represents the dimension of the problem, the quantum bit position can represent 2^l states simultaneously. For simple and efficient design of the QBSO algorithm, we define $\alpha_{ij}(m,n,p)$ and $\beta_{ij}(m,n,p)$ as real numbers and $0\le\alpha_{ij}(m,n,p)\le 1$,

$0 \le \beta_{ij}(m,n,p) \le 1$.Therefore, $\alpha_{ij}(m,n,p) = \sqrt{1-\left(\beta_{ij}(m,n,p)\right)^2}$, and equation (1) can be simplified as

$$\begin{aligned} \boldsymbol{\psi}_i(m,n,p) &= \left[\alpha_{i1}(m,n,p) \quad \alpha_{i2}(m,n,p) \quad \cdots \quad \alpha_{il}(m,n,p)\right] \\ &= \left[\psi_{i1}(m,n,p) \quad \psi_{i2}(m,n,p) \quad \cdots \quad \psi_{il}(m,n,p)\right] \end{aligned} \tag{2}$$

The evolutionary process of quantum bit position is mainly completed through quantum rotation gate [7]. In our algorithm, for simplicity, the *j*-th quantum bit ψ_{ij} is updated as

$$\psi_{ij}(m+1,n,p) = \mathrm{abs}\left(\psi_{ij}(m,n,p) \times \cos\theta_{ij}^{t+1} - \sqrt{1-\left(\psi_{ij}(m,n,p)\right)^2} \times \sin\theta_{ij}^{t+1}\right) \tag{3}$$

Where abs(·) is an absolute function which makes the quantum bit in the real domain [0, 1], and θ_{ij}^{t+1} is the quantum rotation angle, which can be calculated through equation (5), t is the iteration number of the algorithm, where $t = N_c \cdot (n-1) + N_c \cdot N_{re} \cdot (p-1) + m$, N_c represents the number of chemotaxis step, N_{re} represents the number of reproduction step, N_{ed} represents the number of elimination-dispersal step.

If $\theta_{ij}^{t+1} = 0$, a quantum bit ψ_{ij} is updated in a certain small probability by the operator which can be described below.

$$\psi_{ij}(m+1,n,p) = \sqrt{1-\left(\psi_{ij}(m,n,p)\right)^2} \tag{4}$$

Quantum-inspired bacterial swarming optimization algorithm is a novel multi-agent optimization system inspired by social behavior metaphor of agents. Each agent, called quantum bacterium, forages in an *l*-dimensional space according to the historical experiences of its own and its colleagues'. There are S quantum bacteria that are in a space of *l* dimensions in a quantum swarm, the *i*-th quantum bacterium's bit position in the space is $\mathbf{x}_i(m,n,p) = [x_{i1}, x_{i2}, \cdots, x_{il}]$, ($i = 1,2,\cdots,S$), which is a latent solution. The *i*-th quantum bacterium's quantum bit position at the mth chemotactic step of the nth reproduction loop in the pth elimination-dispersal event can be written as $\boldsymbol{\psi}_i(m,n,p) = [\psi_{i1}(m,n,p), \psi_{i2}(m,n,p), \cdots, \psi_{il}(m,n,p)]$ and $\mathbf{b}(m,n,p) = [b_1(m,n,p), b_2(m,n,p), \cdots, b_l(m,n,p)]$ is the global optimal bit position discovered by the whole quantum bacterium population until now. Let $\mathbf{x}_i(m,n,p)$ is the position of the ith bacterium at the mth chemotactic step of the nth reproduction loop in the pth elimination-dispersal event in the population of the S quantum bacteria. The *i*-th quantum bacterium is updated by the following quantum moving equations:

$$\theta_{ij}^{t+1} = e_1\left(b_j\left(m,n,p\right) - x_{ij}\left(m,n,p\right)\right) \tag{5}$$

$$\psi_{ij}\left(m+1,n,p\right) = \begin{cases} \sqrt{1-\left(\psi_{ij}\left(m,n,p\right)\right)^2}, & \text{if } \left(\theta_{ij}^{t+1}=0 \text{ and } \rho_{ij}\left(m+1,n,p\right)<c_1\right); \\ \text{abs}\left(\psi_{ij}\left(m,n,p\right)\times\cos\theta_{ij}^{t+1} - \sqrt{1-\left(\psi_{ij}\left(m,n,p\right)\right)^2}\times\sin\theta_{ij}^{t+1}\right), & \text{else.} \end{cases} \tag{6}$$

$$x_{ij}\left(m+1,n,p\right) = \begin{cases} 1, & \text{if} \quad \gamma_{ij}\left(m+1,n,p\right) > \left(\psi_{ij}\left(m+1,n,p\right)\right)^2; \\ 0, & \text{if} \quad \gamma_{ij}\left(m+1,n,p\right) \le \left(\psi_{ij}\left(m+1,n,p\right)\right)^2. \end{cases} \tag{7}$$

where $\left(i=1,2,\cdots,S\right)$, $\left(j=1,2,\cdots,l\right)$, ρ_{ij} is uniform random number between 0 and 1, c_1 is mutation probability which is a constant among $[0,1/l]$, $\gamma_{ij}\in[0,1]$ is uniform random number, $\left(\psi_{ij}\left(m+1,n,p\right)\right)^2$ represents the selection probability of bit position state. The value of e_1 expresses the relative important degree of **b** in the food foraging process, we can define the value of e_1 as attracting effect factor, which is similar to cell to cell communications.

The fitness value of the ith quantum bacterium at $\mathbf{x}_i\left(m,n,p\right)$ is represented by $J^i\left(m,n,p\right)$. In this paper the minimum fitness value $J_{\min}$ is defined as the global optimum.

After N_c chemotactic steps, the fitness values for the ith bacterium in the chemotactic loop are accumulated and calculated by:

$$J_{health}^i = \sum_{m=1}^{N_c+1} J^i\left(m,n,p\right) \tag{8}$$

where J_{health}^i represents the degree of health of the ith quantum bacterium.

The smaller the J_{health}^i is, the healthier the bacterium is. To simulate the reproduction character in nature and to accelerate the swarming speed, all the bacteria are sorted according to their health values in an ascending order and each of the first S_r ($S_r = S/2$, for convenience S is assumed to be a positive even integer) bacteria splits into two bacteria with no mutations. The characters including location and step length of the mother bacterium are reproduced to the children bacteria. Through this selection process the remaining S_r unhealthier bacteria are eliminated and discarded. Therefore, the number of the bacteria keeps constant in the whole process.

After N_c chemotactic steps, we adopt the reproduction process which is similar to the BFO algorithm, namely, the S_r bacteria with the highest J_{health} values die and the other S_r bacteria with the best values split.

For the purpose of improving the global search ability, elimination-dispersal event is defined after N_{re} steps of reproduction. The bacteria are eliminated and dispersed to random positions in the optimization domain according to the probability p_{ed}. This elimination-dispersal event helps the bacterium avoid being trapped into local optima. The number of the event is denoted as N_{ed}.

The process of the proposed QBSO algorithm can be summarized as follows:

Initialize simulation parameters, including $N_c, N_{re}, N_{ed}, S, p_{ed}$, the quantum position of each bacterium and the bit position of each bacterium.

Evaluate the performance of each bacterium.

for $p = 1 : N_{ed}$

for $n = 1 : N_{re}$

for $m = 1 : N_c$

Update the quantum position of each quantum bacterium according to equation (5) and (6), the bit position of each quantum bacterium is updated according to (7).

Update the global optimal position discovered by the whole quantum bacterium population until now.

end

Calculated the accumulated fitness values for each bacterium in the chemotactic loop, and the S_r bacteria with the highest J_{health} die and the other S_r bacteria each split into two bacteria with no mutations.

end

Each bacterium is eliminated and dispersed to random bit position and quantum position in the optimization domain according to the probability p_{ed}.

end

The global bit position of the bacteria can be obtained in vector **b** .

From what we have discussed above, we can see that we introduce the idea or the advantage of BFO to design the QBSO algorithm, and simultaneously we introduce the concept of quantum to accelerate the convergence rate. So our algorithm has the advantage of both the BFO algorithm and quantum theory.

3 The Performance of the QBSO Algorithm

We use four benchmark functions $\left(F_1(\mathbf{x}), F_2(\mathbf{x}), F_3(\mathbf{x}), F_4(\mathbf{x})\right)$ to evaluate the performance of the QBSO algorithm. We set initial population and maximum generation of the four evolutionary algorithms identical. For GA, QGA, PSO and QBSO, the population size is set to 20, i.e., $S = 20$ in the QBSO algorithm. For GA, the crossover probability and the mutation probability are set to 0.8 and 0.02, respectively, and the GA is configured to replace 85% of its population each generation, 17 of every 20 population members. As for QGA, the rotation angle of quantum gates decreases linearly from 0.1π at the first generation to 0.005π at the

last generation. In PSO, the two acceleration coefficients are equal to 2, and $V_{\max} = 4$ [8]. For QBSO, we set the number of chemotaxis step $N_c = 100$, the number of reproduction step is $N_{re} = 5$, the number of elimination-dispersal step is $N_{ed} = 2$, so that the maximum generation of the problem is $N_c \cdot N_{re} \cdot N_{ed} = 1000$, which is convenient to compare our algorithm with other classical algorithms, and the attracting effect factor $e_1 = 0.12$, $c_1 = 0.1/l$ and $p_{ed} = 0.25$.

$$F_1(\mathbf{x}) = \frac{1}{4000}\left(\sum_{i=1}^{n}(x_i - 100)^2\right) - \left(\prod_{i=1}^{n}\cos\left(\frac{x_i - 100}{\sqrt{i}}\right)\right) + 1, (-600 \le x_i \le 600, i = 1, 2, \cdots n)$$

$$F_2(\mathbf{x}) = \sum_{i=1}^{n-1} 100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2, (-50 \le x_i \le 50, i = 1, 2, \cdots n)$$

$$F_3(\mathbf{x}) = 2 \times 418.9829 + \sum_{i=1}^{n} -x_i \sin\left(\sqrt{|x_i|}\right), (-500 \le x_i \le 500, i = 1, 2, \cdots n)$$

$$F_4(\mathbf{x}) = \sum_{i=1}^{n}\left(x_i^2 - 10\cos(2\pi x_i) + 10\right), (-5.12 \le x_i \le 5.12, i = 1, 2, \cdots n)$$

In the following simulations, we use binary-encoding, and the length of every variable is 15 bits. We also set $n = 2$ for all benchmark functions, i.e. $i = 1, 2$. All the results are the average of 200 times.

The first function we use is Griewank function. $\mathbf{x}$ is in the interval of $[-600, 600]$. The global minimum value for this function is 0 and the corresponding global optimum solution is $\mathbf{x}_{\text{opt}} = (x_1, x_2, \cdots, x_n) = (100, 100, \cdots, 100)$. From Figure 1, we can see that although classic algorithms have fast convergence rate, but they all trap into local convergence. The figure also presents that our algorithm has the potential to have a much smaller convergence value. So our algorithm overcomes the disadvantage of local convergence and has a more accurate convergence value.

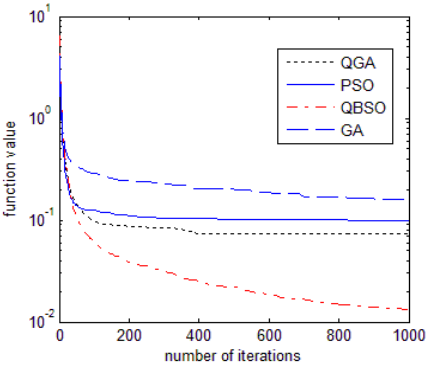


Fig. 1. The performance of four algorithms using Griewank function

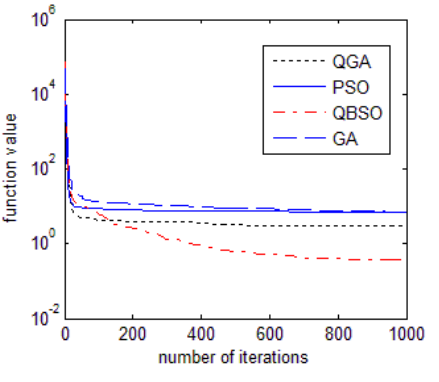


Fig. 2. The performance of four algorithms using Rosenbrock function

The second function is Rosenbrock function. $\mathbf{x}$ is in the interval of $[-50,50]$. Global minimum value for this function is 0 and the corresponding global optimum solution is $\mathbf{x}_{\text{opt}} = (x_1, x_2, \cdots, x_n) = (1,1,\cdots,1)$. From Figure 2, we can see that GA and PSO have the similar performance, while QGA outperforms GA and PSO. Our algorithm has a very accurate convergence value compared to the other three algorithms, which reaches the convergence value at 150 iterations while QGA obtains the same value at 1000 iterations.

The third function is Schwefel function whose value is 0 at its global minimum solution $\mathbf{x}_{\text{opt}} = (x_1, x_2, \cdots, x_n) = (420.9867, 420.9867, \cdots, 420.9867)$. $\mathbf{x}$ is in the interval of $[-500,500]$. The function has a second best minimum far from the global minimum where many search algorithms are trapped. Moreover, the global minimum is near the bounds of the domain. From Figure 3, we can see that although QGA has a fast convergence rate, it has an inaccurate convergence value; although PSO and GA have more accurate convergence value, they all have a slow convergence rate while our algorithm outperforms GA, QGA and PSO.

The fourth function is Rastrigin function whose value is 0 at its global minimum solution $\mathbf{x}_{\text{opt}} = (x_1, x_2, \cdots, x_n) = (0,0,\cdots,0)$. $\mathbf{x}$ is in the interval of $[-5.12,5.12]$. The function is based on Sphere function with the addition of cosine modulation to produce many local minima. The locations of the minima are regularly distributed. The difficult part about finding optimal solutions to this function is that an optimization algorithm can easily be trapped in a local optimum on its way towards the global optimum. From Figure 4, we can see that GA, QGA and PSO have similar performance, but they all trap into local convergence. Our algorithm overcomes the disadvantage of local convergence and has much higher convergence value.

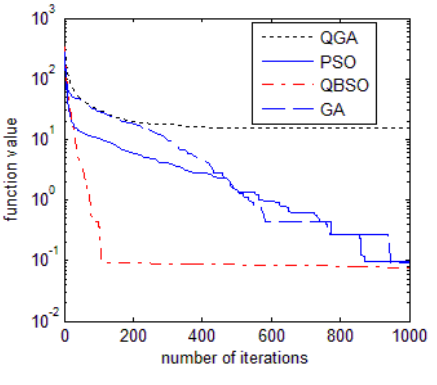


Fig. 3. The performance of four algorithms using Schwefel function

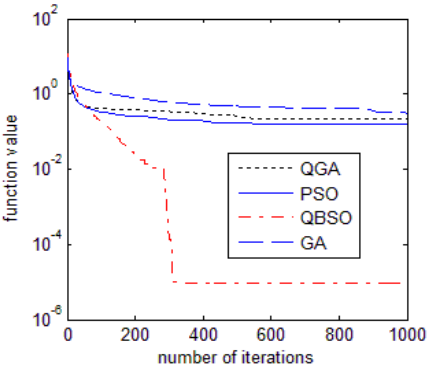


Fig. 4. The performance of four algorithms using Rastrigin function

4 Conclusion and Future Work

This paper has proposed the QBSO algorithm which is a novel algorithm for discrete optimization problems. Though testing classical Benchmark functions, we can see that our algorithm outperforms other classical evolutionary algorithms. In this paper, some parameters are not the best, so through changing certain parameters we can get a better result. On the other hand, the QBSO algorithm can only solve single objective problems, but can't solve multi-objective problems.

Acknowledgements. This work was supported by National Natural Science Foundation of China (No. 61102106 and No. 61102105).

References

1. Kennedy, J., Eberhart, J.: Discrete binary version of the particle swarm optimization. In: Proc. IEEE International Conference on Computational Cybernetics and Simulation, pp. 4104–4108 (1997)
2. Zhao, Y., Zheng, J.L.: Multiuser detection using the particle swarm optimization algorithm in DS-CDMA communication systems. J. Tsinghua. Univ (Sci. & Tech.) 44, 840–842 (2004)
3. Han, K.H., Kim, J.H.: Genetic quantum algorithm and its application to combinatorial optimization problems. In: Proceedings of the 2000 IEEE Conference on Evolutionary Computation, pp. 1354–1360. IEEE Press, Piscataway (2000)
4. Li, B., Zhuang, Z.-Q.: Genetic Algorithm Based-On the Quantum Probability Representation. In: Yin, H., Allinson, N.M., Freeman, R., Keane, J.A., Hubbard, S. (eds.) IDEAL 2002. LNCS, vol. 2412, pp. 500–505. Springer, Heidelberg (2002)
5. Gao, H.Y., Diao, M.: Quantum particle swarm optimization for MC-CDMA multiuser detection. In: 2009 International Conference on Artificial Intelligence and Computational Intelligence, vol. 2, pp. 132–136 (2009)
6. Mishra, S.: A hybrid least square-fuzzy bacterial foraging strategy for harmonic estimation. IEEE Transaction on Evolutionary Computation 9, 61–73 (2005)
7. Gao, H.Y., Cao, J.L., Diao, M.: A simple quantum-inspired particle swarm optimization and its application. Information Technology Journal 10(12), 2315–2321 (2011)
8. Zhao, Z.J., Peng, Z., Zheng, S.L., Shang, J.N.: Cognitive radio spectrum allocation using evolutionary algorithms. IEEE Transactions on Wireless Communications 8(9), 4421–4425 (2009)

Swarm Intelligence in Cloud Environment

Anirban Kundu and Chunlin Ji

Kuang-Chi Institute of Advanced Technology
Shenzhen, Guangdong, P.R. China 518057
{anirban.kundu,chunlin.ji}@kuang-chi.org

Abstract. In this paper, the major goal is to show the swarm intelligence power in cloud based scenario. Heterogeneous environment has been configured at server-side network of cloud. Swarm intelligence has been adopted for enhancing performance of overall system network. Specific location at server-side of network is going to be selected by swarm intelligence for accessing desired elements. Flexibility, robustness and self-organization have been considered as main features of swarm intelligence.

Keywords: Swarm Intelligence, Distributed system, Cloud.

1 Introduction

Swarm intelligence (SI) is the combined behavior of decentralized, self-organized systems, natural or artificial. The concept is engaged in effort on artificial intelligence. The expression was originally initiated by "Gerardo Beni" and "Jing Wang" in 1989, in the perspective of cellular robotic systems [1].

Swarm assumption has been built-up and discussed in [2]. Swarm behavior involves dynamism. SI systems are typically made up of a population of ordinary agents cooperating locally with one another and with their environment. The inspiration often comes from nature, especially biological systems. The agents follow very simple rules, and although there is no centralized control configuration dictating how individual agents should act, local, and to a certain degree arbitrary, communications among such agents direct to emergence of "intelligent" global behavior, unknown to the individual agents. Natural examples of SI include ant colonies, bird flocking, animal herding, bacterial growth, and fish schooling [3].

Group occurs at all magnitudes and across a variety of sequential permanence from the transient congregation of midges to the mandatory regulations of herring [4].

Organization of rest of the paper is as follows: Section 2 presents related works section. Proposed approach is described in Section 3. Experimental results have been shown in Section 4. Section 5 concludes the paper.

2 Related Works

A system is a collection of machines, workstations, servers and some other resources connected by networks. Distributed performance computing [5] in heterogeneous systems employs the distributed objects as applications. These applications are

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 37–44, 2012.

arranged in such a manner that the same type of user requests can be executed in distinct machines which are situated in different locations. Sometimes, these machines fall in the same group or cluster at same location [6]. Formation of distributed network requires transferring information in a high speed [7]. In real-time scenario, there may be several instances when server assignments are not appropriate. High performance can be achieved using cluster of workstations [8]. These workstations can be of similar types or distinct types in respect of configurations. Therefore, practical situation of the distributed network needs multi-processing power within the network activities to speed-up the task with synchronization.

3 Proposed Approach

In our proposed cloud, typically a lot of servers are being utilized to accommodate all users' requests in real-time basis based on requirement of physical memory, virtual memory, and disk space. Cloud has some interesting behaviors like servers are clustered as sub-network within the network. Cooperative transport is one of the major points for synchronization. Hierarchical structure of servers and workstations are maintained by specific protocols. These characteristics are also true in swarm intelligence. Peer-to-peer data transfer is accomplished by direct interactions whereas monitoring system handles all situations in network using indirect interactions. Sometimes, individual system behavior modifies cloud environment, which in turn modifies the behavior of other individuals of the system. This can be called as local-to-global and global-to-local transition. This indirect interaction is known as "Stigmergy" in swarm intelligence. Intelligent group activities are required for maintaining synchronization between all server machines within network. High computations are typically handled by the high-end cluster machines whereas small computations are handled by the low processors. These types of distinction of tasks are clearly visible in swarm activities as the division of labor and adaptive task allocation. In the proposed approach, nearest network and nearest server within that network has been utilized to execute client based programs using similarity concepts. This similarity concept is similar to the discovery of the shortest paths between source and destination. Collective behavior of swarm intelligence is visible in proposed cloud based system network. Feedback mechanism is used for controlling any abnormalities. It is called as load shifting within this paper. Random behavior in multiple interactions is also common in proposed cloud system and swarm system (refer Fig. 1).

In this paper, a distributed environment has been measured. It consists of typical computer resources, workstations and clusters. This gathering of equipments presents a huge widespread computing resource including memory, cycles, storage and bandwidth. Proposed system has a great prospective for high performance computing in this approach. An important characteristic of this structural arrangement is that it exhibits heterogeneity of many types including hardware, operating system, file system, and network. Heterogeneity creates a challenge that it must be managed to enable the parts of the proposed system to work together. At the same time, it also presents an opportunity that is the variety of different resources which suggests that it is possible to select the best resources for a particular user request. The variety and amount of computing resources in the proposed system offers a great prospective for high performance computing.

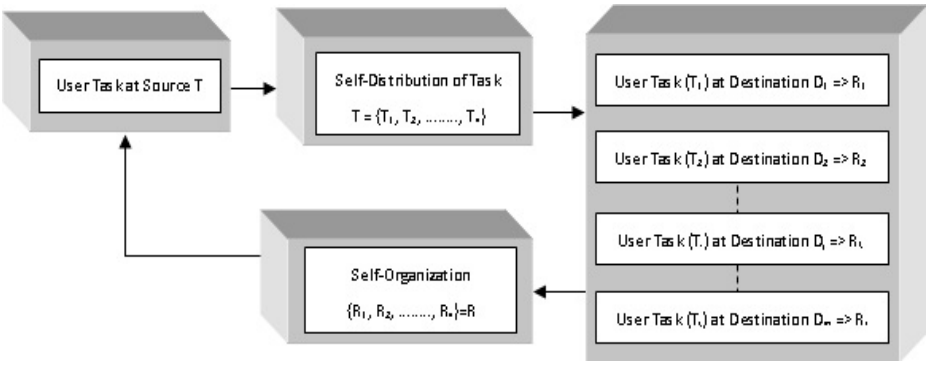


Fig. 1. Implementation of Self-Organization Concept in Cloud System Network

$$\text{SimilarityN}\big(\text{N(i)}, \text{N(j)}\big) = \frac{\text{AoAN}(\text{AppN}(\text{N(i)}) \cap \text{AppN}(\text{N(j)}))}{\text{AoAN}(\text{AppN}(\text{N(i)}) \cup \text{AppN}(\text{N(j)}))}. \quad (1)$$

where, N(i) = i^{th} network;
N(j) = j^{th} network;
AppN() = Function of "Number of Applications of individual network";
AoAN() = Function of "Amount of Applications of considered networks";
SimilarityN() = Function of "Similarity Measurement between networks";

$$\text{SimilarityS}\big(\text{S(i)}, \text{S(j)}\big) = \frac{\text{AoAS}(\text{AppS}(\text{S(i)}) \cap \text{AppS}(\text{S(j)}))}{\text{AoAS}(\text{AppS}(\text{S(i)}) \cup \text{AppS}(\text{S(j)}))}. \quad (2)$$

where, S(i) = i^{th} server;
S(j) = j^{th} server;
AppS() = Function of "Number of Applications of individual server";
AoAS() = Function of "Amount of Applications of considered servers";
SimilarityS() = Function of "Similarity Measurement between servers";

"SimilarityN" and "SimilarityS" are defined in Equation 1 and Equation 2 respectively to facilitate nearest network and nearest server. Related networks would be responsible to execute user specific tasks. Network manager directs the user initiated information and data to the nearest network for carrying out assignment. Related server is detected to process the user request. Output information would be sent to user-end following counter path.

Our system behaves as Software-as-a-Service (SaaS) providing full fledged implementation of network based activities with dynamic scalability, openness, distributive nature and transparencies. Software is customized according to users.

Designers should provide some specific methodologies for managing flawless activities in dynamic scalability in real-time. Algorithm 1 explains the overall cloud activity for specific applications in Web services.

Algorithm 1: Cloud Activity in Web Services

Input: Request from user (user application, user data with time stamp, user IP)
Output: Network selection for program execution of user
Step 1: Each server maintains the information of all other servers within a specific network. A network manager maintains the information about all other related networks within the Internet.
Step 2: When a client requests for data, the information related to the user is broadcasted on the Web.
Step 3: Determine nearest network (N_j)
Step 4: N_j detects the user request and accepts it.
Step 5: Specific server (S_k) is appointed for handling client requests (C_i).
Step 6: After the transaction is over, S_k sends the update of C_i to each related server of N_j. Therefore, the information is propagated to $\sum_{k=1}^{n} Sk$. Consider, total number of related servers within $N_j = n$.
Step 7: Calculate access frequency of the user request for particular application by comparing with other applications
Step 8: Network N_j transmits the update to other related networks.
Step 9: Each related network updates the related servers.
Step 10: Stop.

Nearest network has been determined using a seed which has been chosen based on user. Another seed is chosen based on the IP address of the user location. Hash function is selected considering the characteristics of these seeds. Particular network is selected comparing the current load with the threshold load of the network (refer Fig. 2).

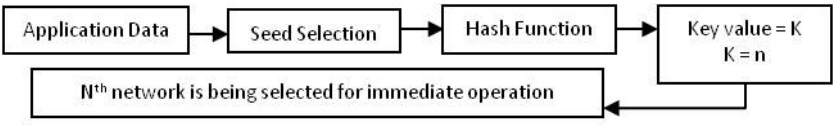


Fig. 2. Flowchart for network selection

There are mainly two points which have to be verified before sending the actual information. Higher access frequency is considered first for transmission over the network. Old information should be sent earlier in case of having same access frequencies. In similar way, information is transmitted to related servers after it is being reached to the specific network.

Initially at server-side, the document for client's Internet Protocol (IP) address has been stored by the Web portal system using TCP/IP connection and then further submitted to the interface layer to activate the proposed distributed system architecture. The user defined document for the input data for a specific program/activity should also be saved in the same procedure. I/P sender module takes care of these documents and prepare the required formatted materials for next level of processing to search for a particular sub-network within the server-side system based on the classified tasks using dynamic scheduling technique. Scheduler checks for an active server of a particular network at first, and then it checks whether the server is busy or free. The particular server is being selected for the user prescribed operation if the server is active and free at any time instance. Otherwise, another server would be

selected for the same operation. I/P sender module also send the required materials to I/P receiver module following the standardized typical communication techniques at port level. I/P receiver module initiate the particular application. The output files would be stored in a particular location of each server machines. O/P sender module collects the data and sends it back to the interface. O/P receiver module fetches the required output files coming to its direction at a particular port level and stores the required information to the prescribed places related to the external IP address of the particular user. Web portal takes care of these generated documents and sends them to the specific destination through TCP/IP connection (refer Fig. 3).

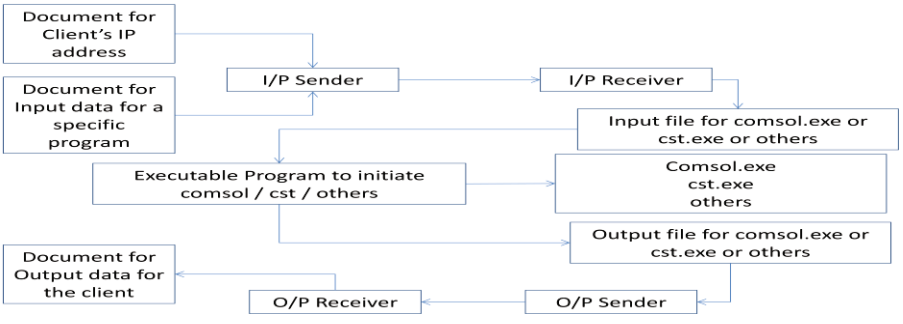


Fig. 3. Interaction between Sender & Receiver modules

Proposed system framework detects the nearest network and server for a particular application at any specific time instance based on the user request.

There is one option for handling emergency situation of system crash. It is known as load shift. The required output would not be received by the interface, if the server system crashes down. Therefore, the whole system network needs a supervision to tackle this type of scenario. A special monitoring sub-system at interface has been settled for monitoring about the status of particular network as well as particular server(s) of that network. If the monitor detects any unusual activity within network and/or server, it would execute the same procedure in some other servers of the same or different network. If there is no response from the network and/or server, then also the interface does the same procedure in an alternate network and/or server. A log file contains external IP information, the application name, internal IP information and the number of files required as output of the prescribed program module. Missing link means information about untraceable particular program request to a specific server of a specific sub-network. Algorithm 2 describes the load shifting technique as follows:

Algorithm 2: Emergency_Load_Shift

Input: Total list for a particular task, $T = \int_{i=1}^{n} T_i = \{T_1, T_2, \ldots, T_i, \ldots, T_n\}$

Output: Execute missing links to complete task

Step 1: Wait for 't' time units

Step 2: If (All links have reached interface) {Then goto Step 5}

Step 3: Else {Find missing links of $T = \int_{i=1}^{n} T_i$}

Step 4: Execute missing links in different servers (Sj) of sub-network (N_i)

Step 5: Stop

4 Experimental Results

All stages of proposed approach have been illustrated in this section. Fig. 4 demonstrates dynamic activity of IP_SEND module for controlling initial client data preserving organization between the interface and other related server machines of different internal networks of the disseminated setting. It would be necessary to switch the fractional productions of the user's program.

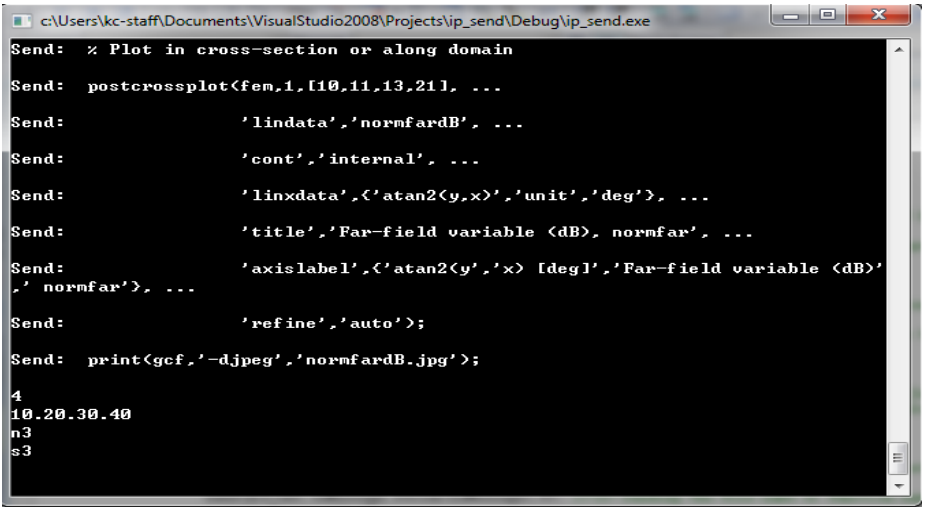


Fig. 4. Presentation of IP_SEND Module

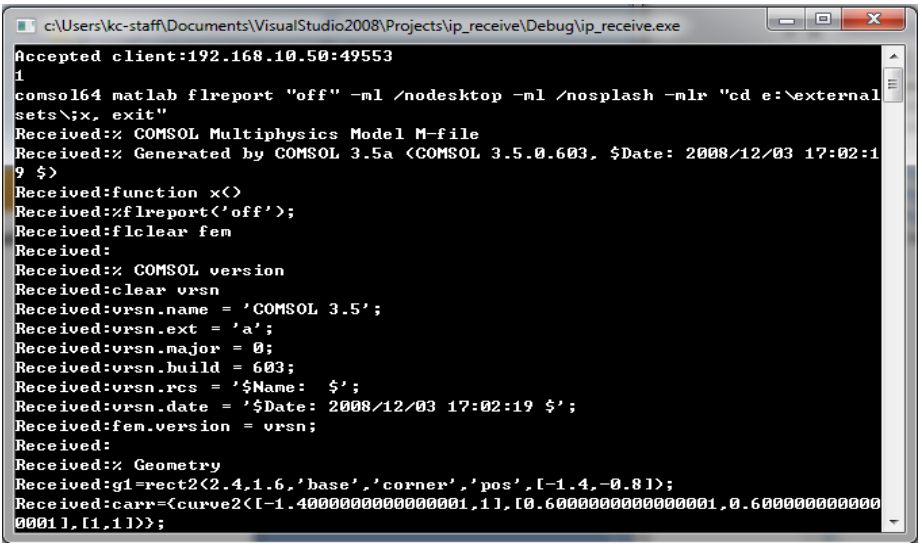


Fig. 5. Presentation of IP_RECEIVE Module

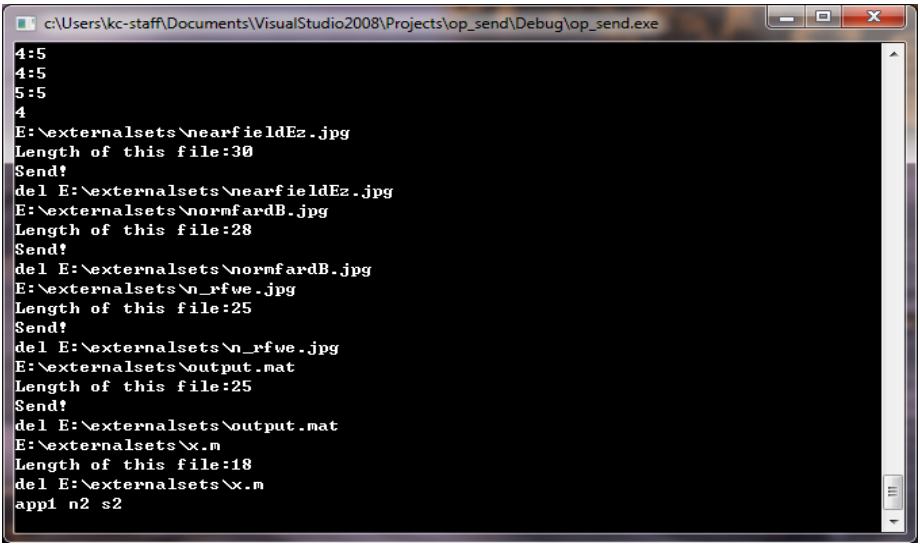


Fig. 6. Presentation of OP_SEND Module

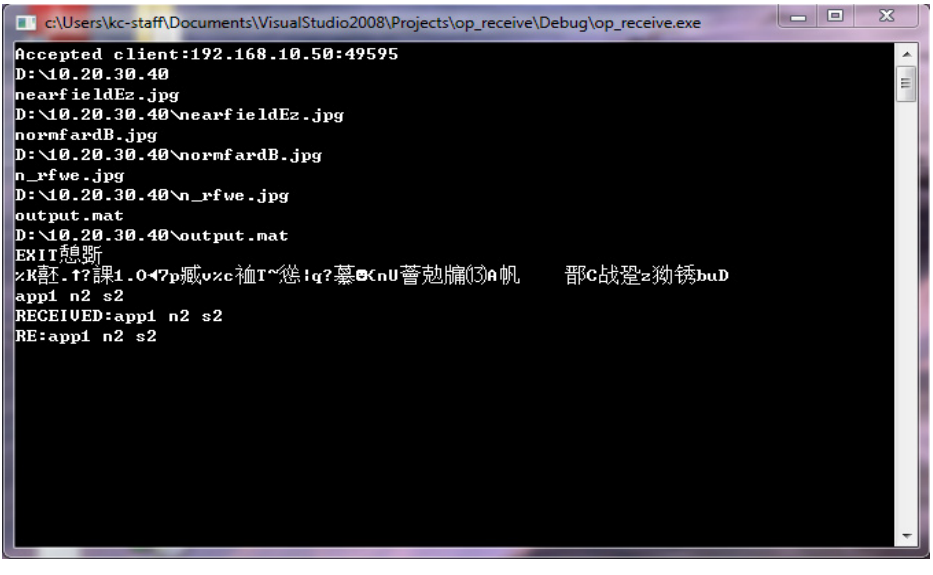


Fig. 7. Presentation of OP_RECEIVE Module

Fig. 5 depicts IP_RECEIVE unit accountable to carry out implementation of client's program. It receives output type along with server IP, nearest network and nearest server. It creates a result file irrespective of output types. The downloaded file(s) would be stored within predefined repository of the server in a provisional mode. Then, the module identifies the meticulous executables at kernel level. Fig. 6

symbolizes an example of real-time presentation of OP_SEND module. This unit receives outputs of coding preferred by users and then relocates these records to interface. Fig. 7 represents OP_RECEIVE module at interface. It receives outputs and stores it inside selected index. The specific server is being released after its execution.

5 Conclusion

In this paper, distributed environment has been achieved using swarm intelligence. Synchronized system network has been formed with different types of servers. "Stigmergy" is successfully applied to the proposed system for enhancing swarm concept. Self-organizing behavior is implemented artificially in the cloud to prepare and control the system as swarm movement in real-time.

Acknowledgments. This research was supported by grants from the National High Technology Research and Development Program of China (863 Program No. 2012AA030401) and Shenzhen Municipal Government of China (No. CXB201109210099A, No. CXB201109210103A).

References

1. http://en.wikipedia.org/wiki/Swarm_intelligence
2. Miller, P.: Swarm Theory. National Geographic Magazine (July 2007)
3. Ridge, E., Kudenko, D., Kazakov, D., Curry, E.: Moving Nature-Inspired Algorithms to Parallel, Asynchronous and Decentralised Environments. In: Self-Organization and Autonomic Informatics (I), vol. 135 (2005)
4. Kaiser, C., Kröckel, J., Bodendorf, F.: Swarm Intelligence for Analyzing Opinions in Online Communities. In: Proceedings of the 43rd Hawaii International Conference on System Sciences (2010)
5. Spinnato, P., Albada, G.D.V., Sloot, P.M.A.: Performance Modeling of Distributed Hybrid Architectures. IEEE Transactions on Parallel and Distributed Systems 15(1) (January 2004)
6. Liu, Y., Xiao, L., Liu, X., Ni, L.M., Zhang, X.: Location Awareness in Unstructured Peer-to-Peer Systems. IEEE Transactions on Parallel and Distributed Systems 16(2) (February 2005)
7. Eckart, B., He, X., Wu, Q., Xie, C.: A Dynamic Performance-Based Flow Control Method for High-Speed Data Transfer. IEEE Transactions on Parallel and Distributed Systems 21(1) (January 2010)
8. Pakin, S.: The Design and Implementation of a Domain-Specific Language for Network Performance Testing. IEEE Transactions on Parallel and Distributed Systems 18(10) (October 2007)

Swarm Intelligence Supported e-Remanufacturing

Bo Xing, Wen-Jing Gao, Fulufhelo V. Nelwamondo,
Kimberly Battle, and Tshilidzi Marwala

Faculty of Engineering and the Built Environment, University of Johannesburg,
Auckland Park, 2006, Johannesburg, South Africa
{wgao2011,bxing2009}@gmail.com

Abstract. e-Remanufacturing has nowadays become a superior option for product recovery management system. So far, many different approaches have been followed in order to increase the efficiency of remanufacturing process. Swarm intelligence (SI), a relatively new bio-inspired family of methods, seeks inspiration in the behavior of swarms of insects or other animals. After applied in other fields with success, SI started to gather the interest of researchers working in the field of remanufacturing. In this paper we provide a survey of SI methods that have been used in e-remanufacturing.

Keywords: swarm intelligence (SI), ant colony optimization (ACO), artificial bee colony (ABC), particle swarm optimization (PSO), artificial immune system (AIS), e-remanufacturing.

1 Introduction

Remanufacturing is an end-of-life strategy that reduces the use of raw materials and saves energy while preserving the value added during the design and manufacturing processes [1]. To successfully implement remanufacturing strategy, remanufacturers must manage a number of product recovery activities. e-Remanufacturing, an enhanced version of remanufacturing, is a process that consists of a set of e-activities which are based on and executed through information technologies.

2 SI in e-Remanufacturing

This section thoroughly reviews SI-based techniques used in e-remanufacturing. The papers that are presented thereafter are categorized primarily according to the following remanufacturing process, namely, e-retrieval, e-reproduction, and e-redistribution. For each category, a brief background is introduced first.

2.1 e-Retrieval

With the rapid development of advanced information technologies, e-retrieval has evolved into the use of electronic technologies to streamline and enable the

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 45–52, 2012.

procurement processes between the various parties. In e-remanufacturing context, the activities involved in e-retrieval can be further broken down into the following subclasses: e-collection, e-sorting, and e-logistics.

2.1.1 SI-Based Approaches in e-Collection

Internet of things (IoT), a novel paradigm where the pervasive presence around us a variety of things or objects which, through unique addressing schemes, are able to interact with each other and cooperate with their neighbors to reach common goals [2]. With the advent of IoT era, the way that how used products are collected will be dramatically changed. Equipped with their mobile phones, end users can trigger the used products return process by simply scanning the embedded information tags. The communications between end users and collectors will then be established through wireless sensor networks (WSN). In the literature, different SI techniques have been applied to WSN. In [3] a centralized approach to data gathering and communication for wireless sensor networks was introduced. The ant colony optimization (ACO) method is used in the based station to form a near-optimal chain. The simulation results showed that the proposed AntChain algorithm performs much better than other methods. In large-scale WSN, efficient service discovery and data transmission mechanisms are both essential and challenging. In [4], a novel scalable action-based service discovery protocol (ASDP) using ACO in WSN was introduced. The routing for WSN is also a key and hard problem. Inspired by some of biological systems such as ants and bees, the authors of [5], [6], and [7] introduced different adaptive intelligent routing scheme for WSN built on ACO and artificial bee colony (ABC). Interested readers please refer to [8] for more information about the applications of SI in routing for WSN. Apart from these, assuring a minimum level of security in WSN is also a hot research topic. Based on ACO, the authors of [9] presented a bio-inspired trust and reputation model, called BTRM-WSN, aiming at providing trust and reputation in WSN. We refer readers to [10] for more information about the applications of SI methods in this regard.

2.1.2 SI-Based Approaches in e-Sorting

Following the collection, and at some point in time, the inspection and sorting process starts [11]. After an item is inspected it is classified as either remanufacturable or un-remanufacturable. With the assistance of radio frequency identification (RFID), the life cycle information of a used product, especially its middle-of-life and end-of-life data is retrievable. Under these circumstances, SI-based data mining approaches play an important role. In [12], the authors reported the use of MAX-MIN ant systems in the data mining field capable of extracting comprehensible classifiers from data. A common problem in data mining is the presence of noise in the data being mined. The authors of [13] presented a study on knowledge acquirement from trained artificial neural networks for classification problems. The proposed method employed touring ant colony optimization (TACO) algorithm for extracting accurate and comprehensible rule from databases. In another study, an ant colony decision rule algorithm (ACDR) [14] was proposed to discovery classification rule in data sets of a distributed database.

Databases may become subject to missing values in either the data acquisition or data-storage process. Problems in an RFID sensor and a break in the data transmission line are prime examples of how data can go missing. Therefore, the author of [15] introduced various methods (e.g., the combination auto-associative neural networks with PSO, called AANN-PSO) dealing with issue. Overall the SI methods have achieved a tremendous success in data mining area. It is obvious that if the data patterns regarding used products can be recognized, the efficiency of the entire e-remanufacturing will certainly increase. Interested readers please refer to [16] for more detailed applications of SI in this regard.

2.1.3 SI-Based Approaches in e-Logistics

e-Logistics, in e-remanufacturing context, is applying the concepts of reverse logistics electronically to those aspects of business conducted via the Internet. Mediated by the advanced information technology, e-logistics revolutionizes the way of how used products are successfully remanufactured. With the aid of e-logistics, decentralized control becomes a possible, in which collection activities are outsourced to third-party or fourth party, remanufactured products are produced at remanufacturing facilities, while the redistribution of remanufactured products is rely on online channel. The unique characteristics of SI make it the most suitable for dealing with this problem. In [17], the author argued for self-organization of logistics systems based on principles derived from the foraging activities of ant colonies. Successful reverse logistics requires a cooperative integration between all the partners in the network. By modeling a generic supply chain with suppliers, logistics providers and distributors as a distributed optimization problem, the authors of [18] introduced ACO to deal with it. Lumpy demand forces capacity planners to maximize the profit of individual factories as well as simultaneously take advantage of outsourcing from its supply chain and even competitors. The study of [19] proposed an ant algorithm for solving a set of the addressed problems with different economic objectives and constraints of negotiating parties. In recent years, various multi-agent based models have been developed to automate buyer-seller negotiations in e-logistics applications. In [20], a hybrid case-based reasoning approach was presented to establish adaptive negotiation strategy for buyer-seller negotiations. Meanwhile a novel learning approach including such as PSO was introduced in a three-phase negotiation life cycle. In e-remanufacturing context, 4PL (i.e., forth party logistics provider) denotes those companies which "orchestrate" the supply chain, e.g., acquiring large sets of orders from large shippers and then re-distribute these orders among a set of other companies with actual transport capacity [21]. Therefore 4PL is often used as a return service provider in practice. In a recent study [22], the authors introduced a modified PSO with mutation operator extension to solve the problem of activity assignment of 4PL with preemptive structure.

2.2 e-Reproduction

Although sharing some similarities with the production of a new product, several additional constraints make the production of a remanufactured product more complex

than producing its counterpart new one. Thanks to the development of information technology, e-production provides a platform for remanufacturers to perform more cost effective and environmentally friendly remanufacturing operations than ever before. In e-remanufacturing context, the activities involved in e-production can be further broken down into the following subclasses: e-disassembly & assembly, and e-production planning & scheduling.

2.2.1 SI-Based Approaches in e-Disassembly and Assembly

The demand for product remanufacturing has forced companies to consider ease of assembly and disassembly during the design phase of their products. Evaluating these processes in a virtual environment during the early stages of design not only increases the impact of design modifications on the final product, but also eliminates the time, cost, and material associated with the construction of physical prototypes [23]. A disassembly or assembly sequence is considered to be optimal when it minimizes disassembly or assembly cost and satisfies the process constraints. The disassembly or assembly cost relates to disassembly or assembly operations, motions, and direction changes. Different SI techniques have been reported in the literature pertaining to this problem. In [24], an ACO method was utilized for generation of optimized robotic assembly sequences. The authors of [25] presented ANTBAL, an ACO algorithm for balancing mixed-model assembly lines. The proposed algorithm accounts for zoning constraints and parallel workstations and aims to minimize the number of operators in the assembly line for a given cycle time. Apart from this goal, ANTBAL also looked for solutions that smooth the workload among workstation. Another work dedicated to assembly line balancing is [26] in which a new mechanism to induce diversity in an existing multi-objective ACO algorithm for the 1/3 variant of the time and space assembly line balancing problem, a realistic variant often found in the automotive industry. In e-remanufacturing environment, sensors (e.g., RIFD) implanted into products during their production can address various uncertainties encountered at disassembly stage. A representative study can be found in [27]. To facilitate the use of RFID in such case, the antenna design of RFID is a critical factor that has to be taken into account in practice. Traditionally design engineers often construct small antennas using their knowledge and intuition, as there is no simple analytical solution relating antenna structure to performance. This, however, does not guarantee optimal results, particularly for larger, more complex antennas. Bear this in mind, ACO algorithm was employed in [28] for solving RFID antenna design problem. Computational results for a range of antenna sizes showed that ACO is a very effective design tool for RFID antennas.

2.2.2 SI-Based Approaches in e-Production Planning and Scheduling

Production planning and scheduling of remanufacturing is very complex in nature due to the presence of different uncertainties (e.g., yield of reusable parts) and under the consideration of the overall system's dynamics. Therefore it should always be handled differently from that of normal manufacturing. With the assistance of information

technology, computer-aided process planning (CAPP) forms an important interface between computer-aided design (CAD) and computer-aided manufacturing (CAM). In practice CAPP is concerned with determining the sequence of individual manufacturing operations required to produce a product as per technical specifications given in the part drawing. As the complexity of the product increases, the number of feasible sequences increases exponentially and there is a need to choose the best among them. In [29], an application of ACO algorithm as a global search technique for the quick identification of the optimal operations sequence by considering various feasibility constrains was presented. Other examples of utilizing SI approaches (e.g., ACO and PSO) in production planning and scheduling context can also be seen from the following work such as [30], [31], and [32].

2.3 e-Redistribution

The ultimate goal of remanufacturing is to resell remanufactured products to customers. Internet technology makes such remarketing more productive and more competitive. It allows remanufacturers to release the valuable information that resides within and make it available throughout the organization. The activities involved in this e-redistribution can be further broken down into the following subclasses, namely, e-auction, and e-commerce.

2.3.1 SI-Based Approaches in e-Auction

More recently, the developments in information technology are paving the way for an increasing use of the Internet for conducting auctions, popularly known as e-auction [33]. In [34] an online price quotation model was introduced for the purpose of negotiating the buy-back price and exchanging the new part with the old one. An enhanced PSO (EPSO) was introduced by the authors to generate a relatively good solution within a given time. Since agent-based technology is widely used in the area of e-auction, different learning strategies need to be developed. In this regard, SI techniques can also find a room for these applications. For example in [35], PSO is applied inside a co-evolutionary training environment to evolve the weights of the neural network for the purpose of game learning. Other instances of applying SI method to learning include the following example papers: [36], and [37].

2.3.2 SI-Based Approaches in e-Commerce

The novel technology Web 2.0, which is defined as “the philosophy of mutually maximizing collective intelligence and added values for each participant by formalized and dynamic information sharing and creation” [38], is believed will dramatically changed how people buy and sell goods. In the context of e-commerce, buyers must incur higher search costs to locate desired products within larger and larger numbers of products. In terms of web intelligence, the authors of [39] introduced an ACO-based method for the purpose of distinguishing irrelevant information and enhancing the amount of the relevant information in respect to a user’s query. In another study [40],

an ant-based web searching algorithm called Seangàn, was presented. To evaluate the performance of Seangàn, the authors run several experiments with information about culture, history, and other curiosities from the web pages of Ireland. In e-remanufacturing environment, increasing business-to-customer (B2C) e-commerce with a variety of small internet orders may have a tremendous impact on remanufacturers' warehousing system. Automated storage/retrieval system (AS/RS) is a valid alternative under these circumstances. In [41] the authors proposed an auto-access multilevel conveying device (IMCD) with three-dimensional movement is integrated into the AS/RS for handling B2C e-commerce logistics. In another paper [42], the authors employed two ACO algorithms, i.e., ant system (AS) and MAX-MIN ant system (MMAS), to deal with batch order picking (BOP) problem. The ant colony system (ACS) algorithm was employed in such study to locate near-optimal routes for the IMCD. In the work of [43], age artificial immune system (AAIS) was introduced for optimal order pickings in an AS/RS with multiple input/output stations.

3 Conclusions

During the past decades, with the rapid advances in information technology, a lot of efforts have been devoted to e-remanufacturing research. In this paper, we have provided an overview of the applications of SI-based approaches in such field. The relevant studies are clustered based on e-remanufacturing progress. Brief descriptions of representative papers are provided in our work. We hope this paper will be a useful starting point for those wishing to do future e-remanufacturing research or simply to keep abreast of the latest developments in this field.

References

1. Zwolinski, P., Lopez-Ontiveros, M.-A., Brissaud, D.: Integrated design of remanufacturable products based on product profiles. J. of Cleaner Production 14, 1333–1345 (2006)
2. Atzori, L., Iera, A., Morabito, G.: The Internet of things: a survey. Computer Networks 54, 2787–2805 (2010)
3. Ding, N.N.: Data gathering and communication for wireless sensor networks using ant colony optimization, MSc Thesis, in Department of System and Computer Engineering, Carleton University (2005)
4. Huo, H., Gao, D., Niu, Y., Gao, S.: ASDP: An Action-Based Service Discovery Protocol Using Ant Colony Algorithm in Wireless Sensor Networks. In: Zhang, H., Olariu, S., Cao, J., Johnson, D.B. (eds.) MSN 2007. LNCS, vol. 4864, pp. 338–349. Springer, Heidelberg (2007)
5. Wang, X., Li, Q., Xiong, N., Pan, Y.: Ant Colony Optimization-Based Location-Aware Routing for Wireless Sensor Networks. In: Li, Y., Huynh, D.T., Das, S.K., Du, D.-Z. (eds.) WASA 2008. LNCS, vol. 5258, pp. 109–120. Springer, Heidelberg (2008)
6. Iyengar, S.S., et al.: Biologically inspired cooperative routing for wireless mobile sensor networks. IEEE Systems Journal 1(1), 29–37 (2007)

7. Saleem, M., Farooq, M.: BeeSensor: A Bee-Inspired Power Aware Routing Protocol for Wireless Sensor Networks. In: Giacobini, M. (ed.) EvoWorkshops 2007. LNCS, vol. 4448, pp. 81–90. Springer, Heidelberg (2007)
8. Saleem, M., Caro, G.A.D., Farooq, M.: Swarm intelligence based routing protocol for wireless sensor networks: survey and future directions. Information Sciences 181, 4597–4624 (2011)
9. Mármol, F.G., Pérez, G.M.: Providing trust in wireless sensor networks using a bio-inspired technique. Telecommun. Syst. (2010), doi:10.1007/s11235-010-9281-7
10. Kolias, C., Kambourakis, G., Maragoudakis, M.: Swarm intelligence in intrusion detection: a survey. Computers & Security 30, 625–642 (2011)
11. Konstantaras, I., Skouri, K., Jaber, M.Y.: Lot sizing for a recoverable product with inspection and sorting. Computers & Industrial Engineering 58, 452–462 (2010)
12. De Backer, M., Haesen, R., Martens, D., Baesens, B.: A Stigmergy Based Approach to Data Mining. In: Zhang, S., Jarvis, R. (eds.) AI 2005. LNCS (LNAI), vol. 3809, pp. 975–978. Springer, Heidelberg (2005)
13. Özbakır, L., et al.: TACO-miner: an ant colony based algorithm for rule extraction from trained neural networks. Expert Systems with Applications 36, 12295–12305 (2009)
14. Xie, L., Mei, H.: The application of the ant colony decision rule algorithm on distributed data mining. Communications of the IIMA 7(4), 85–94 (2007)
15. Marwala, T.: Computational intelligence for missing data imputation, estimation and management: knowledge optimization techniques. IGI Global Publications, Information Science Reference Imprint, New York (2009)
16. Grosan, C., Abraham, A., Chis, M.: Swarm intelligence in data mining. SCI, vol. 34, pp. 1–20. Springer, Heidelberg (2006)
17. Shibeshi, A.G.: Ant based logistics self-organizaiton: starting fundamentals and research design. In: Faculty of Technology, Policy and Management (2009)
18. Silva, C.A., et al.: Distributed supply chain management using ant colony optimization. European J. of Operational Research 199, 349–358 (2009)
19. Wang, K.-J., Chen, M.-J.: Cooperative capacity planning and resource allocation by mutual outsourcing using ant algorithm in a decentralized supply chain. Expert Systems with Applications 36, 2831–2842 (2009)
20. Fang, F., Wong, T.N.: Applying hybrid case-based reasoning in agent-based negotiations for supply chain management. Expert Systems with Applications 37, 8322–8332 (2010)
21. Robu, V., et al.: A multi-agent platform for auction-based allocation of loads in transportation logistics. Expert Systems with Applications 38, 3483–3491 (2011)
22. Chen, K.-H., Su, C.-T.: Activity assigning of fourth party logistics by particle swarm optimization-based preemptive fuzzy integer goal programming. Expert Systems with Applications 37, 3630–3637 (2010)
23. Coutee, A.S.: Virtual assembly and disassembly analysis: an exploration into virtual object interactions and haptic feedback, in Mechanical Engineering. Georgia Institute of Technology (2004)
24. Sharma, S., et al.: Generation of optimized robotic assembly sequence using ant colony optimization. In: Proceedings of the 4th IEEE Conference on Automation Science and Engineering. Key Bridge Marriott. IEEE, Washington DC (2008)
25. Vilarinho, P.M., Simaria, A.S.: ANTBAL: an ant colony optimization algorithm for balancing mixed-model assembly lines with parallel workstations. Int. J. of Production Research 44(2), 291–303 (2006)

26. Chica, M., et al.: A new diversity induction mechanism for a multi-objective ant colony algorithm to solve a real-world time and space assembly line balancing problem. Memetic. Comp. (2010), doi:10.1007/s12293-010-0035-6
27. Ilgin, M.A., Gupta, S.M.: Recovery of sensor embedded washing machines using a multi-kanban controlled disassembly line. Robotics and Computer-Integrated Manufacturing 27, 318–334 (2011)
28. Lewis, A., Randall, M., Galehdar, A., Thiel, D., Weis, G.: Using Ant Colony Optimisation to Construct Meander-Line RFID Antennas. In: Lewis, A., Mostaghim, S., Randall, M. (eds.) Biologically-Inspired Optimisation Methods. SCI, vol. 210, pp. 189–217. Springer, Heidelberg (2009)
29. Krishna, A.G., Rao, K.M.: Optimisation of operations sequence in CAPP using an ant colony algorithm. Int. J. Adv. Manuf. Technol. 29, 159–164 (2006)
30. Mahdavi, I., et al.: P-ACO approach to assignment problem in FMSs. World Academy of Science, Engineering and Technology 42, 196–203 (2008)
31. Jerald, J., et al.: Scheduling optimisation of flexible manufacturing systems using particle swarm optimisation algorithm. Int. J. Adv. Manuf. Technol. 25, 964–971 (2005)
32. Li, L., Qiao, F., Wu, Q.: ACO-Based Scheduling of Parallel Batch Processing Machines with Incompatible Job Families to Minimize Total Weighted Tardiness. In: Dorigo, M., Birattari, M., Blum, C., Clerc, M., Stützle, T., Winfield, A.F.T. (eds.) ANTS 2008. LNCS, vol. 5217, pp. 219–226. Springer, Heidelberg (2008)
33. Gupta, S., Koulamas, C., Kyparisis, G.J.: e-Business: a review of research published in Production and Operations Management (1992-2008). Production and Operations Management 18(6), 604–620 (2009)
34. Li, S.G., Rong, Y.L.: The research of online price quotation for the automobile parts exchange programme. Int. J. Computer Integrated Manufacturing 22(3), 245–256 (2009)
35. Franken, C.J.: PSO-based coevolutionary game learning. In: Faculty of Engineering, Built-Environment and Information Technology, University of Pretoria, Pretoria (2004)
36. Wu, Y., McCall, J., Corne, D.: Two novel ant colony optimization approaches for Bayesian network structure learning. In: Proceedings of IEEE World Congress on Computational Intelligence (WCCI 2010). CCIB. IEEE, Barcelona (2010)
37. Wong, L.-H., Looi, C.-K.: Adaptable learning pathway generation with ant colony optimization. Educational Technology & Society 12(3), 309–326 (2009)
38. Wijaya, S., et al.: Web 2.0-based webstrategies for three different types of organizations. Computers in Human Behavior (2010), doi:10.1016/j.chb.2010.07.041
39. Kouzas, G., Kayafas, E., Loumos, V.: Ant Seeker: An Algorithm for Enhanced Web Search. In: Maglogiannis, I., Karpouzis, K., Bramer, M. (eds.) Artificial Intelligence Applications and Innovations. IFIP, vol. 204, pp. 649–656. Springer, Boston (2006)
40. Boryczka, U., Polak, I.: Cumulation of Pheromone Values in Web Searching Algorithm. In: Cyran, K.A., Kozielski, S., Peters, J.F., Stańczyk, U., Wakulicz-Deja, A. (eds.) Man-Machine Interactions. AISC, vol. 59, pp. 515–522. Springer, Heidelberg (2009)
41. Hu, K.-Y., Chang, T.-S.: An innovative automated storage and retrieval system for B2C e-commerce logistics. Int. J. Adv. Manuf. Technol. (2009), doi:10.1007/s00170-009-2292-4
42. Xing, B., et al.: Ant colony optimization for automated storage and retrieval system. In: Proceedings of The Annual IEEE Congress on Evolutionary Computation (IEEE CEC 2010). CCIB. IEEE, Barcelona (2010)
43. Mak, K.L., Lau, P.S.K.: Order pickings in an AS/RS with multiple I/O stations using an artificial immune system with aging antibodies. Engineering Letters 16(1) (2008)

Grey-Based Particle Swarm Optimization Algorithm

Ming-Feng Yeh, Cheng Wen, and Min-Shyang Leu

Department of Electrical Engineering, Lunghwa University of Science and Technology, Taoyuan, 33327 Taiwan
{mfyeh,chengwen,unit484}@mail.lhu.edu.tw

Abstract. In order to apply grey relational analysis to the evolutionary process, a modified grey relational analysis is introduced in this study. Then, with the help of such a grey relational analysis, this study also proposed a grey-based particle swarm optimization algorithm in which both inertia weight and acceleration coefficients are varying over the generations. In each generation, every particle has its own algorithm parameters and those parameters may differ for different particles. The proposed PSO algorithm is applied to solve the optimization problems of twelve test functions for illustration. Simulation results are compared with the other three variants of PSO to demonstrate the search performance of the proposed algorithm.

Keywords: Acceleration coefficients, Grey relational analysis, Inertia weight, Particle swarm optimization.

1 Introduction

Particle swarm optimization (PSO), introduced by Kennedy and Eberhart in 1995 [1]-[2], was inspired by the social behavior of bird flocking and fish schooling. The PSO uses a simple mechanism that imitates their swarm behaviors to guide the particles to search for globally optimal solutions. Similar to other evolutionary computation technique, it is also a population-based iterative algorithm. Owing to its simplicity of implementation and ability to quickly converge to a reasonably good solution [3], the PSO has been successfully applied in solving many real-world optimization problems [4]-[6]. On the other hand, grey system theory, introduced by Deng in 1989 [7], was proposed to solve the system with incomplete (partial known and partial unknown) information. In grey system theory, one of the essential topics is grey relational analysis which can perform as a similarity measure for finite sequences. Studies [8]-[9] have successfully shown that grey relational analysis can be applied to cluster analysis or other applications.

How to accelerate the convergence speed and how to avoid the local optimal solution are two important issues in the PSO research. Generally speaking, those issues can be solved by controlling of algorithm parameters, i.e., the inertia weight or the acceleration coefficients [3]. A linearly varying inertia weight [10] and the time-varying acceleration coefficients [11] have been widely used to improve the search performance of PSO. Those control schemes are based on with the corresponding

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 53–62, 2012.

iteration number using linear or nonlinear rules. The change rate of the inertia weight or the acceleration coefficients is depended upon a predefined maximum number of allowable iterations. Although the time-varying control schemes could be used to improve the PSO algorithm, each particle uses the identical inertia weight and the same acceleration coefficients to update the corresponding velocity vectors in each generation. That is to say, they may suffer from improperly updating the parameters because no information on the evolutionary state that reflects the diversity of the population is identified or utilized. This study attempts to propose a modified grey relational analysis such that it can use almost the same criterion to measure the relational grades in the evolutionary process. Then based on the modified approach, this study also proposes a novel approach to determine the inertia weight and acceleration coefficients such that each particle has its own parameters and those parameters may differ for different particles.

2 Grey Relational Analysis and Particle Swarm Optimization

2.1 Grey Relational Analysis

Grey relational analysis is a similarity measure for finite sequences with incomplete information [7]. Assume that the reference sequence is defined as $x = (x_1, x_2, x_3,\ldots, x_n)$ and the comparative sequences are given by $y_j = (y_{j1}, y_{j2}, \ldots, y_{jn})$, $j = 1, 2, 3, \ldots, m$. The grey relational coefficient between x and y_j at the kth datum, $k = 1, 2, 3, \ldots, n$, is defined as follows.

$$r(x_k, y_{jk}) = \frac{\Delta_{\min} + \xi \cdot \Delta_{\max}}{\Delta_{jk} + \xi \cdot \Delta_{\max}}, \tag{1}$$

where $\Delta_{jk} = |x_k - y_{jk}|$, $\Delta_{\max} = \max_j \max_k \Delta_{jk}$, $\Delta_{\min} = \min_j \min_k \Delta_{jk}$, and $\xi \in (0, 1]$, which is a distinguishing coefficient to control the resolution between $\Delta_{\max}$ and $\Delta_{\min}$. The corresponding grey relational grade is

$$g(x, y_j) = \sum_{k=1}^{n} \left[\alpha_k \cdot r(x_k, y_{jk})\right], \tag{2}$$

where α_k is the weighting factor of grey relational coefficient $r(x_k, y_{jk})$ and $\sum_{k=1}^{n} \alpha_k = 1$. The selection of the weighting factor for a relational coefficient reflects the importance of that datum. In general, we can select it as $\alpha_k = 1/n$ for all k. The best comparative sequence is determined as the one with the largest relational grade.

2.2 Particle Swarm Optimization and Its Variants

In PSO, a swarm of particles are represented as potential solution, and each particle i is associated with two vectors, i.e., the velocity vector $V_i = (v_{i1}, v_{i2}, ..., v_{iD})$ and the position vector $X_i = (x_{i1}, x_{i2}, ..., x_{iD})$, where D represents the dimensions of the solution

space. The velocity and the position of each particle are initialized by random vectors within the corresponding ranges. During the evolutionary process, the trajectory of each individual in the search space is adjusted by dynamically altering the velocity of each particle, according to its own flying experience (*pBest*) and the flying experience of the other particles (*gBest*) in the search space. That is, the velocity and position of the *i*th particle on dimension *d* are updated as

$$v_{id} = wv_{id} + c_1 rand_{1d}(pBest_{id} - x_{id}) + c_2 rand_{2d}(gBest_d - x_{id}), \tag{3}$$

$$x_{id} = x_{id} + v_{id}, \tag{4}$$

where w is the inertia weight, c_1 and c_2 are the acceleration coefficients, and $rand_{1d}$ and $rand_{2d}$ are two uniformly distributed random numbers independently generated within [0, 1] for the dth dimension [10]. In (3), $pBest_i$ represents the position with the best fitness found so far for the ith particle, and *gBest* is the best position discovered by the whole particles. In addition, the second and third parts of (3) are known as the "cognitive" and "social" components, respectively.

Except the original PSO algorithm, Shi and Eberhart in [10] proposed the PSO with a linearly varying inertia weight w over the generations (PSO-LVIW) to improve the performance of PSO. The corresponding mathematical representation is

$$w = w_{\max} - (w_{\max} - w_{\min})\frac{t}{T} \tag{5}$$

where t is the current generation number and T is a predefined maximum number of generations. Besides, the maximal and minimal weights $w_{\max}$ and $w_{\min}$ are usually set to 0.9 and 0.4, respectively.

The PSO algorithm with time-varying acceleration coefficients (PSO-TVAC) is another widely used strategy to improve the performance of PSO. With a large cognitive component (a larger c_1) and a small social component (a smaller c_2) at the beginning, particles are allowed to move around the search space, instead of moving toward the population best. On the other hand, a small cognitive component and a large social component allow the particles to converge to the global optima in the latter part of the evolutionary process. This modification can be mathematically represented as follows [11]:

$$c_1 = (c_{1f} - c_{1i})\frac{t}{T} + c_{1i} \tag{6}$$

$$c_2 = (c_{2f} - c_{2i})\frac{t}{T} + c_{2i} \tag{7}$$

where c_{1i}, c_{1f}, c_{2i}, and c_{2f} are constants.

3 Grey-Based Particle Swarm Optimization Algorithm

While the fittest particle *gBest* is regarded as the reference sequence and all particles X's are viewed as the comparative ones, grey relational analysis could be applied to

analyze the similarity between them. Then the values of both inertia weight and acceleration coefficients of a specific particle are determined according to the corresponding relational grade. Since the result of grey relational analysis may differ for different generations, the algorithm parameters are varying over the generations.

3.1 Modified Grey Relational Analysis

According to (1), grey relational coefficients between the fittest particle *gBest* and the *i*th particle X_i at the *d*th dimension can be rewritten as

$$r_{id} = r(gBest_d, x_{id}) = \frac{\Delta_{\min} + \xi \cdot \Delta_{\max}}{\Delta_{id} + \xi \cdot \Delta_{\max}}, \tag{8}$$

where $\Delta_{id} = |gBest_d - x_{id}|$, $\Delta_{\max} = \max_i \max_d \Delta_{id}$, $\Delta_{\min} = \min_i \min_d \Delta_{id}$, and $\xi \in (0, 1]$. Then the corresponding relational grade is given as

$$g_i = g(gBest, X_i) = \sum_{d=1}^{D} r_{id} / D, \tag{9}$$

where *D* represents the dimensions of the solution space. While grey relational analysis is applied to the evolution algorithm such as PSO algorithm, the extreme terms $\Delta_{\max}$ and $\Delta_{\min}$, given in (8), seem to be in the form of "local" version. That is to say, the determination of the extreme terms only takes the distribution of particles in that generation into consideration, but does not consider the population distribution in other generations. Therefore this study attempts to propose a "global" version grey relational analysis such that it can use almost the same criterion to measure the relational grades in the evolutionary process.

In the latter part of the search process, all particles will converge to the fittest particle. Among them, some of particles are identical to *gBest*. At this situation, $\Delta_{\min} = 0$. As far as the entire evolution process is considered, the global value of the minimal term $\Delta_{\min}$ therefore is 0. In other word, the minimal term $\Delta_{\min}$ can be omitted while computing the grey relational coefficients in the sense of "global" version. On the other hand, the global value of the maximal term, denoted by $\Delta^*_{\max}$, is depended on the range of the search space. Assume that the search space is $\prod_{d=1}^{D} [x_{d.\min}, x_{d.\max}]$. It can be seen that the upper bound of $\Delta^*_{\max}$ is $\max_d\{x_{d,\max} - x_{d,\min}\}$. Rather than using the upper bound as the actual value of $\Delta^*_{\max}$, this study utilizes the following updating rule to obtain that value. Let the initial value of $\Delta^*_{\max}$ be $\max_d\{x_{d,\max} - x_{d,\min}\}/4$. Then the adaption rule for $\Delta^*_{\max}$ is

If $\Delta_{\max} \geq \Delta^*_{\max}$, then $\Delta^*_{\max} = \Delta_{\max}$; otherwise, $\Delta^*_{\max}$ remains unchanged.

By this way, the actual value of $\Delta^*_{\max}$ can be obtained through the evolution process. With the help of $\Delta^*_{\max}$, the formula of grey relational coefficient becomes as

$$r_{id} = r(gBest_d, x_{id}) = \frac{\xi \cdot \Delta^*_{\max}}{\Delta_{id} + \xi \cdot \Delta^*_{\max}}. \tag{10}$$

As can be seen, $r_{id} \in [\xi/(1+\xi), 1]$. The result further imply that $g_i \in [\xi/(1+\xi), 1]$.

3.2 Grey-Based PSO Algorithm

Shi and Eberhart in [10] proposed a linearly varying inertia weight w over the generations to improve the performance of PSO. They had observed that the optimal solution can be improved by varying the value of w from 0.9 at the beginning of the search to 0.4 at the end of the search for most problems. A particle with a smaller relational grade generally represents that it is far away from the fittest particle. That particle therefore can be regarded as being in the exploration state. In this study, such a particle should be assigned a larger inertia weight. On the contrary, a particle with a larger relational grade is treated as being in the exploitation state. Hence that particle is assigned a smaller inertia weight. To sum up, the lager the grey relational grade g_i is, the smaller the inertia weight w_i is, and vice versa.

Owing to $g_i \in [\xi/(1+\xi), 1]$ and $w \in [0.4, 0.9]$, the relationship between g_i and w_i can be simply represented by the following linear scaling scheme:

$$w_i = -0.5(1+\xi)g_i + (0.9 + 0.5\xi) \tag{11}$$

For example, $w_i = -g_i + 1.4$ if $\xi = 1$.

Ratnaweera *et al* in [11] suggested that, with a larger cognitive component and small social component at the beginning, particles are allowed to move around the search space, instead of moving toward the population best. Besides, a small cognitive component and a large social component allow the particles to converge the global optima in the latter part of the search. In this study, a particle with a larger relational grade is regarded as being in the exploitation state. At this situation, a larger c_2 and a smaller c_1 could allow that particle to converge to the global optimum. On the contrary, a particle with a smaller relational grade is assigned the acceleration coefficients of a smaller c_2 and a larger c_1 to help for exploring local optimums and maintaining the diversity of the swarm. To sum up, the larger the grey relational grade g_i is, the larger the coefficient c_2 is and the smaller the coefficient c_1 is.

The relationship between the relational grade and the acceleration coefficients can be also determined by the linear scaling scheme. Assume that the interval $[c_{\min}, c_{\max}]$ is chosen to clamp the coefficient c_2. Then, as for particle i, the relationship between c_{2i} and g_i used in this study is selected as

$$c_{2i} = (1+\xi)(c_{\max} - c_{\min})g_i + [(1+\xi)c_{\min} - \xi c_{\max}]. \tag{12}$$

For example, $c_{2i} = 2g_i + 0.5$ if $\xi = 1$, $c_{\min} = 1.5$ and $c_{\max} = 2.5$. Once c_{2i} is determined, the acceleration coefficient c_{1i} can be obtained by

$$c_{1i} = 4.0 - c_{2i} \tag{13}$$

The above formula is derived from the suggestion of Kennedy and Eberhart in [2], that is, both acceleration coefficients are fixed at the value of 2.0.

As far as the clamping interval of acceleration coefficients is concerned, it can be seen that c_1 is limited in $[4-c_{\max}, 4-c_{\min}]$ while c_2 is bounded in $[c_{\min}, c_{\max}]$. Both grey-based acceleration coefficients therefore have the same interval width. A large interval width could allow use of the wide range of the search space, so as to prevent the premature convergence due to lack of population diversity. Due to this reason, the

interval width should be large at the early part of the evolutionary process. On the other hand, when the algorithm is converging to the optimal solution, fine-tuning of the solutions becomes necessary to find the global optimum efficiently. Therefore a small interval width could enhance convergence toward to the global optimum in the latter part of the search. These concerns give rise to the motivation to propose a time-varying boundary for the acceleration coefficients as follows.

$$c_{\max} = (C_{\text{final}} - C_{\max})\frac{t}{T} + C_{\max}, \tag{14}$$

$$c_{\min} = (C_{\text{final}} - C_{\min})\frac{t}{T} + C_{\min}, \tag{15}$$

where $C_{\max} \geq C_{\text{final}} \geq C_{\min}$, and C_{final} represents the final value of the acceleration coefficient c_2.

This study utilizes the grey relational grade to determine the inertia weight and the acceleration coefficients. During the evolutionary process, the position of each particle may differ for different generations. It is obvious that the relational grade may also differ for different generations. Hence, both inertia weight and acceleration coefficients are varying over the generations. In addition, with the proposed time-varying parameters, the updating rule for the velocity of the ith particle becomes as

$$v_{id} = w_i v_{id} + c_{1i} rand_1(pBest_{id} - x_{id}) + c_{2i} rand_2(gBest_d - x_{id}), \tag{16}$$

where $rand_1$ and $rand_2$ are two uniformly distributed random numbers independently generated within [0, 1].

4 Simulation Results

In order to demonstrate the search performance of the proposed grey-based PSO algorithm, twelve test functions are used to verify it. The selected benchmark functions are listed in Table 1, where the first six functions are unimodal and the rest are multimodal. The corresponding dimensions, search spaces, global optimum values, and acceptance levels of the test functions are also listed in the same table. In this study, the results are compared with the PSO with linearly varying inertia weight (PSO-LVIW) [10], the PSO with linearly time-varying acceleration coefficients (HPSO-TVAC) [11], and the adaptive particle swarm optimization (APSO) [3].

For a fair comparison among all the PSO algorithms, they are tested using the same population size of 20, a value of which is commonly adopted in PSO [3], and the same number of 2×10^5 function evaluations, i.e., 10,000 generations, for each functions. Furthermore, each algorithm was run 30 independent times for each test function. Other parameters setting of the grey-based PSO algorithm are stated as follows: $w_{\max}$ = 0.9, $w_{\min}$ = 0.4, $C_{\max}$ = 2.5, $C_{\min}$ = 1.5, C_{final} = 1.5, and the distinguishing coefficient ξ is 1.0. In addition, the parameters setting for the PSO-LVIW is $w_{\max}$ = 0.9 and $w_{\min}$ = 0.4, while the PSO-TVAC is c_{1i} = 2.5, c_{1f} = 0.5, c_{2i} = 0.5, and c_{2f} = 2.5.

Table 1. Twelve test functions used in this study

Test function	D	Range	Global $f_{\min}$	Acceptance
$f_1(x)=\sum_{i=1}^{D} x_i^2$	30	$[-100, 100]^D$	0	0.01
$f_2(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	30	$[-10, 10]^D$	0	0.01
$f_3(x)=\sum_{i=1}^{D}(\sum_{j=1}^{i} x_i)^2$	30	$[-100, 100]^D$	0	100
$f_4(x)=\sum_{i=1}^{D-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	30	$[-10, 10]^D$	0	100
$f_5(x)=\sum_{i=1}^{D}(\lfloor x_i+0.5\rfloor)^2$	30	$[-100, 100]^D$	0	0
$f_6(x)=\sum_{i=1}^{D} ix_i^4+random[0,1)$	30	$[-1.28, 1.28]^D$	0	0.01
$f_7(x)=-\sum_{i=1}^{D} x_i \sin(\sqrt{\lvert x_i\rvert})$	30	$[-500, 500]^D$	−12,569.5	−10,000
$f_8(x)=\sum_{i=1}^{D}[x_i^2-10\cos(2\pi x_i)+10]$	30	$[-5.12, 5.12]^D$	0	50
$f_9(x)=\sum_{i=1}^{D}[y_i^2-10\cos(2\pi y_i)+10]$ where $y_i=\begin{cases} x_i, & \lvert x_i\rvert<0.5,\\ round(2x_i)/2, & \lvert x_i\rvert\ge 0.5.\end{cases}$	30	$[-5.12, 5.12]^D$	0	50
$f_{10}(x)=-20\exp\left(-0.2\sqrt{\sum_{i=1}^{D} x_i^2/D}\right)-\exp\left(\sum_{i=1}^{D}\cos(2\pi x_i)/D\right)+20+e$	30	$[-32, 32]^D$	0	0.01
$f_{11}(x)=\sum_{i=1}^{D}\frac{x_i^2}{4000}-\prod_{i=1}^{D}\cos(x_i/\sqrt{i})+1$	30	$[-600, 600]^D$	0	0.01
$f_{12}(x)=\frac{\pi}{30}\{10\sin^2(\pi y_1)+\sum_{i=1}^{D}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_D-1)^2\}+\sum_{i=1}^{D} u(x_i,5,100,4)$ where $y_i=1+(x_i+1)/4$ and $u(x_i,a,k,m)=\begin{cases} k(x_i-a)^m, & x_i>a,\\ 0, & -a\le x_i\le a,\\ k(-x_i-a)^m, & x_i<-a.\end{cases}$	30	$[-50, 50]^D$	0	0.05

All the programs coded by Matlab version R14 were executed by a personal computer with Intel Pentium Dual CPU @ 1.60-GHz processor, 2.0-GB RAM and Windows XP2 operating system.

Table 2 lists the numerical results in terms of the mean and standard deviation of the solutions obtained in the 30 independent runs by each algorithm. Boldface in the table indicates the best result(s) among the algorithms. All the PSO algorithms can attain the minimum value of f_5. The main reason is that it is a region rather than a single point in f_5 that is the optimum. For the unimodal test functions, the proposed grey-based PSO algorithm can attain the best accuracy on functions f_1, f_2, f_3, f_5, and f_6, and the second best on f_4, whereas the APSO has the highest accuracy only on

functions f_4 and f_5. On the other hand, the grey-based PSO algorithm also could achieve the best result on the complex multimodal functions f_9, f_{10} and f_{11}, but performs the worst result on function f_{12}. On functions f_7 and f_8, the proposed PSO performs worse than the APSO, but better than the PSO-LVIW and HPSO-TVAV.

Table 3 lists the comparisons on the convergence speed of each PSO variant in terms of the mean number of iterations needed to reach the acceptable solution given in Table 1 and the corresponding mean computational time. The successful ratio of each algorithm for each test function is also given in the same table. The APSO attains the smallest mean number of iterations on nine out of twelve test functions, whereas the grey-based PSO only on two functions. However, a shorter evolutionary process can not directly imply that it uses a lesser computational time. Many existing PSO variants, including the APSO and grey-based PSO, have added extra operations that cost the computational time. In the simulations, the computational time per iteration required by the APSO, in average, is 1.61 times as long as that required by the grey PSO. As seen in Table 3, the HPSO-TVAC uses the least computational time on f_3, f_6, f_7, and f_{12}, whereas the APSO costs the shortest time only on f_5. The grey-based PSO could attain a much smaller computational time on other seven test functions. Besides, the PSO-LVIW generally performs the worst results on convergence speed as well as solution accuracy.

Table 2. Search result comparisons on 12 test functions

		PSO-LVIW	HPSO-TVAC	APSO	Grey-PSO
f_1	Mean	3.16×10^{-52}	9.36×10^{-13}	2.31×10^{-149}	**0**
	Std. Dev.	6.11×10^{-52}	3.62×10^{-12}	4.95×10^{-149}	**0**
f_2	Mean	2.04×10^{-29}	5.33×10^{-6}	4.18×10^{-79}	**0**
	Std. Dev.	4.05×10^{-29}	9.51×10^{-6}	9.96×10^{-79}	**0**
f_3	Mean	1.11×10^{-1}	2.66×10^{-1}	1.71×10^{-10}	$\mathbf{4.37\times10^{-42}}$
	Std. Dev.	1.27×10^{-1}	6.98×10^{-1}	2.87×10^{-10}	$\mathbf{5.91\times10^{-42}}$
f_4	Mean	26.93	63.46	**2.72**	4.89
	Std. Dev.	30.33	31.25	4.03	$\mathbf{7.35\times10^{-2}}$
f_5	Mean	**0**	**0**	**0**	**0**
	Std. Dev.	**0**	**0**	**0**	**0**
f_6	Mean	8.29×10^{-3}	7.50×10^{-3}	5.01×10^{-3}	$\mathbf{4.12\times10^{-3}}$
	Std. Dev.	1.74×10^{-3}	1.86×10^{-2}	$\mathbf{1.19\times10^{-3}}$	3.41×10^{-3}
f_7	Mean	−10243.02	−10316.36	**−11259.9**	−10338.75
	Std. Dev.	205.60	264.32	$\mathbf{2.16\times10^{-11}}$	188.03
f_8	Mean	40.11	37.48	$\mathbf{7.57\times10^{-15}}$	1.29×10^{-5}
	Std. Dev.	8.64	9.67	$\mathbf{1.01\times10^{-14}}$	4.31×10^{-5}
f_9	Mean	33.17	38.20	8.86×10^{-16}	$\mathbf{1.67\times10^{-21}}$
	Std. Dev.	15.34	8.29	3.01×10^{-15}	$\mathbf{4.08\times10^{-21}}$
f_{10}	Mean	1.15×10^{-14}	9.57×10^{-6}	1.11×10^{-14}	$\mathbf{8.88\times10^{-16}}$
	Std. Dev.	3.55×10^{-15}	3.68×10^{-5}	5.65×10^{-15}	**0**
f_{11}	Mean	2.79×10^{-3}	3.29×10^{-3}	2.03×10^{-3}	**0**
	Std. Dev.	4.15×10^{-3}	4.23×10^{-3}	3.94×10^{-3}	**0**
f_{12}	Mean	2.28×10^{-3}	2.76×10^{-2}	$\mathbf{8.24\times10^{-40}}$	2.23×10^{-1}
	Std. Dev.	1.64×10^{-3}	6.15×10^{-2}	$\mathbf{6.18\times10^{-39}}$	5.17×10^{-2}

Table 3. Convergence speed and succesful rate comparisons on 12 test functions

		PSO-LVIW	HPSO-TVAC	APSO	Grey-PSO
f_1	Mean Epochs	5495.4	2327.5	**1065.8**	1575.8
	Time (sec)	0.9119	0.5354	0.5086	**0.4191**
	Ratio (%)	**100.0**	**100.0**	**100.0**	**100.0**
f_2	Mean Epochs	5289.4	2336.8	**1081.0**	1400.9
	Time (sec)	0.9118	0.5683	0.5293	**0.3918**
	Ratio (%)	**100.0**	**100.0**	**100.0**	**100.0**
f_3	Mean Epochs	7470.5	3571.1	**1996.5**	5311.9
	Time (sec)	2.3859	**0.8243**	0.9736	1.4810
	Ratio (%)	**96.67**	93.33	**96.67**	**96.67**
f_4	Mean Epochs	5193.3	2490.1	**1058.3**	1253.6
	Time (sec)	0.9953	0.5208	0.4565	**0.3248**
	Ratio (%)	**100.0**	**100.0**	**100.0**	**100.0**
f_5	Mean Epochs	5649.2	2447.2	**795.3**	1783.4
	Time (sec)	1.0309	0.4535	**0.3421**	0.4072
	Ratio (%)	**100.0**	**100.0**	**100.0**	**100.0**
f_6	Mean Epochs	9253.7	5589.1	4959.0	**4511.8**
	Time (sec)	2.9209	**1.7359**	3.2196	2.8259
	Ratio (%)	83.33	86.67	**93.33**	**93.33**
f_7	Mean Epochs	3279.3	1480.8	**955.8**	6174.3
	Time (sec)	1.0953	**0.4992**	0.5601	2.1694
	Ratio (%)	63.33	66.67	**76.67**	73.33
f_8	Mean Epochs	5885.5	2776.1	**1087.2**	1405.6
	Time (sec)	1.2939	0.6105	0.4908	**0.4856**
	Ratio (%)	96.67	93.33	**100.0**	**100.0**
f_9	Mean Epochs	7715.3	3364.1	**926.2**	1712.8
	Time (sec)	2.2411	0.9206	0.4646	**0.4635**
	Ratio (%)	**100.0**	**100.0**	**100.0**	**100.0**
f_{10}	Mean Epochs	5774.0	2642.7	2846.5	**1783.8**
	Time (sec)	1.2505	0.5511	1.3141	**0.4881**
	Ratio (%)	93.33	90.00	96.67	**100.0**
f_{11}	Mean Epochs	5686.5	2430.9	**1027.1**	1610.5
	Time (sec)	1.5537	0.6483	0.5747	**0.5127**
	Ratio (%)	63.33	56.67	66.67	**100.0**
f_{12}	Mean Epochs	5470.6	**2580.9**	3156.5	6861.5
	Time (sec)	1.7837	**0.9268**	1.8770	2.2309
	Ratio (%)	86.67	86.67	**93.33**	83.33

5 Conclusions

Based on grey relational analysis and linear scaling scheme, this paper proposes a grey-based particle swarm optimization. By this way, each particle has its own algorithm parameters, which may differ in the different generation. Besides, in each generation, the parameters may differ for different particles. The proposed method is applied to solve the optimization problems of twelve test functions for illustration. Simulation results show that the grey-based PSO outperforms the PSO-LVIW, PSO-TVAC and APSO on the solution accuracy and computational time in most of the considered problems.

Acknowledgments. This work was supported by the National Science Council, Taiwan, Republic of China, under Grants NSC 100-2221-E-262-002.

References

1. Kennedy, J., Eberhart, R.C.: A new optimizer using particle swarm theory. In: Proc. 6th Intl. Symp. Micro Machine Human Sci., pp. 39–43. IEEE Press, New York (1995)
2. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proc. 1991 IEEE Neural Netw., pp. 1942–1948. IEEE Press, New York (1995)
3. Zhan, Z.H., Zhan, J.Z., Li, Y., Chung, H.S.H.: Adaptive particle swarm optimization. IEEE Trans. Syst. Man, Cybern., B. 39(6), 1362–1381 (2009)
4. AlRashidi, M.R., El-Hawary, M.E.: A survey of particle swarm optimization applications in electric power systems. IEEE Trans. Evol. Comput. 13(4), 913–918 (2009)
5. Lin, C.J., Hsieh, M.H.: Classification of mental task from EEG data using neural networks based on particle swarm optimization. Neurocomputing 72(4-6), 1121–1130 (2009)
6. Wai, R.J., Lee, J.D., Chuang, K.L.: Real-time PID control strategy for maglev transportation system via particle swarm optimization. IEEE Trans. Ind. Electron. 58(2), 629–646 (2011)
7. Deng, J.L.: Introduction to grey system theory. J. Grey Syst. 1(1), 1–24 (1989)
8. Yeh, M.-F., Chang, C.-T., Leu, M.-S.: Financial distress prediction model via greyART network and grey model. In: Zeng, Z., Wang, J. (eds.) Advances in Neural Network Research and Applications. LNEE, vol. 67, pp. 91–100. Springer, Heidelberg (2010)
9. Yeh, M.F., Leu, M.S.: Grey adaptive growing CMAC network. Appl. Soft Comput. 11(8), 5400–5410 (2011)
10. Shi, Y., Eberhart, R.C.: A modified particle swarm optimizer. In: Proc. IEEE World Congr. Comput. Intell., pp. 69–73. IEEE Press, New York (1998)
11. 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)

Quantum-Behaved Particle Swarm Optimization Algorithm Based on Border Mutation and Chaos for Vehicle Routing Problem

Ya Li[1], Dan Li[2], and Dong Wang[1]

[1] Dept. of Computer, School of Electrical&Information Engineering, Foshan University, Foshan Guangdong 528000, China
[2] Technology&Information Center, XinYang Power Supply Company, Xinyang Henan 464000, China
li-bh@163.com, xyld@tom.com, wdong@fosu.edu.cn

Abstract. A quantum-behaved particle swarm optimization based on border mutation and chaos is proposed for vehicle routing problem(VRP).Based on classical Quantum-Behaved Particle Swarm Optimization algorithm(QPSO), when the algorithm is trapped in local optimum, chaotic search is used for the optimal particles to enhance the optimization ability of the algorithm, avoid getting into local optimum and premature convergence. To those cross-border particles, mutation strategy is used to increase the variety of swarm and strengthen the global search capability. This algorithm is applied to vehicle routing problem to achieve good results.

Keywords: QPSO, Chaos, Border mutation.

1 Introduction

Vehicle Routing Problem is an important content of physical distribution management. Vehicle Routing Problem means if the clients's demands and locations are known,the shortest route or the minimum transportation cost of every client are assigned. VRP is proved to be a NP-complete problem, as the numbers of clients increase, the solutions increase exponentially. So the exact algorithms are not fit for such problem. Currently the use of heuristic algorithms, such as: Particle Swarm Optimization algorithm, genetic algorithm, ant colony optimization algorithm can better solve the problem. Among them, the PSO (Particle Swarm Optimization, PSO) algorithm with parallel processing features, good robustness, ease of implementation, the larger probability to find a global optimal solution of optimization problems, etc, caused widespread concern in the majority of scholars.

Particle Swarm Optimization algorithm is a new kind of overall evolutional algorithm invented by doctor Eberhart and doctor Kennedy in 1995[1].Since it has been proposed, many improvements of PSO version has emerged in order to improve algorithm performance. Shi and Eberhart introduced the inertia weight w into the algorithm in 1998 and suggested the w should be adjusted dynamically to insure

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 63–73, 2012.

accuracy and convergence rate of this algorithm[2]. Many scholars called the evolution equation as Standard Particle Swarm Optimization algorithm. Clerc introduced the shrinkage factor K into evolutionary equation to ensue the convergence of the algorithm[3], while relaxed the speed limit. Gao Ying etc introduced the chaos into the Particle Swarm Optimization algorithm. Making use of the chaos with randomness, ergodicity and extreme sensitivity to initial conditions, not repeated through all the state within a certain range according to its own laws and other characteristics, chaotic search is used for the optimal particles to avoid falling into local optimal solution, the maximum possible to find the global optimum[4]. Quantum-Behaved Particle Swarm Optimization algorithm puted forward by Sun etc in 2004 is a new kind of Particle Swarm Optimization algorithm based on quantum model. The algorithm is becoming a promising algorithm because of the characters of less parameters, more convenient control and more powerful global convergence properties[5].

Quantum-Behaved Particle Swarm Optimization based on Border mutation and Chaos for Vehicle Routing Problem is proposed in this paper based on a real vector encoding mode. According to document [4]and [5],Logistic chaotic sequence is putted into the QPSO algorithm for solving the vehicle routing problem, To those cross-border particles, mutation strategy is used to increase the variety of swarm and strengthen the global search capability. Test show that the algorithm is better to deal with VRP than PSO,QPSO.

2 A Description of the Vehicle Routing Problem

Mathematical description of VRP is as follows[6]: k-known clients and a distribution center, the demand of the i-th client is q_i, m cars depart from the distribution center and finally back to distribution centers. Each car's carrying capacity is Q, $q_i \leq Q$, an optimal service route (less operating costs) is need to find.

d_{ij} means operating cost from point i to point j. (such as time, distance, cost ,etc).The distribution center's number is 0, each client's number is i(i = 1, 2, ···k),each car's number is s(s= 1, 2, ···m).some variables are defined as follows:

$$x_{ijs} = \begin{cases} 1 & \text{car s go from point i to point j ;} \\ 0 & \text{else ;} \end{cases} \tag{1}$$

$$y_{is} = \begin{cases} 1 & \text{car s service for client i ;} \\ 0 & \text{else ;} \end{cases} \tag{2}$$

The goal to make the model is to find the least total operating cost. Operating cost is proportional to the routes of cars. The shorter the route of car, the less fuel consumption of cars, the less theworking hour of drivers, and thus the less total

operating cost. Therefore, the mathematical model of VRP for finding the shortest total routes of cars as objective function is built as follows:

$$\min Z = \sum_{i=0}^{k}\sum_{j=0}^{k}\sum_{s=1}^{m} d_{ij}x_{ijs} \tag{3}$$

$$\sum_{i=0}^{k} q_i y_{is} \le Q, \qquad s = 1,2,\cdots,m; \tag{4}$$

$$\sum_{s=1}^{m} y_{is} = 1, \qquad i = 1,2,\cdots,k; \tag{5}$$

$$\sum_{i=0}^{k} x_{ijs} = y_{is,} \qquad j = 1,2,\cdots,k; \quad s = 1,2,\cdots,m \tag{6}$$

$$\sum_{j=0}^{k} x_{ijs} = y_{is,} \qquad i = 1,2,\cdots,k; \quad s = 1,2,\cdots,m \tag{7}$$

Formula(4)shows the maximum carrying capacity limit of cars, it means each car's actual carrying capacity must not exceed the car's maximum carrying capacity. Formula(5)ensure each client's task must be finished by one car. Formula(6)and Formula(7)means cars can service each client only once, if a car ascertains its service scope, the car should pass each client's position in its service scope and pass only once.

As can be seen from the model, the optimization problem contains inequality constraints, which is not conducive to find the solution by heuristic search. Penalty term is added to the object function, the modified objective function is:

$$\min Z = \sum_{i=0}^{k}\sum_{j=0}^{k}\sum_{s=1}^{m} d_{ij}x_{ijs} + R\cdot\sum_{s=1}^{m}\max(\sum_{i=1}^{k} q_i y_{is} - Q) \tag{8}$$

In formula(8)penalty coefficient R is a very large positive number. All the second part of formula(8)is the punitive part if the car's actual carrying capacity exceed the maximum carrying capacity. If the solution is not feasible to the VRP, a great fitness solution will be calculated, which make the search process converges to a feasible solution.

3 Quantum-Behaved Particle Swarm Optimization Algorithm

Based on Clerc's research production about the convergence behavior of particles[7],Sun et presented a new Particle Swarm Optimization algorithm based on quantum mechanics. The model is based on DELTA potential well, and considers that the particles have the behavior of quantum particles, according to this model, Quantum-behaved Particle Swarm Optimization is proposed[8]. In the quantum space,

the nature of the particles to meet the aggregate state is completely different, They can search in the whole search space, therefore, The performance of QPSO global search is far better than the standard PSO algorithm. In the quantum space, the velocity and position of the particles can not be determined at the same time, Therefore, the wave function ψ (x, t) (its physical meaning: the square of the wave function is the probability density which a particle appears at a point in space) is used to describe the state of the particle, and particles's probability density function in space of a point is obtained by solving the Schrodinger equation. Then the equation of the particle's position gotten by Monte Carlo random simulation method is:

$$x(t) = P \pm \frac{L}{2} In\left(1/u\right) \tag{9}$$

Parameter u is random number between 0 and 1.

$$L(t+1) = 2 \cdot \beta \cdot \left| mbest - x(t) \right| \tag{10}$$

The resulting evolution equation of QPSO algorithm:

$$mbest = \sum_{i=1}^{M} P_{i1}/M = \left(\sum_{i=1}^{M} P_{i1}/M, \sum_{i=1}^{M} P_{i2}/M, \cdots \sum_{i=1}^{M} P_{iD}/M \right) \tag{11}$$

$$p_{id=} \phi * p_{id} + (1-\phi) * p_{gd} \tag{12}$$

Finally, the position equation of the particle is:

$$x_{id}(t+1) = p_{id} \pm \beta \cdot \left| mbest_d - x_{id}(t) \right| \cdot In\left(1/u\right) \tag{13}$$

Among them, β is the contraction coefficient of expansion, which is the only parameter of QPSO algorithm ,generally take β = 1.0-iter / $iter_{max}$ * 0.5, iter is the current number of iterations, $iter_{max}$ represents the maximum number of iterations. M is the number of particles in population. D is the particle dimension. ϕ is random numbers uniformly distributed in [0, 1]. Mbest is the average position of all particles's best position. The same with PSO, P_i means the best position experienced by particle i, P_g means the best position experienced by all particles.

4 Quantum-Behaved Particle Swarm Optimization Algorithm Based on Border Mutation and Chaos

4.1 Chaotic Motion

Chaos is a general phenomenon in the non-linear systems given by deterministic equations with random state of motion. The change of chaotic variable within a certain range is random, ergodicity and regularity, chaotic variable use of these

features to refine search, make the algorithm out of local optima, to maintain population diversity, improve the global search performance. There is a lot of rules about chaos, the most classic is the logistic model [9]:

$$y_{n+1} = 4y_n(1-y_n), n = 1,2,3,\cdots \tag{14}$$

Where y_n is the chaotic variable, $y_n \in [0,1]$.

In the running process of Quantum particle swarm optimization algorithm, if a particle found a current optimal position, other particles will move closer to him quickly. If the position is local minima, the algorithm will fall into local optimum, so-called premature convergence phenomenon. Experiments show that when the quantum particle swarm optimization is premature convergence or global convergence, particles in group will be a "gathering" phenomenon. Either all particles gathered at a particular location, or gathered in a few specific locations. According to document[10], average particle distance of particles and fitness variance of particles are the standards to judge whether the algorithm going into the premature convergence.

In the improved algorithm, when the algorithm is the premature convergence, using the ergodicity of chaos, chaotic search is used for the optimal solution p_g ,the method is[11]:

p_g is mapped to the Logistic equation to the definition of the domain [0,1]:

$$y_1^k = \frac{p_g^k - x_{\min}^k}{x_{\max}^k - x_{\min}^k} \tag{15}$$

y_1^k is put into Logistic equation for T iterations, $y_{n+1}^k = 4y_n^k(1-y_n^k), n = 1,2,3,\cdots$ the chaotic sequence $y^k = \left(y_1^k, y_2^k, \cdots y_T^k\right)$ has been gotten.

The chaotic sequence is inverse mapped back to the original solution space by the following equation:

$$p_{g,m}^{*k} = x_{\min}^k + \left(x_{\max}^k - x_{\min}^k\right)y_m^k, m = 1,2,\cdots T \tag{16}$$

Compute each fitness value of each feasible solution vector in feasible solution sequence, and retain the feasible solution vector corresponding to the best fitness value, denoted by p_g^{*k}. Randomly select a particle from the current particle swarm, replace the selected particle's position vector with p_g^{*k}.

4.2 Boundary Mutation Strategy

In QPSO algorithm, when the particles fly over the border of the search field, the value of the boundary position is usually given to the particle:

$$if \quad x_{id} > x_{\max} \quad x_{id=} x_{\max} \tag{17}$$

Or

$$if \quad x_{id} < x_{\min} \quad x_{id=} x_{\min} \tag{18}$$

After this treatment, all the cross-border particles are gathered at the boundary, if there is local optimum, the particle is easy to fall into the local minima, and thus can not find the true optimal solution; the same time, with the boundary particles increases, the diversity of species will be reduced, the ability of global search of the algorithm will certainly be affected. Improved algorithm presented in this paper made the following improvements for the cross-border particles.

$$if \quad x_{id} > x_{\max} \quad x_{id=} x_{\max} - c \times rand() \tag{19}$$

or

$$if \quad x_{id} < x_{\min} \quad x_{id=} x_{\min} + c \times rand() \tag{20}$$

Among them, $c = x_{\max} - x_{\min}$ .As can be seen from the above process, after the mutation operation on the cross-border particles, the particles will not gathered at the boundary, they will be re-distributed in the feasible space. Through this operation, the particles are in the feasible space, the shortcoming of original algorithm which is easy to fall into the border local optimum is overcomed, and the diversity of the population is increased, the algorithm's global search capability is also improved.

4.3 Encoding and Decoding

For the vehicle routing problem, this paper presents a encoding method based on real vector. This method without increasing the dimension, represent the car and the client's delivery order in the encoding. For the VRP problem with k clients, m cars, the state of the particle is represented with k-dimensional real vectors. For each dimension of the vector, the integer part means the car service for the clients, the same integer part means the same car service for the clients. Fraction means the delivery order of the clients serviced by the same car.

Encoding and decoding process is as follows:

1) K arbitrary real vectors generated randomly between 1 and m+1 represent the states of the particles.
2) The integer parts of k real vectors are taken, if the integer part is same, they will be putted into the same group.
3) In the same group, the client's delivery order is formed according to the order of the fractional part of the real vectors.

For example, assume there are eight clients, two cars, the codes are as follows:

Client: 1 2 3 4 5 6 7 8
X: 1.12 1.40 2.83 2.19 2.74 1.35 2.50 2.71

According to the decoding method above, the x is rounded, the same integral parts are divided into the same group, the following two groups are obtained:

(1.12, 1.40, 1.35), (2.83, 2.19, 2.74, 2.50, 2.71),

Then, within each group, according to the fractional part of X arranged from small to large, results are as follows: (1.12,1.35,1.40), (2.19,2.50,2.71,2.74,2.83),the above states will be mapped to the corresponding clients, which is the distribution plan of this set of codes:

The first car line: 0-1-6-2-0
The second car line: 0-4-7-8-5-3-0

Using this encoding method, the particle dimensions are same with the number of clients, ordering and rounding operation are executed only once when decoding, when the particle status need to update, re-adjust the state of each particle is more convenient, which can save the calculation time for large-scale problems. And the encoding method can use the rules of the standard particle swarm algorithm, so the inherent characteristics of the particle swarm algorithm can be played.

4.4 Specific Processes of Algorithm

(1) Particle Swarm is initialized. Parameters are initialized, the maximum number of the iterations $iter_{\max}$, the number of particles N, chaos search iterations T are set. Arbitrary real vectors of k randomly generated between 1 and m+1 represent the states of the particles.
(2) The distribution plan is got in accordance with Section 4.3 which provides encoding and decoding methods.
(3) The fitness values of the particles are calculated, and the personal best value p_i and the global optimum p_g are updated.
(4) If the stop condition (usually the default computing precision or the number of iterations)is reached, the search is stopped, output the result, or else go to step 5.
(5) According to the average distance of particles and fitness variance of particles to judge whether the algorithm go into premature. If the algorithm goes into the premature and go to step (7), otherwise go to step (6).
(6) The next positions of particles are calculated in accordance with formula (11) (12) (13). If the particles have crossed the border, the particle position should be modified in accordance with the contents of section 4.2. Go to step (3).
(7) The optimal particle is selected for chaotic optimization according to the contents of section 4.1. Go to step (3).

5 Simulation and Analysis

The experimental environment: Pentium (R) Dual-Core 3. 00GHz CPU, 2G RAM, Windows7 OS, Microsoft VC++6.0 programming software.

Test instances in document [12] is used, clients are known to 8 points and a distribution center, each client's demand q = [1 2 1 2 1 4 2 2] (in tonnes). There are two cars in distribution center to service, each car's maximum load is 8 tons, the distance (in km) between each client and distribution center are known as shown in Table 1 (The distribution center's number is 0):

Table 1. The distance between each client and distribution center

d_{ij}	0	1	2	3	4	5	6	7	8
0	0.0	4.0	6.0	7.5	9.0	20.0	10.0	16.0	8.0
1	4.0	0.0	6.5	4.0	10.0	5.0	7.5	11.0	10.0
2	6.0	6.5	0.0	7.5	10.0	10.0	7.5	7.5	7.5
3	7.5	4.0	7.5	0.0	10.0	5.0	9.0	9.0	15.0
4	9.0	10.0	10.0	10.0	0.0	10.0	7.5	7.5	10.0
5	20.0	5.0	10.0	5.0	10.0	0.0	7.0	9.0	7.5
6	10.0	7.5	7.5	9.0	7.5	7.0	0.0	7.0	10.0
7	16.0	11.0	7.5	9.0	7.5	9.0	7.0	0.0	10.0
8	8.0	10.0	7.5	15.0	10.0	7.5	10.0	10.0	0.0

To illustrate the effectiveness of the algorithm proposed in this paper, six kinds of algorithms were used to solve the VRP problem ,which are PSO algorithm, QPSO algorithm, SPSO algorithm, GWPSO algorithm, GCPSO algorithm and the algorithm proposed in this paper, known the optimal solution is 67.5 (km), the optimal solution line is:

The first car line: 0-4-7-6-0
The second car line: 0-1-3-5-8-2-0

The number of particles of six kinds of algorithms N is 20; the maximum number of the iterations itermax is 500; In PSO,SPSO algorithms, w=1,c1=c2=1.4;In GCPSO algorithms, c1=c2=2.05;In GWPSO algorithms, c1=c2=1.4,wmax=0.9,wmin=0.4; the number of the chaotic iterations T of the algorithm proposed in this paper is 20.

The experimental results for 20 times by six algorithms according to the above parameters shown in Table 2:

Table 2. The optimal solution values for 20 times experiments by six algorithms

number of experiment	PSO /km	SPSO /km	GWPSO /km	GCPSO /km	QPSO /km	The proposed algorithm /km
1	70.0	72.5	73.0	75.5	70.0	70.0
2	71.0	75.0	69.5	72.5	71.0	70.0
3	77.0	76.0	71.5	71.0	77.0	67.5
4	71.0	84.5	71.5	70.0	71.0	70.0
5	73.5	73.0	69.0	70.0	73.5	71.5
6	72.0	76.0	71.5	77.5	72.0	71.0
7	73.0	78.0	70.0	69.0	73.0	72.5
8	72.5	77.0	70.0	73.5	72.5	70.0
9	71.5	78.0	70.0	78.0	71.5	69.0
10	72.0	78.0	69.5	70.0	72.0	69.5
11	67.5	75.0	70.0	72.5	67.5	70.0
12	75.5	70.0	72.5	70.0	75.5	70.0
13	70.0	74.5	73.0	75.0	70.0	69.5
14	74.5	83.5	72.5	73.5	74.5	70.0
15	69.0	73.0	70.0	69.0	69.0	67.5
16	73.0	74.5	74.0	71.5	73.0	69.0
17	70.0	72.0	71.0	70.0	70.0	71.0
18	70.0	72.0	71.5	74.0	70.0	67.5
19	69.0	70.0	69.0	74.0	69.0	70.0
20	70.0	71.0	71.5	69.0	70.0	69.0

The comparison of experimental results of six algorithms shown in Table 3:

Table 3. The comparison of experimental results of six algorithms

algorithms	The average solution/km	The standard deviation of the solution	Optimal solution/km	The number of Optimal solution
PSO	71.35	2.38	67.5	1
SPSO	75.18	3.83	70.0	0
GWPSO	71.02	1.42	69.0	0
GCPSO	72.28	2.71	69.0	0
QPSO	71.60	2.29	67.5	1
The proposed algorithm	69.73	1.25	67.5	3

As can be seen from table 3, the proposed algorithm is more accuracy, better stability than other five algorithms.

6 Conclusion

Optimized for the vehicle routing problem, this paper presents an improved quantum particle swarm optimization algorithm. The proposed algorithm based on the QPSO algorithm uses Real vector-based encoding, when the algorithm is trapped in local minima, Chaotic search is used on the best particle to enhance the ability of the algorithm optimization and avoid getting into local optimum and premature convergence. At the same time for cross-border particles, the mutation strategy is used to improve the population diversity, enhance the global search capability. Tests show that the algorithm is the best to deal with VRP Compared with other five algorithms.

Acknowledgments. This paper is supported by Natural Science Foundation of Guangdong Province (10152800001000029) and Guangdong Province scientific and technological project (2011B010200031).

References

1. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: 4th IEEE International Conference on Neural Networks, pp. 1942–1948. IEEE Press, Australia (1995)
2. Shi, Y., Eberhart, R.C.: A modified particle swarm optimizer. In: The IEEE International Conference on Evolutionary Computation, pp. 69–73. IEEE Press, Piscataway (1998)

3. Clerc, M.: The swarm and the queen towards a deterministic and adaptive particle swarm optimization. In: The Congress on Evolutionary Computation, pp. 1951–1957. IEEE Press, Piscataway (1999)
4. Gao, Y., Xie, S.L.: Chaos Particle Swarm Optimization Algorithm. J. Computer Science 31(8), 13–15 (2004)
5. Sun, J., Xu, W.B., Feng, B.: A global search strategy of quantum-behaved particle swarm optimization. In: The IEEE Conference on Cybernetics and Intelligent Systems, pp. 111–116. IEEE Press, Piscataway (2004)
6. Li, N., Zhou, T., Sun, D.B.: Particle swarm optimization for vehicle routing problem. J. Systems Engineering 19(6), 597 (2004)
7. Clerc, M., Kennedy, J.: The particle swarm: Explosion, stability and convergence in a multi-dimensional complex space. J. IEEE Transactions on Evolutionary Computation 6(1), 58–73 (2002)
8. Sun, J., Feng, B., Xu, W.B.: Particle swarm optimization with particles having quantum behavior. In: The Congress on Evolutionary Computation, pp. 325–331. IEEE Press, Portland (2004)
9. Gao, S., Yang, J.Y.: Research on Chaos Particle Swarm Optimization Algorithm. J. Pattern Recognition and Artificial Intelligence. 19(2), 266–270 (2006)
10. Duan, X.D., Gao, H.X., Zhang, X.D., Liu, X.D.: Relations between Population Structure and Population Diversity of Particle Swarm Optimization Algorithm. J. Computer Science. 34(11), 164–166 (2007)
11. Meng, H.J., Zhen, P., Mei, G.H., Xie, Z.: Particle Swarm Optimization Technical report of Zhejiang University of Technology (2007)

An Improved MOPSO with a Crowding Distance Based External Archive Maintenance Strategy*

Wei-xing Li[1], Qian Zhou[1], Yu Zhu[2], and Feng Pan[1,**]

[1] School of Automation, Beijing Institute of Technology (BIT), 5 South Zhongguancun Street,Haidian District BeiJing ,100081 P.R. China
[2] China Academy of Space Technology
andropan@gmail.com

Abstract. For multi-objective optimization algorithms, the maintenance policy of external archive has a great impact on the performance of convergence and solution diversity. Considering the dilemma of large population and external archive, an improved strategy of external archive maintenance based on crowding distance is proposed, which requires less particle numbers and smaller archive size, resulting in the computation cost reduction. Furthermore, the information entropy of *gbest* is analyzed to emphasize the diversity improvement of non-dominant solutions and well-distribution on the Pareto-optimal front. Numerical experiments of benchmark functions demonstrate the effectiveness and efficiency of proposed multi-objective particle swarm optimization.

Keywords: Multi-objective optimization, Particle Swam Optimizer, Pareto-optimal front, information entropy.

1 Introduction

The multi-objective optimization problem (*MOP*) is a class widespread optimization problem with several hard-solving characteristics, such as high-dimensional, discontinuous, non-convex, multimodal, and/or NP-complete, etc. Different from those deterministic methods, swarm intelligence optimization techniques offer a series of efficient algorithms, e.g. particle swarm optimization [1] (*PSO*), genetic algorithm (*GA*) and evolutionary algorithm (*EA*), etc.

The information sharing mechanism in *PSO* is significantly different with *GA* and *EA*. Hu and Eberhart [2] designed a dynamic neighborhood *MOPSO*. At each iteration, a particle finds the nearest particles as neighbors, based on the distance which is calculated in the fitness space of the first fixed objective function. The dynamic neighborhood encourages the information sharing of *pbest* and not concentrates on a single *gbest* or *pbest*. As a further study, combined with the

* Supported by: National Natural Science Foundation (60903005).
** Corresponding author.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 74–82, 2012.

dynamic neighborhood strategy, Hu [3] introduce an external repository, so as to improve particles uniformly distributed along the Pareto frontier and enhance the diversity.

V. L. Huang [14] etc. presents an approach to incorporate Pareto dominance into the differential evolution (DE) algorithm in order to solve optimization problems with more than one objective by using the DE algorithm. This algorithm uses an external archive to store non-dominated solutions. In this paper, authors propose a new harmonic average distance to measure the crowding degree of the solutions more accurately.

Parsopoulous and Vrahatis apply each object in the update equations of particle's velocity for double objective optimization problems in [4] and [5]. *CMOPSO*, one of the earliest methods to use external archive to store non-dominant solutions, adopt grid method in [6] and [7], which partitions the objective space to several hypercube. In [8], Comprehensive Learning *PSO* (*CLPSO*) with external archive, making use of other particle's position to update personal velocity, is applied to solve MOP. Li [9], adopting the method of *NSGA-II* to maintain external archive, proposes a non-dominated sort PSO (*NSPSO*). A modified *MOSPO* [10] updates global optimum based on descending order of the particle's crowding distance, moreover brings in small probability variation mechanism to enhance the global optimization ability and to control the number of Pareto-optimal solutions. The crowding distance based maintenance strategy is also applied to the dynamic *MOPSO* [11] and *NSGA-II* [12].

In this paper, an improved crowding distance for external archive maintenance is discussed in the 3-rd section. Furthermore, the information entropy is introduced to evaluate the *gbest* archive and swarm diversity in the 4-th section. Four benchmark functions which are *ZDT*1, *ZDT*2, *ZDT*3, and *ZDT*6 are tested in the numerical experiments section.

2 Strategy of External Archive Maintenance

A multi-objective problem with n decision variable and m sub-objectives is described as follows:

$$\begin{aligned} \min/\max:\ & y=\boldsymbol{F}(\boldsymbol{x})=(f_1(\boldsymbol{x}),f_2(\boldsymbol{x}),f_3(\boldsymbol{x})\ldots f_m(\boldsymbol{x})),\boldsymbol{x}\in[\boldsymbol{x}_{min},\boldsymbol{x}_{max}] \\ s.t.\ & \begin{cases} g_i(\boldsymbol{x})\le 0 & i=1,2,3,\ldots,q \\ h_j(\boldsymbol{x})=0 & i=1,2,3,\ldots,p \end{cases} \end{aligned} \tag{1}$$

Where, x is n dimensions decision space. y is m dimensions objective space. $F(x)$ defines m mapping functions which is from decision space to objective space.

In the case of a *MOPSO* algorithm, it is required to guarantee the solutions which are not only the Pareto optimal, but are also uniformly distributed in the Pareto front. Especially, the *gbest* have to be selected form several non-dominated *pbest* set; and the relationship between *pbest* and the current particles need to differentiate and manage.

2.1 General Framework for External Archive Maintenance

At each iteration, the non-dominant solutions are stored in the external archive. General operations includes: removing the dominated members of external archive; adding the non-dominant solutions into external archive; ranking the non-dominant solutions according to a certain objective value; removing several non-dominant solutions with some maintenance policy when the external archive size exceeds, e.g. crowding distance, diversity, etc.

2.2 External Archive Maintenance Improvement

According to the abovementioned framework, nothing will be operated when the size of external archive is less than the defined size. It puts up a question that: if the diversity of non-dominant solutions in the external archive is poor, it will result in particles, following the "direct" of these non-dominant solutions, "flock" together around a certain region of the Pareto-optimal front.

The non-dominant solutions in the external archive at some time distributes as Fig. 1 shown. Round dot corresponds to non-dominant solution. Obviously, non-dominant solutions *B*, *C*, and *D* are quite close and have more selected probability if the algorithm chooses *gbest* stochastically. As time wore on, the diversity will become worse.

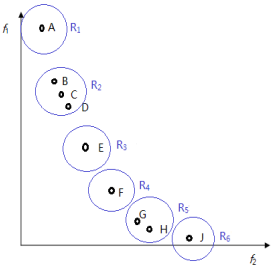


Fig. 1. The distribution of the non-dominant solutions at certain time

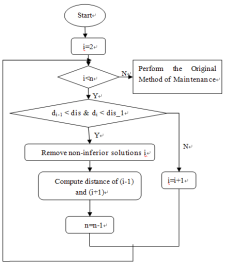


Fig. 2. The flow chat of the improved strategy of external archive maintenance

In view of the above, an improvement is descripted as follows: In each iteration, no matter the size of external archive outnumber the defined size, the distances between each non-dominant solution and adjacent non-dominant solution are calculated first, and then compared with a metric d_{is}, defined in Eq.(2); those non-dominant solution whose distances are all less than d_{is} are removed and the distances matrix are updated again. Where, n is the number of non-dominant solutions, f_i is the (i)th non-dominant solution. Fig. 2 shows the flow chat of the improved strategy of external archive maintenance.

$$d_{is} = \frac{\sum_{i=1}^{n-1} \| f_i - f_{i+1} \|}{n+1} \tag{2}$$

3 Analysis of Information Entropy and Multi-objective Particle Swarm Optimization

3.1 Analysis of *gbest*'s Information Entropy

The information entropy [13], defined as the information probabilistic measurement in Eq.(3), is introduced to analyze the proposed improvement.

$$H(x) = E(\log_2 \frac{1}{p_i}) = -\sum_{i=1}^{k} p_i \ln p_i \tag{3}$$

Supposed X is a certain event, and the result has various probabilities, recording as x_1, x_2, … , x_k. And its homologous probability is p_1, p_2, …, p_k. $H(x)$ is the information entropy of x. It means the average information content of n stochastic information. The information entropy increase implies a wider choice of *gbest* that benefits the Pareto optimal front distribution.

It is assumed that the size of external archive is less then a setting value in the original *gbest* maintenance scenario, as Fig.1 shown. The non-dominant solutions are classified into several *R-region* (R_1, R_2,…R_N) with a distance R. With the roulette wheel selection method for *gbest*, the non-dominant solutions in the external archive are chosen by probability equally. So the selected probability of every *R-region* depends on the number of non-dominant solutions in the area. The state space is defined as $S=\{R_1, R_2\ldots, R_N\}$, and the corresponding probability $p(R_i)$ is calculated as following:

$$p(R_i) = \frac{m_i}{N}, i = 1, 2, ..., N \tag{4}$$

Where, m_i is the number of non-dominant solutions in R_i-*region*. N is the total number of the non-dominant solutions of external archive. Then, the entropy of X can be represented:

$$H(x) = -\sum_{i=1}^{k} p_i \log_2 p_i = -\sum_{i=1}^{k} \frac{m_i}{N} \log_2 \frac{m_i}{N} \tag{5}$$

For the original *gbest* maintenance scenario, there is no operation when the external archive size is less than the defined size, moreover, the number of non-dominant solutions in every R_i-*region* are quiet different. According to the maximum entropy property, information entropy achieve the maximum value, only if X is equal to $\{R1, R2, \ldots, R_k\}$ for equal probability. That is:

$$H(x) = -\sum_{i=1}^{k} \frac{m_i}{N} \log_2 \frac{m_i}{N} \le -\sum_{i=1}^{k} \frac{1}{k} \log_2 \frac{1}{k} = \log_2 k \tag{6}$$

The decrease of information entropy reduces the *gbest* diversity, so that it will attenuate the explosion ability and the swarm may entrap in local region. Yet the strategy in section 2.2 sets up *gbest* located in each *R-region* with equal probability. Furthermore, according to the same analysis and calculation procedure as above, the maximum of entropy will be obtained, which provides a more homogenic selection mechanism for non-dominant solutions.

An example is taken to explain the information entropy analysis. In the scenario (with 9 non-dominant solutions) plotted in Fig. 1, the entropy of X with the original maintenance method could be calculated as following.

$$H_1(x) = -\sum_{i=1}^{n} \frac{m_i}{N_1} \log_2 \frac{m_i}{N_1} = -(\frac{1}{9} \cdot \log_2 \frac{1}{9} + \frac{3}{9} \cdot \log_2 \frac{2}{9} + \frac{1}{9} \cdot \log_2 \frac{1}{9} + \frac{1}{9} \cdot \log_2 \frac{1}{9} + \frac{2}{9} \cdot \log_2 \frac{2}{9} + \frac{1}{9} \cdot \log_2 \frac{1}{9}) = 2.4194$$

As another scenario plotted in Fig. 3, the improved strategy of maintenance is applied. C and G are removed which means the number of non-dominant solution is 7 (N_2=7), and *R-region* is reclassified that R_2-*region* is divided into R_{21}-*region* and R_{22}-*region*. The entropy of X is:

$$H_2(x) = -\sum_{i=1}^{n} \frac{m_i}{N_2} \log_2 \frac{m_i}{N_2} = -\sum_{i=1}^{n} \frac{1}{7} \log_2 \frac{1}{7} = 2.8074$$

Compared $H_1(X)$ and $H_2(X)$, it is obviously that $H_1(X) < H_2(X)$ and indicates that the potential *gbest* could be selected more widely. It is in favor of the even distribution of non-dominant solutions in the Pareto-optimal front and promotes the optimizing process.

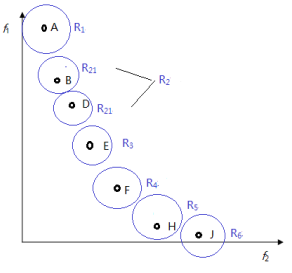


Fig. 3. Improved maintenance

3.2 Procedure of *DISMOPSO*

Combined *MOPSO* algorithm with the proposed maintenance method, a Distance based *MOPSO* (*DISMOPSO*) is proposed, the procedure is described as follows:

Step1 Initialize a swarm.
Step2 Evaluate all particles, and add new non-dominant solutions in the eternal archive.
Step3 Maintain the external archive with the proposed method.
Step4 Select *gbest* and *pbest* for each particle.
Step5 Update velocity and position for each particle.

$$\begin{cases} v_{id}^{(k+1)} = \omega \cdot v_{id}^{(k)} + c_1 \cdot r_{gd} \cdot (p_{gd}^{(k)} - x_{id}^{(k)}) + c_2 \cdot r_{id} \cdot (p_{id}^{(k)} - x_{id}^{(k)}) \\ x_{id}^{(k+1)} = x_{id}^{(k)} + \eta \cdot v_{id}^{(k+1)} \end{cases} \tag{7}$$

Step6 If the termination conditions are achieved, output the result, otherwise go to Step2.

4 Numerical Experiments and Discussion

4.1 Experiments Design

Four multi-objective benchmark functions (*ZDT*1, *ZDT*2, *ZDT*3 and *ZDT*6) are examined. Every function has two objects and the variable is 30 dimensions. Experiments results of *DISMOPSO* are compared with *CMOPSO* [7][8] and *MOCLPSO* [9]. The parameter settings of the three algorithms are listed in Table 1.

Table 1. Settings of Parameter

Parameter	CMOPSO	MOCLPSO	DISMOPSO
w	0.4	0.4	0.4
c_1,c_2	2	2	2
V_{max}	0.5	0.5	0.5
Swarm Size	20	20	20
Dimension	30	30	30
External Archive Size	20	20	20
Max iteration	2000	2000	2000
Grid Size	30	——	——
Study Probability P_l	——	0.1	——
Elite probability P_e	——	0.4	——

Generation distance (*GD*) in Eq.(8) and diversity index (△) in Eq.(9) are adopted to evaluate the optimization performance.

$$GD = \frac{\sqrt{\sum_{i=1}^{n} dist_i^2}}{n} \tag{8}$$

$$\Delta = \frac{h_f + h_l + \sum_{i=1}^{n-1} | h_i - h' |}{h_f + h_l + (n-1)h'} \tag{9}$$

Where, $dist_i$ is the shortest distance between the i-th non-dominant solution and the real Pareto-optimal front; h_i is the distance between adjacent two points; h' is the mean of h_i; h_f and h_l is the distance between boundary solution of algorithm and its corresponding extreme solution respectively.

4.2 Experimental Result and Data Analysis

The four benchmark function results of three algorithms are plotted in Fig.4 to Fig.7.

It is illustrated in Fig.4 and Fig.5 the advantage of *DISMOPSO*. When the size of population and external archive are both 20, the solutions of *DISMOPSO* approximate the Pareto-optimal front is best, and they distribute evenly on the Pareto-optimal front which can approximate the whole of Pareto-optimal front.

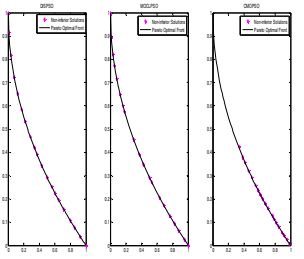


Fig. 4. The testing of *ZDT*1

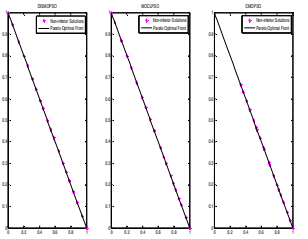


Fig. 5. The testing of *ZDT*2

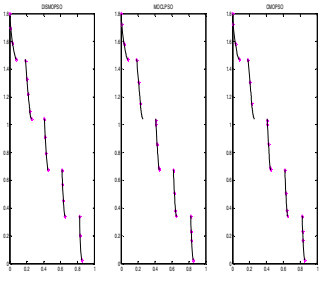


Fig. 6. The testing of ZDT3

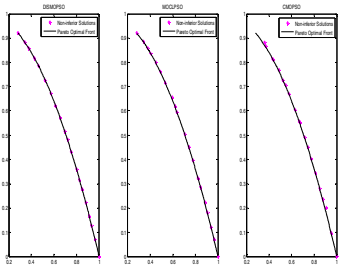


Fig. 7. The testing of *ZDT*6

Table 2. Results of three algorithms

Algorithm		DISMOPSO		CMOPSO-1		MOCLPSO-2	
		GD	△	GD	△	GD	△
ZDT1	Min	1.89×10^{-4}	0.119	1.36×10^{-4}	0.625	1.24×10^{-4}	0.244
	Max	9.57×10^{-4}	0.217	4.83×10^{-4}	0.753	6.77×10^{-4}	0.458
	Mean	4.00×10^{-4}	0.167	2.28×10^{-4}	0.677	2.63×10^{-4}	0.325
	Std.	2.08×10^{-4}	0.026	9.57×10^{-5}	0.038	1.43×10^{-4}	0.053
ZDT2	Min	1.44×10^{-4}	0.123	1.74×10^{-4}	0.505	1.27×10^{-4}	0.222
	Max	7.68×10^{-4}	0.188	2.20×10^{-4}	0.65	2.22×10^{-4}	0.544
	Mean	3.40×10^{-4}	0.157	1.92×10^{-4}	0.578	1.74×10^{-4}	0.337
	Std.	1.89×10^{-4}	0.022	1.26×10^{-5}	0.039	2.35×10^{-4}	0.077
ZDT3	Min	2.34×10^{-4}	0.123	2.16×10^{-4}	0.382	2.21×10-4	0.175
	Max	3.26×10^{-4}	0.244	5.80×10^{-4}	0.699	5.85×10^{-4}	0.436
	Mean	2.76×10^{-4}	0.164	3.44×10^{-4}	0.536	3.09×10^{-4}	0.292
	Std.	2.91×10^{-4}	0.033	1.26×10^{-4}	0.09	9.32×10^{-5}	0.062
ZDT6	Min	0.10×10^{-3}	0.101	0.20×10^{-3}	0.254	0.20×10^{-3}	0.163
	Max	0.83×10^{-2}	0.434	0.30×10^{-1}	0.595	0.15×10^{-1}	0.563
	Mean	0.17×10^{-2}	0.295	0.40×10^{-2}	0.402	0.24×10^{-2}	0.318
	Std.	0.59×10^{-2}	0.089	0.74×10^{-2}	0.087	0.44×10^{-2}	0.101

The solutions of *MOCLPSO* can also reach the Pareto optimal front, but distribute partial area in the Pareto optimal front and the diversity of the solution is not good. As for *CMOPSO*, the solutions can only reflect a part of Pareto-optimal front. Moreover the distribution of them is not homogeneous. Figure 6 and figure 7 present the same result. According to these figures, the property of *DISMOPSO* is the best, and *CMOPSO* is the worst.

The metric *GD* and $\triangle$, averaged by 20 independent experiments, are analyzed in Table 2, All *GD* values of three algorithms are very small, except the results of *ZDT*6. It suggests that almost all the three algorithms can approximate the Pareto-optimal front with high precision. The result of $\triangle$ of *DISMOPSO* is the best among the three algorithms, which indicates a good diversity performance is maintained by *DISMOPSO*.

5 Conclusions

In this paper, the idea of crowding distance of *NSGA-II* is referred. With that, an improved maintenance strategy for external archive is proposed. When the size of external archive is less than the setting value, the maintenance can also take effect. Furthermore, the information entropy of *gbest* is analyzed, which indicate the improved maintenance method increases the information entropy so that the algorithm can select *gbest* more widely and benefit the even distribution of non-dominant solutions in the Pareto-optimal front. The result of experiment also proves that the improved algorithm is effective with smaller size of population and eternal archive. In the future work, the performance of *DISMOPSO* is expected to be improved and it will be applied to those multi-objective benchmark function with constrains.

References

1. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, pp. 1942–1948 (1995)
2. Hu, X., Eberhart, R.C.: Multi-objective Optimization using Dynamic Neighborhood Particle Swarm Optimization. In: Proceedings of the IEEE Congress on Evolutionary Computation, vol. 2, pp. 1677–1681 (May 2002)
3. Hu, X., Eberhart, R.C., Shi, Y.: Particle Swarm with Extended Memory for Multi-objective Optimization. In: Proceedings of the IEEE Swarm Itelligence Symposium, Indianapolis, Indiana, USA, pp. 53–57 (2003)
4. Parsopoulos, K.E., Vrahatis, M.N.: Particle Swarm Optimization Method in Multi-objective Problem. In: Proceedings of the ACM Symposium on Applied Computing, pp. 603–607 (2002)
5. Parsopoulos, K.E., Vrahatis, M.N.: Recent Approaches to Global Optimization Problems through Particle Swarm Optimization. Natural Computing 1(2-3), 235–306 (2002)
6. Coello, C.A., Lechuga, M.S.: MOPSO: a proposal for multiple objective particle swarm optimization. In: Evolutionary Computation (2002)
7. Villalobos-Arias, M., Coello, C.A.: Asymptotic convergence of met-heuristics for Multi-objective optimization problems. Soft Computing 10(11), 1001–1005 (2006)

8. Lei, D., Yan, X.-P.: Multi-objective intelligent optimization problems and application, pp. 38–40. Science Press, Beijing (2009)
9. Li, X.-D.: A Non-Dominated Sorting Particle Swarm Optimizer for Multi-Objective Optimization. In: Cantú-Paz, E., Foster, J.A., Deb, K., Davis, L., Roy, R., O'Reilly, U.-M., Beyer, H.-G., Kendall, G., Wilson, S.W., Harman, M., Wegener, J., Dasgupta, D., Potter, M.A., Schultz, A., Dowsland, K.A., Jonoska, N., Miller, J., Standish, R.K. (eds.) GECCO 2003. LNCS, vol. 2723, pp. 37–48. Springer, Heidelberg (2003)
10. Li, Z.-K., Tan, J.-R., Feng, Y.-X., Fang, H.: Multi-objective particle swarm optimization algorithm based on crowding distance sorting and its application. Computer Integrated Manufacturing Systems 7, 1329–1336 (2008)
11. Wu, W., Yan, G.: Dynamic particle swarm algorithm for multi-objective optimization based on crowding distance. Computer Engineering and Design, 1421–1425 (2011)
12. Yang, S.-X.: Multi-objective particle swarm optimization based on crowding distance. Computer Engineering and Applications, 222–246 (2009)
13. Berger, J.O.: Statistical decision theory and Bayesian analysis. Spring-Verlag world publishing corporation (1985)
14. Huang, V.L., Suganthan, P.N., et al.: Multi-objective differential evolution with external archive and harmonic distance-based diversity measure. Technical Report, Nanyang Technological University (2005)

Exponential Inertia Weight for Particle Swarm Optimization

T.O. Ting[1,*], Yuhui Shi[1], Shi Cheng[2], and Sanghyuk Lee[1]

[1] Dept. Electrical and Electronic Engineering,
Xi'an Jiaotong-Liverpool University, Suzhou, China
[2] Dept. Electrical Engineering and Electronics,
University of Liverpool, Liverpool, UK
toting@xjtlu.edu.cn

Abstract. The exponential inertia weight is proposed in this work aiming to improve the search quality of Particle Swarm Optimization (PSO) algorithm. This idea is based on the adaptive crossover rate used in Differential Evolution (DE) algorithm. The same formula is adopted and applied to inertia weight, *w*. We further investigate the characteristics of the adaptive *w* graphically and careful analysis showed that there exists two important parameters in the equation for adaptive *w*; one acting as the local attractor and the other as the global attractor. The 23 benchmark problems are adopted as test bed in this study; consisting of both high and low dimensional problems. Simulation results showed that the proposed method achieved significant improvement compared to the linearly decreasing method technique that is used widely in literature.

Keywords: Benchmark functions, exponential inertia weight, Particle Swarm Optimization.

1 Introduction

Inertia weight, *w* has been one of the important parameters in Particle Swarm Optimization (PSO) algorithm. It has been known that *w* plays a crucial role in guaranteeing the robustness of PSO. Y. Shi and R. Eberhart first introduced the concept of *w* in PSO [1]. To date, a myriad of investigations concerning this parameter has been carried out [2-4]. The work by Bansal et al. [3] compared 15 inertia weight strategies available from literature. However, only 5 benchmark functions are employed in his work. Han et al. compares several inertia weights in his work [4]. Again, using only 3 benchmark problems is not adequate to validate the results obtained and conclusions made may not be true when more benchmark problems are considered. The improvement contributed by manipulation of *w* can be categorized into few categories, namely exploration and exploitation, mutating *w* and adaptive *w*.

* Corresponding author.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 83–90, 2012.

The first category, exploration and exploitation is based on the concept of incorporating high value of *w* and decreasing its value along the iteration. When *w* is high, the algorithm is capable of global search and as *w* decreases, the local search capability is more significant. This concept is implemented as linearly decreasing *w* as proposed by Shi in [1]. This technique is by far the most popular one and has been applied successfully in many works [5-7]. Many other variants are built upon this concept. Xin et al. introduced multi-stage linearly decreasing inertia weight[8]. In [9], instead of decreasing *w* from 0.9 to 0.4, the *w* is increased from 0.4 to 0.9. The range of variation of *w* is within 0.9 to 0.4 in his work.

The second category of improvement via *w* involves manipulation of *w* in a stochastic manner. Miranda proposed mutated *w* in [10] for reliability calculation in power systems. Feng proposed the chaotic *w* [11-12] based on the linearly decreasing *w* and random *w*. The stochastic mutation of *w* is introduced by Li in [13]. The method is performed along with the linearly decreasing *w*. There are in fact limited works under this category as the stochastic strategies introduce disturbances into the algorithm and these may not perform well on a wide range of problems.

Lastly, the third category implements *w* in an adaptive manner. This is perhaps the trend in many current works. Many works proposed ways to incorporate the information such as ranking [14], diversity [15], convergence, and swarm size[16] into *w* as this will dynamically adjust *w* based on the performance criteria received from the population. Work by Chen [16] relates the inertia weight with problem complexity (dimension size) and population size. If the swarm size is small, a larger inertia weight is employed to improve the global search capability to locate the global optimum. For an optimization problem on multi-dimension complex solution space, a larger inertia weight is employed to strengthen the ability to escape from local optima.

Many works on the inertia weight has been done, however, there is not clear justification of how this parameter can be adjusted to improve the performance of PSO algorithm. Thus, we aim to investigate this property in this work. The rest of the paper is organized as follows. Section 2 explains the proposed method. Parameter settings are described in Section 3. The benchmark problems used as the test bed are described under this section. This is followed by results and discussions in Section 4 and finally the conclusions in Section 5.

2 Proposed Exponential Inertia Weight, *w*

The idea of adaptive *w* in this paper originated from the work by Ao and Chi in [17]. In this reference, the author proposes an Adaptive Crossover Rate, *ACR* for Differential Evolution (DE) algorithm. This *ACR* is defined as:

$$CR = CR_0 \cdot e^{-a\left(\frac{g}{G}\right)^b} \tag{1}$$

where CR_0 is the initial crossover rate = 0.8 or 0.85, g is the current generation number, G is the maximum number of generations, a = 2, b = 2 or 3. The adaptive function for the crossover rate is simply crafted based on the logic of high CR at the early of run to prevent premature convergence and low CR at the end of run to enhance the local search. This concept is exactly the same for the case of inertia weight, w in PSO. Thus, the ACR is converted to adaptive w as follows:

$$w = w_0 \cdot e^{-a\left(\frac{g}{G}\right)^b} \tag{2}$$

whereby w_0 is set to 0.9 in our work here. This value is chosen as an initial value in many works [5-8]. Further, two graphs are plotted, depicting the characteristics of parameters a and b. These are plotted in Figs. 1 and 2 below. From Fig. 1, by increasing a from 0 to 3 with a step of 0.5, it has the ability to push down the value of w, resulting in a curvilinear curve along the iterations. Thus, we name the parameter a as local search attractor. Note that when a=0, the inertia weight becomes a static value of 0.9 as the initial value w_0 is set to 0.9. In the same diagram, take note that the third curve (a=1.0) starts from 0.9 and ends at approximately 0.32; almost similar to linearly decreasing w.

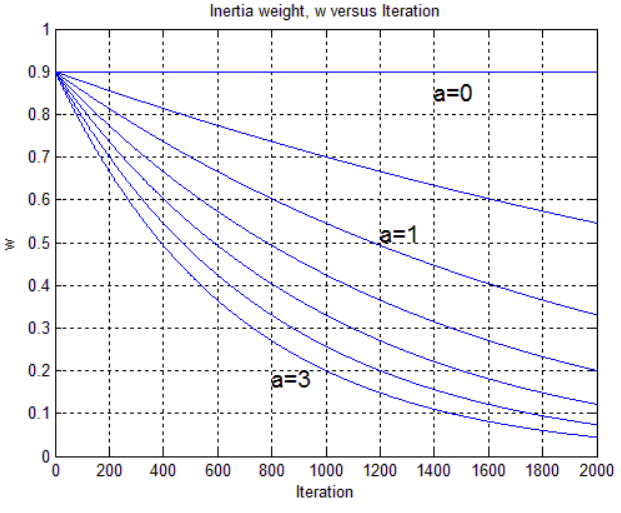


Fig. 1. Characteristics of local search attractor, a varies from 0 to 3 step 0.5 while b is set to 1

On the other hand, the parameter b has the opposite characteristic. When b is increased, it has the ability to pull up the curve resulting in higher value of w at the early run along the iterations. Hence, b is called global search attractor in this context. Again, note that when b=0, it is in fact a static w with the value approximate to 0.32. The third curve from below (b=1) is exactly the same as the curve (a=1) in Fig. 1 as both has the same numerical values (a=1, b=1).

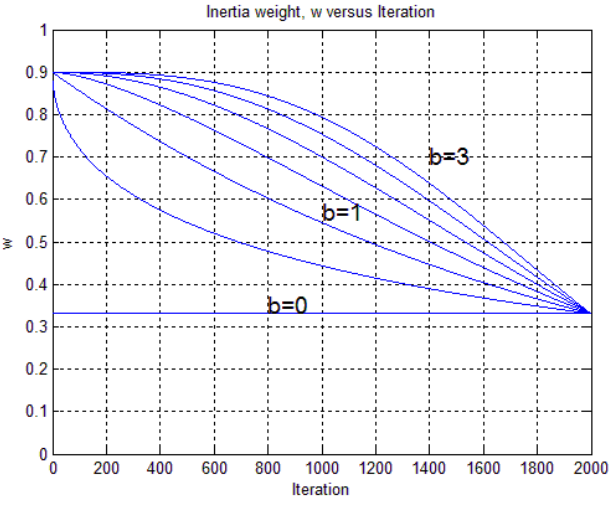


Fig. 2. Characteristics of global search attractor, b varies from 0 to 3 step 0.5 while a is set to 1

Generally, the ability of pulling up and pushing down the value of w using (2) makes the propose method ideal in PSO. Implementing this operator into any program is just a simple task and it does not add significant additional computational cost to the PSO algorithm.

3 Parameter Settings

The following settings are adopted in the PSO algorithm applied in this work. The population size is set to 20, acceleration coefficients; c_1 and c_2 are both set to 2.0. A dimensional value is reinitialized upon violation of either upper or lower boundaries. No maximum velocity, V_{max} is imposed in this setting. The results of linearly decreasing inertia weight are applied as standard comparison for the exponential w. The setting for exponential w is tabulated in Table 1.

Table 1. Setting for for exponential w

Method	Setting	Method	Setting
A	a=1, b=0.5	F	a=0.5, b=1
B	a=1, b=1.0	G	a=1.0, b=1
C	a=1, b=1.5	H	a=1.5, b=1
D	a=1, b=2.0	I	a=2.0, b=1
E	a=1, b=2.5	J	a=2.5, b=1

The widely known 23 benchmark problems [18] are adopted as test bed to validate the effectiveness of exponential w proposed in this work. All the parameter settings applied are similar. Results recorded as mean and standard deviation from 50 trials.

4 Results and Discussions

Results above are the summary of the performance of each method compared to the results of standard PSO. A shaded cell is visible when the mean obtained is better or equal to the standard PSO's. Otherwise, the cell is left empty instead of 0 for better readability. The total number of results that outperform the standard PSO is depicted in the last row of the table. The numerical values of the mean are available in Table 3.

Table 2. Results of simulation using different settings of a and b

	f	Function name	SPSO (mean)	δ	a is fixed at 1.0					b is fixed at 1.0				
					A	B	C	D	E	F	G	H	I	J
High Dimensional f	f_1	*Sphere*	6.61E-05	0.00	■	■	■				■	■	■	■
	f_2	*Schwefel 2.22*	4.19E-06	0.00	■	■	■				■	■	■	■
	f_3	*Schwefel 1.2*	35.98	25.84	■	■	■				■	■	■	■
	f_4	*Schwefel 2.21*	5.34	1.93	■	■	■	■			■	■	■	■
	f_5	*Rosenbrock*	26.69	31.27	■	■	■	■	■	■	■	■	■	■
	f_6	*Step*	8.00E-01	1.11	■	■					■		■	■
	f_7	*Quartic*	3.73E-02	0.01	■	■	■				■	■	■	■
	f_8	*Schwefel*	-7303.35	1132.26	■	■	■	■	■		■	■	■	■
	f_9	*Rastrigin*	24.36	6.42										
	f_{10}	*Ackley*	0.23	0.53										
	f_{11}	*Griewank*	1.55E-02	0.02		■							■	
	f_{12}	*Penalized P8*	4.00E-01	0.59	■	■	■				■	■	■	■
	f_{13}	*Penalized P16*	1.65E-01	0.52	■	■	■				■	■	■	■
Low Dimensional f	f_{14}	*Foxholes*	1.18	0.62										
	f_{15}	*Kowalik*	3.07E-04	0.00	■	■	■				■	■	■	■
	f_{16}	*Six-hump Camel-Back*	-1.0316280	0.00										
	f_{17}	*Branin*	4.02E-01	0.01	■	■	■				■	■	■	■
	f_{18}	*Goldstein-Price*	3.0000001	0.00	■	■	■				■	■	■	■
	f_{19}	*Hartman-3*	-3.8622	0.00	■	■	■	■			■	■	■	■
	f_{20}	*Hartman-6*	-3.2500	0.09	■	■					■	■	■	■
	f_{21}	*Shekel-5*	-5.89	3.48		■	■	■	■	■	■		■	
	f_{22}	*Shekel-7*	-6.78	3.57		■	■	■	■	■			■	
	f_{23}	*Shekel-10*	-7.80	3.57				■	■	■		■		
Total improvements, $\sum$					16	19	16	7	5	4	17	16	19	17

Results from the simulation above show that the use of proposed w is effective in tackling global optimum as generally majority of the methods outperform linearly decreasing method. This is true for methods A, B, C, G, H, I, J whereby 15 and above benchmark functions are solved with improvement. From left to right for methods A-E, as the global attractor, b is increased, the algorithm lack convergence capability. This is due to the reason that as b increases, the value of w increases and thus the algorithm is capable of global optimum and lack convergence capability. Note that f_{21}, f_{22} and f_{23} favor higher w to find the global optimum more accurately. Again, numerical values for these results are tabulated in Tables 3.

To ease our analysis, methods A-J are grouped into three categories, namely global, balance and local categories:

Global search ($b > a$) : Methods C, D, E and F
Balance search ($a = b$) : Methods B and H
Local search ($a > b$) : Methods A, H, I and J

The grouping of the methods above is based on the concept that when the global attractor is greater than the local attractor ($b>a$), the PSO algorithm is capable of global search. Similarly, the algorithm tend to be local searcher when the local attractor is greater than the global attractor ($a>b$). The balance group has the same value for both attractors. Results for each of these groups are recorded in Table 3. For each numerical value, the shading denotes the degree of the results obtained for each benchmark problem among all participating methods. Hereby, brighter background shading denotes better results.

At a glance of Table 3, it is easy to come to a conclusion that the local category methods are favored in this case as in this category majority of the mean recorded are above average (brighter shading). However the drawback of local category is the danger of being trapped in local optima. This is true for the case of f_{21}, f_{22} and f_{23}. For simplicity, we would propose the use of balance method (B and H). Note that B and H are both the same as a and b are both set to unity. In this category, there is a balance between global exploration and local exploitation.

Table 3. Results using global search methods (Methods C, D, E, F and G)

	Global Search Methods				Balance	Local Search Methods			
f	C	D	E	F	B / H	A	H	I	J
f_1	2.21E-06	1.44E-04	1.44E-03	4.63E-01	4.50E-10	2.09E-15	6.93E-14	1.82E-16	1.66E-14
f_2	5.35E-07	1.39E-04	1.10E-04	3.43E-03	1.34E-09	1.32E-10	4.39E-08	1.50E-06	9.14E-06
f_3	11.98	45.96	108.88	828.6	1.98	0.03	0.1	0.08	0.25
f_4	3.76	4.91	6.68	14.81	2.28	0.55	0.87	1.06	1.38
f_5	20.78	18.02	19.35	20.35	12.13	12.72	20.65	18.86	24.33
f_6	8.80E-01	2.40E+00	3.26E+00	3.68E+00	3.20E-01	4.80E-01	2.56E+00	5.60E-01	6.80E-01
f_7	2.97E-02	4.24E-02	4.90E-02	7.18E-02	2.31E-02	1.84E-02	2.14E-02	2.07E-02	2.35E-02
f_8	-8434.94	-8053.32	-7777.45	-4931.8	-8948.23	-9007.49	-8764.27	-8834.95	-8908.41
f_9	27.24	26.93	28.48	24.55	27.08	29.13	27.52	29.35	31.68
f_{10}	0.64	0.6	0.88	1.21	0.38	0.51	1.05	1.09	1.22
f_{11}	1.88E-02	1.80E-02	1.69E-02	7.88E-02	1.52E-02	1.73E-02	1.95E-02	1.51E-02	1.34E-02
f_{12}	1.88E-01	8.18E-01	1.24E+00	5.15E+00	2.34E-01	1.89E-01	2.12E-01	1.50E-01	2.66E-01
f_{13}	1.25E-01	1.87E-01	1.62E+00	1.19E+01	3.19E-02	7.67E-03	3.47E-02	6.67E-02	2.04E-01
f_{14}	1.36	1.22	1.41	1.24	1.3	1.43	1.38	1.29	1.1
f_{15}	3.07E-04	3.08E-04	3.08E-04	3.12E-04	3.07E-04	3.07E-04	3.07E-04	3.07E-04	3.07E-04
f_{16}	-1.03163	-1.03163	-1.03163	-1.03163	-1.03163	-1.03163	-1.03163	-1.03163	-1.03163
f_{17}	3.99E-01	4.02E-01	4.05E-01	4.09E-01	3.99E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01
f_{18}	3	3.000001	3.000005	3.000032	3	3	3	3	3
f_{19}	-3.8627	-3.8623	-3.8613	-3.8609	-3.8628	-3.8628	-3.8628	-3.8628	-3.8628
f_{20}	-3.2374	-3.2222	-3.2482	-3.2224	-3.2662	-3.2555	-3.2691	-3.2668	-3.2744
f_{21}	-6.35	-6.64	-6.2	-6.43	-6.1	-5.38	-5.51	-6.8	-6.59
f_{22}	-7.27	-7.84	-7.67	-7.14	-7.43	-6.35	-6.63	-7.5	-6.43
f_{23}	-7.33	-8.53	-8.12	-7.88	-7.68	-6.45	-8.28	-5.92	-6.31

We then apply method A (a=1, b=0.5) to half of the population and the other half uses setting D (a=1, b=2). Note that method A is capable of local search ($a>b$) and method D is capable of global search ($b>a$). The result came to be as expected; 19 out of 23 functions are solved with improvement compared to linearly decreasing w. Nevertheless, for convenience, we recommend the setting of a=1 and b=1 for general purposes. Besides, we also run two additional simulations. The first one involve

choosing either method A or D in a random manner. In the second simulation, we apply a switch from D to A after half of the total generation. Both simulations have the same conclusion as mentioned above with 19 and 18 improvements as compared to the linearly decreasing *w*.

5 Conclusions

In this work, we proposed the exponential inertia weight, *w* to improve the search quality of Particle Swarm Optimization (PSO) algorithm. This exponential *w* has simple mathematical term shown by Eq. (2). The mathematical term originated from the work of Ao and Chi in [17] that is based on the concept of adaptive crossover rate used in Differential Evolution (DE) algorithm. The same formula is adopted and applied to inertia weight, *w*. We further investigate the characteristics of the adaptive *w* graphically and careful analysis showed that there exist two important parameters in the equation for adaptive *w*; one acting as the local attractor, *a* and the other as the global attractor, *b*. We further analyze that the algorithm is capable of global search and local search when ($b>a$) and ($a>b$) respectively. Simulation results showed that the proposed method has better performance in comparison to the linearly decreasing inertia weight that is used widely in many significant works. Among all ten methods, A to J; seven methods (A, B, C, G, H, I, J) managed to obtained better results for 15 and above benchmark problems as compared to linearly decreasing *w*. For convenience, we recommend the setting of both local and global attractors to unity values ($a=b=1.0$). The proposed technique is reliable as 23 benchmark problems are adopted to validate the robustness of the exponential *w*. Further works should investigate and relate information such as convergence, diversity, swarm size, number of dimensions etc. to either local attractor, *a* or global attractor, *b*. Once an effective relationship is found, *a* and *b* will be adjusted automatically and hence resulting in an adaptive *w*. This remains as an important work for future.

References

1. Shi, Y., Eberhart, R.: A Modified Particle Swarm Optimizer. In: IEEE World Congress on Computational Intelligence Evolutionary Computation Proceedings, 1998, pp. 69–73 (1998)
2. Hussain, Z., Noor, M.H.M., Ahmad, K.A., et al.: Evaluation of Spreading Factor Inertial Weight PSO for FLC of FES-Assisted Paraplegic Indoor Rowing Exercise. In: 2011 IEEE 7th International Colloquium on Signal Processing and its Applications (CSPA), pp. 430–434 (2011)
3. Bansal, J.C., Singh, P.K., Saraswat, M., et al.: Inertia Weight Strategies in Particle Swarm Optimization. In: 2011 Third World Congress on Nature and Biologically Inspired Computing (NaBIC), pp. 633–640 (2011)
4. Han, W., Yang, P., Ren, H., et al.: Comparison Study of several Kinds of Inertia Weights for PSO. In: 2010 IEEE International Conference on Progress in Informatics and Computing (PIC), vol. 1, pp. 280–284 (2010)

5. Mekhamer, S.F., Moustafa, Y.G., EI-Sherif, N., et al.: A Modified Particle Swarm Optimizer Applied to the Solution of the Economic Dispatch Problem. In: 2004 International Conference on Electrical, Electronic and Computer Engineering, ICEEC 2004, pp. 725–731 (2004)
6. Zhu, Z., Zhou, J., Ji, Z., et al.: DNA Sequence Compression using Adaptive Particle Swarm Optimization-Based Memetic Algorithm. IEEE Transactions on Evolutionary Computation 15, 643–658 (2011)
7. Seo, J.-H., Im, C.-H., Heo, C.G., et al.: Multimodal Function Optimization Based on Particle Swarm Optimization. IEEE Transactions on Magnetics 42, 1095–1098 (2006)
8. Xin, J., Chen, G., Hai, Y.: A Particle Swarm Optimizer with Multi-Stage Linearly-Decreasing Inertia Weight. In: International Joint Conference on Computational Sciences and Optimization, CSO 2009, vol. 1, pp. 505–508 (2009)
9. Zheng, Y.-L., Ma, L.-H., Zhang, L.-Y., et al.: Empirical Study of Particle Swarm Optimizer with an Increasing Inertia Weight. In: The 2003 Congress on Evolutionary Computation, CEC 2003, vol. 1, pp. 221–226 (2003)
10. Miranda, V., de Magalhaes Carvalho, L., da Rosa, M.A., et al.: Improving Power System Reliability Calculation Efficiency with EPSO Variants. IEEE Transactions on Power Systems 24, 1772–1779 (2009)
11. Feng, Y., Teng, G.-F., Wang, A.-X., et al.: Chaotic Inertia Weight in Particle Swarm Optimization. In: Second International Conference on Innovative Computing, Information and Control, ICICIC 2007, p. 475 (2007)
12. Feng, Y., Teng, G.-F., Wang, A.-X.: Comparing with Chaotic Inertia Weights in Particle Swarm Optimization. In: 2007 International Conference on Machine Learning and Cybernetics, vol. 1, pp. 329–333 (2007)
13. Li, H.-R., Gao, Y.-L.: Particle Swarm Optimization Algorithm with Exponent Decreasing Inertia Weight and Stochastic Mutation. In: Second International Conference on Information and Computing Science, ICIC 2009, vol. 1, pp. 66–69 (2009)
14. Mahor, A., Prasad, V., Rangnekar, S.: Scheduling of Cascaded Hydro Power System: A New Self Adaptive Inertia Weight Particle Swarm Optimization Approach. In: International Conference on Advances in Recent Technologies in Communication and Computing, ARTCom 2009, pp. 565–570 (2009)
15. Zhan, Z.-H., Zhang, J., Li, Y., et al.: Adaptive Particle Swarm Optimization. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 39, 1362–1381 (2009)
16. Dong, C., Wang, G., Chen, Z., et al.: A Method of Self-Adaptive Inertia Weight for PSO. In: 2008 International Conference on Computer Science and Software Engineering, vol. 1, pp. 1195–1198 (2008)
17. Ao, Y., Chi, H.: An Adaptive Differential Evolution to Solve Constrained Optimization Problems in Engineering Design. Scientific Research 2, 65–77 (2010)
18. Yao, X., Liu, Y., Lin, G.: Evolutionary Programming made Faster. IEEE Transactions on Evolutionary Computation 3, 82–102 (1999)

A Coevolutionary Memetic Particle Swarm Optimizer

Jiarui Zhou[1,2], Zhen Ji[1,*], Zexuan Zhu[1], and Siping Chen[1,2]

[1] College of Biomedical Engineering and Instrument Science, Zhejiang University, Hangzhou 310027, China
[2] Shenzhen City Key Laboratory of Embedded System Design, College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China

Abstract. This paper presents a coevolutionary memetic particle swarm optimizer (CMPSO) for the global optimization of numerical functions. CMPSO simplifies the update rules of the global evolution and utilizes five different effective local search strategies for individual improvement. The combination of the local search strategy and its corresponding computational budget is defined as coevolutionary meme (CM). CMPSO co-evolves both CMs and a single particle position recording the historical best solution that is optimized by the CMs in each generation. The experimental results on 7 unimodal and 22 multimodal benchmark functions demonstrate that CMPSO obtains better performance than other representative state-of-the-art PSO variances. Particularly, CMPSO is shown to have higher convergence speed.

Keywords: Particle swarm optimization, coevolution, coevolutionary meme, local search strategies.

1 Introduction

Inspired by the collective behavior of natural creatures, the swarm intelligence (SI) algorithms are wildly applied in solving complex real world problems. Two crucial issues determine the convergence performance of the SI algorithms:

- ***Effectiveness*:** The fitness improvement obtained by the algorithm in each generation. Particularly, effectiveness refers to how well the algorithm can adapt its searching behavior in different stages of the optimization, so that the fitness improvement of each generation is maximized.

* All correspondence should be addressed to Prof. Zhen Ji, Email: jizhen@szu.edu.cn, Tel: +86 755 26557413. This work was supported in part by the National Natural Science Foundation of China, under Grants 61171125 and 61001185, in part by the Fok Ying-Tung Education Foundation, Guangdong Natural Science Foundation, under Grant 10151806001000002, in part by the Foundation for Distinguished Young Talents in Higher Education of Guangdong, under Grant LYM10113, and in part by the Shenzhen City Foundation for Distinguished Young Scientists.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 91–100, 2012.

- ***Attainableness*:** The goodness of the solutions found by the algorithm. For instance, in a minimization problem, the smaller the fitness value is obtained by the algorithm, the better the *Attainableness* it is.

An ideal SI algorithm has the highest *Effectiveness* and the best *Attainableness*, so that it is able to obtain the optimal solution with minimal number of generations. But these two issues are commonly conflicting. Individuals of the worst fitness values are more likely to obtain the largest fitness improvement in each evolutionary generation. Contrarily, individuals near the global optima, which have the best fitness values, are usually fine-tuning in the optimum region, and therefore attain smaller or even none fitness improvement. To accelerate the convergence of SI algorithms, the balance between *Effectiveness* and *Attainableness* should be well maintained.

Traditional SI algorithms like the particle swarm optimization (PSO) [1] normally utilize permanent parameters settings and update strategies throughout the search. Their *Effectiveness* is deteriorated for not able to adjust the searching behavior along with the optimization process. To overcome this defect, some "adaptive" algorithms are proposed. For instance, J. J. Liang et al. [2] proposed a comprehensive learning particle swarm optimizer (CLPSO) for the global optimization of multimodal functions. CLPSO employs a novel comprehensive learning strategy whereby all particles historical best information is involved in position update. It is shown to work well on complex multimodal functions. Z. H. Zhan et al. [3] adjusted the conventional PSO by introducing orthogonal experimental design (OED) in position learning. The algorithm, namely OLPSO, achieves promising results on both unimodal and multimodal problems. M. A. M. de Oca et al. [4] proposed the Tuned IPSOLS by introducing six local search strategies in the global evolution process of PSO. The algorithm is shown to obtain high performance on large-scale optimization problems. Z. Y. Yang et al. [5] proposed a new generalized adaptive differential evolution optimizer (GaDE) by employing novel generalized adaptation scheme in parameters selection. Experimental results demonstrate that the GaDE is competitive in both performance and scalability aspects. However, the contradiction between *Effectiveness* and *Attainableness* still remains in these algorithms.

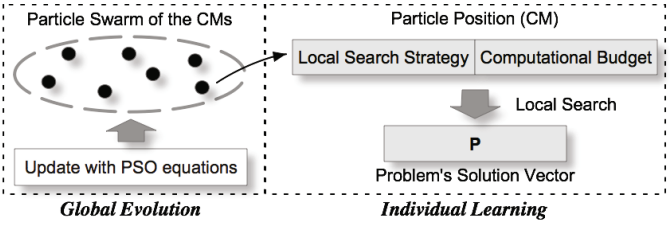


Fig. 1. Schematic diagram of the CMPSO

In this paper we propose a coevolutionary memetic particle swarm optimizer (CMPSO). Particularly, CMPSO utilizes simplified PSO update equations for global evolution. Five effective local search strategies, including the Davidon, Fletcher and Powells Quasi-Newton Strategy (DFP) [6], the Davies, Swann, and Campey with

Gram-Schmidt orthogonalization (DSCG) [7], the chaos particle swarm optimizer (CPSO) [8], the random walks in Dirichlet environment (RWDE) [9], and the simulated annealing (SA) [10] are employed for individual learning. These strategies are effective in optimizing the problems on different stages of the search. The combination of the local search strategy and its corresponding computational budget is defined as coevolutionary meme (CM). CMPSO co-evolves a single position vector and the CMs, as shown in Fig. 1. The position vector **P**, namely the candidate solution vector of the problem, records only the historical best solution and represents the *Attainableness* of the algorithm. All CMs are performed on **P** in each generation and adjusted according to their *Effectiveness*. Thereby the confliction between *Effectiveness* and *Attainableness* is harmonized. Experimental results on 7 unimodal and 22 multimodal benchmark functions demonstrate that CMPSO is capable of attaining better performance than other representative PSO improvements.

The remainder of this paper is organized as follows. Section 2 describes the procedure of CMPSO. Section 3 presents the experimental results of CMPSO and other representative algorithms on the benchmark functions. Finally the conclusion is given in Section 4.

2 Coevolutionary Memetic Particle Swarm Optimizer

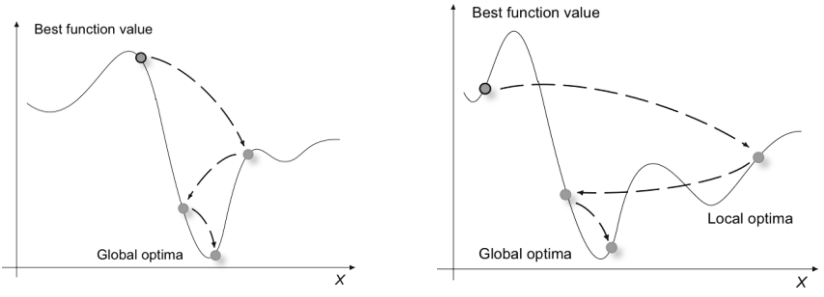
2.1 Coevolutionary Meme

In CMPSO, the DFP, DSCG, CPSO, RWDE, and SA are utilized as the local search strategies. These strategies are effective in optimizing different problems [6-10]. The strengths of these strategies are summarized in Table 1.

Table 1. Strengths of the local search strategies

Index	Strategy	Effective in
1	DFP	Unimodal problems
2	DSCG	Unimodal problems
3	CPSO	Unimodal and simple multimodal problems
4	RWDE	Complex multimodal problems
5	SA	Unimodal and multimodal problems, less effective

The *Effectiveness* of the CMPSO is promoted by selecting proper local search strategies and their corresponding computational budgets for particular searching stages. As shown in Fig. 2, when the search position is near the global optima, the DFP or DSCG strategy could be employed to accelerate the convergence speed [11]. Contrarily, the selection of RWDE strategy allows the solution vector to explore in a large region and escape from local optima. SA is less effective compared to other strategies, but it can keep drilling down in the fitness landscape on both unimodal and multimodal problems. The computational budget of a local search strategy denotes the certainty of selecting this strategy. Larger budget indicates the strategy is believed to be able to obtain higher fitness improvement in the current generation, and CMPSO allocates sufficient computational resources to it. On the contrast, smaller budget means the CMPSO is just having a trial on an uncertain strategy.



(a) Accelerate the convergence speed by utilizing DFP and DSCG

(b) Prevent premature convergence by utilizing RWDE and SA

Fig. 2. Adapting local search strategies to particular searching stages

The selection of local search strategies and their computational budgets determines the overall *Effectiveness* of the algorithm. To optimize the selection, we introduce a novel coevolutionary framework of PSO. In this study, the combination of a local search strategy and its corresponding computational budget is defined as a coevolutionary meme (CM). Coined by R. Dawkins, the word 'meme' represents the transmissible and replicable unit information stored in brain that serves as instruction for carrying out behavior [12]. In the context of memetic algorithm, meme is also interchangeable with local search strategy [11]. Here, the CM is defined as:

$$\mathbf{M}_i = [S_i, C_i] \tag{1}$$

in which S_i denotes the type index of the local search strategy, taking value in $R_S = \{1, 2, 3, 4, 5\}$. For instance, $S_i = 1$ indicates to select DFP for individual learning, $S_i = 2$ denotes the selection of DSCG, and so on. Variable C_i is the computational budget defined in terms of number of fitness evaluations (FEs). C_i is set in the range $R_C = [100, 1000]$. In CMPSO, the CMs are optimized in the global search of PSO and all the individual learning processes are performed on the same solution vector **P**.

2.2 Tabu Vector and Reinitialization Operation

To prevent premature convergence, a tabu vector and a reinitialization operation are introduced in CMPSO. The fitness improvement of each CM is denoted as follows:

$$\delta = fitness\_old - fitness\_new \tag{2}$$

where *fitness_old* and *fitness_new* are the fitness values obtained by the local search strategy before and after the individual learning process respectively. If the fitness improvement on the kth generation is $\delta^k = 0$, the computational budget used in the individual learning is added to a tabu vector:

$$\mathbf{T}[S^k] = \mathbf{T}[S^k] + C^k \tag{3}$$

where S^k and C^k is the local search strategy index and its computational budget selected in the kth generation evolution.

The values in the tabu vector record the total computational budget a local search strategy takes without gaining any fitness improvement. When all the values in the tabu vector reach the upper bound of the computational budget range R_C, indicating that no individual learning strategy can optimize the current solution vector **P** within the maximum FEs budget, the reinitialization operation is performed by randomly sampling the solution vector **P** in range $R_P = [R_{\min}, R_{\max}]$. Variable $R_{\min}$ and $R_{\max}$ are the lower and upper bound of the solution space, respectively.

Algorithm 1. Procedure of the CMPSO

```
1:  BEGIN
2:  Initialize a particle swarm ps of CMs by randomly sampling each particle
    position M_i in R_S ⊗ R_C;
3:  Set the global best position M_gbest = M_1, the fitness improvement of M_gbest i.e.,
    δ_gbest = 0;
4:  Randomly sample the candidate solution vector P in R_P;
6:  Initialize the tabu vector T by setting all the values to 0;
7:  while stopping criterion is not satisfied do
8:      for each particle ps_i in the swarm do
9:          Update particle velocity V_i and position M_i based on (4) and (5);
10:         Select local search strategy S_i and computational budget C_i based on M_i;
11:         fitness_old = f (P);
12:           Perform individual learning with strategy S_i using computational budget
              C_i on P;
13:         fitness_new = f (P);
14:         Calculate the fitness improvement δ_i based on (2);
15:         if δ_i > δ_gbest then
16:             M_gbest = M_i;
17:             δ_gbest = δ_i;
18:         end if
19:         if δ_i = 0 then
20:             Update the tabu vector T based on (3);
21:             if all values in T reach the upper bound of R_C then
22:                 Perform reinitialization operation of P;
23:             end if
24:         end if
25:     end for
26: end while
27: END
```

2.3 The CMPSO Algorithm

CMPSO utilizes simplified update equations to optimize the particle swarm. Unlike conventional PSOs, in CMPSO the particle positions in the global evolution are not the solutions of the objective problem, but the CMs. The update equations of the global evolution are illustrated as follows:

$$\mathbf{V}_i^{k+1} = \mathbf{V}_i^k \Big/ 2 + r \times (\mathbf{M}_{gbest}^k - \mathbf{M}_i^k) \tag{4}$$

$$\mathbf{M}_i^{k+1} = \mathbf{M}_i^k + \mathbf{V}_i^{k+1} \tag{5}$$

where $\mathbf{V}_i^k$ and $\mathbf{M}_i^k$ are the velocity and position (CM) vector of the ith particle on the kth generation, respectively. Vector $\mathbf{M}_{gbest}$ is the global best CM that obtains the largest fitness improvement in its corresponding individual learning. Procedure of the CMPSO is demonstrated in Algorithm 1, in which $f(\cdot)$ denotes the fitness function.

By using global evolution to optimize the coevolutionary meme, the CMPSO is able to adapt the individual learning process to different stages of the optimization, thereby improving the overall *Effectiveness*. By performing all the individual learning processes on the same solution vector **P,** the *Attainableness* of the algorithm is optimized.

3 Experimental Results

7 unimodal and 22 multimodal benchmark functions from [13], [3], [14], and [15] are chosen to evaluate the performance of CMPSO. Four representative state-of-the-art PSO variances, including the PSO*w* [1], CLPSO [2], local OLPSO (OLPSO-L) [3], and global OLPSO (OLPSO-G) [3] are selected for comparison study. The parameter settings of all the algorithms are summarized in Table 2. The 29 benchmark functions are summarized in Table 3.

Table 2. Parameter settings of the algorithms

Algorithm	Parameters
PSO*w*	$\|ps\| = 20$, $w = 0.5$, $c1 = c2 = 2.0$
CLPSO	$\|ps\| = 20$, $w_0 = 0.9$, $w_1 = 0.7$, $c = 1.49445$, $m = 8$
OLPSO-L	$\|ps\| = 40$, $w = 0.9$, $c = 2.0$, $G = 5$
OLPSO-G	$\|ps\| = 40$, $w = 0.9$, $c = 2.0$, $G = 5$
CMPSO	$\|ps\| = 20$

To ensure fair comparisons, all the algorithms are given the same maximum number of 1E+05 FEs. The mean results and the variances of the fitness values obtained by all algorithms over 50 runs are reported in Table 4.

The results in Table 4 show that CMPSO obtains better performance compared to other PSO variances on both unimodal and multimodal benchmark functions. Particularly, CMPSO obtains the best mean values on 25 benchmark functions and the smallest variances on 22 functions. The CMPSO is competitive in performance and more stable than the other state-of-the-art PSOs. It is also worth noting that the CMPSO is easy to use for taking only one parameter, i.e. the swarm size $|ps|$.

Table 3. Twenty-nine benchmark functions used in the experiment

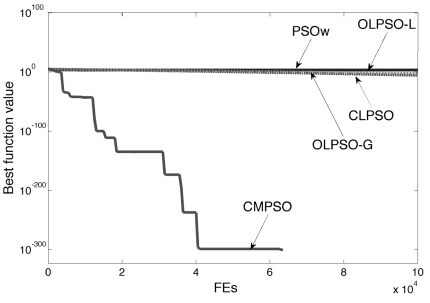
	Type	Description	Dimension	Range	Ref.
F_1	Unimodal	Sphere Model	30	[-100, 100]	[13]
F_2		Schwefel's Problem 2.22	30	[-10, 10]	[13]
F_3		Schwefel's Problem 1.2	30	[-100, 100]	[13]
F_4		Schwefel's Problem 2.2	30	[-100, 100]	[13]
F_5		Generalized Rosenbrock's Function	30	[-30, 30]	[13]
F_6		Step Function	30	[-100, 100]	[13]
F_7		Quartic Function	30	[-1.28, 1.28]	[13]
F_8	Multimodal	Generalized Schwefel's Problem 2.26	30	[-500, 500]	[13]
F_9		Generalized Rastrigin's Function	30	[-5.12, 5.12]	[13]
F_{10}		Ackley's Function	30	[-32, 32]	[13]
F_{11}		Generalized Griewank Function	30	[-600, 600]	[13]
F_{12}		Generalized Penalized Function 1	30	[-50, 50]	[13]
F_{13}		Generalized Penalized Function 2	30	[-50, 50]	[13]
F_{14}	Rotated and Shifted Multimodal	Rotated Schwefel	30	[-500, 500]	[3]
F_{15}		Rotated Rastrigin	30	[-5.12, 5.12]	[3]
F_{16}		Rotated Ackley	30	[-32, 32]	[3]
F_{17}		Rotated Griewank	30	[-600, 600]	[3]
F_{18}		Shifted Rosenbrock	30	[-100, 100]	[3]
F_{19}		Shifted Rastrigin	30	[-5, 5]	[3]
F_{20}	Composition Multimodal	Composition function 1	10	[-5, 5]	[14]
F_{21}		Composition function 2	10	[-5, 5]	[14]
F_{22}		Composition function 3	10	[-5, 5]	[14]
F_{23}		Composition function 4	10	[-5, 5]	[14]
F_{24}		Composition function 5	10	[-5, 5]	[14]
F_{25}		Composition function 6	10	[-5, 5]	[14]
F_{26}	Hybrid Composition Multimodal	Hybrid Composition Function F_{15}	10	[-5, 5]	[15]
F_{27}		Rotated Hybrid Composition Function F_{18}	10	[-5, 5]	[15]
F_{28}		Rotated Hybrid Composition Function F_{21}	10	[-5, 5]	[15]
F_{29}		Rotated Hybrid Composition Function F_{24}	10	[-5, 5]	[15]

The average convergence traces of all the algorithms over 50 runs on the benchmark functions F_1, F_8, F_{14}, F_{20}, and F_{27}, which are representatives of the five function groups, are illustrated in Fig. 3. To present the convergence trace more clearly, the vertical axis (Y-axis) is using the natural logarithmic scale.

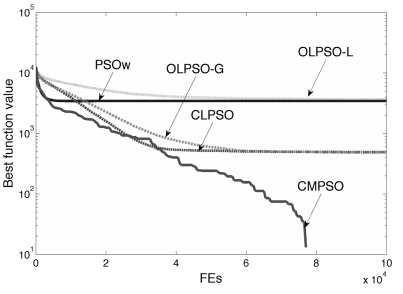
The results in Fig. 3 show that with properly selected CM for individual learning, the convergence speed of CMPSO is faster than other PSO variances. At the late stage of the search, when the other algorithms are almost stagnant, CMPSO keeps drilling down in the fitness landscape. The CMPSO is capable of preventing the premature convergence.

Table 4. Average results and variances obtained by the algorithms over 50 runs

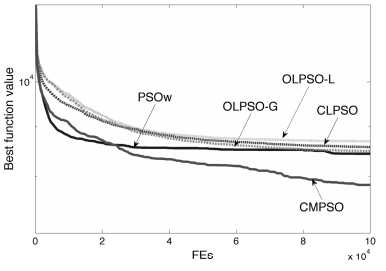
	F_1	F_2	F_3	F_4	F_5	F_6
PSO*w*	1.60E+03	3.76E+01	1.62E+04	1.48E+00	2.02E+05	1.78E+03
	± 1.74E+07	± 3.06E+02	± 9.02E+07	± 1.02E+00	± 1.59E+11	± 2.23E+07
CLPSO	1.52E-06	3.92E-05	8.90E+03	2.52E+01	1.95E+02	**0.00E+00**
	± 1.56E-12	± 3.39E-10	± 6.53E+06	± 6.94E+01	± 6.66E+03	**± 0.00E+00**
OLPSO-L	1.96E+02	1.02E+01	1.34E+04	3.94E+01	7.03E+06	8.29E+01
	± 2.70E+04	± 5.90E+01	± 1.11E+07	± 3.30E+01	± 6.14E+13	± 1.00E+04
OLPSO-G	3.16E-03	3.15E-03	**1.26E+03**	3.32E+00	1.69E+02	**0.00E+00**
	± 3.06E-06	± 1.52E-06	**± 5.83E+05**	± 9.38E-01	± 9.74E+03	**± 0.00E+00**
CMPSO	**0.00E+00**	**1.13E-157**	1.59E+04	**3.34E-20**	**1.65E+01**	**0.00E+00**
	± 0.00E+00	**± 0.00E+00**	± 3.07E+08	**± 5.43E-38**	**± 1.39E+01**	**± 0.00E+00**
	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}
PSO*w*	1.35E+01	3.44E+03	1.27E+02	1.02E+01	2.17E+01	1.72E-01
	± 2.92E+01	± 5.15E+05	± 1.12E+03	± 6.23E+01	± 1.48E+03	± 1.11E-01
CLPSO	9.92E+00	4.93E+02	**1.68E+00**	3.59E-01	7.58E-05	4.04E-08
	± 1.61E-01	± 9.37E+04	**± 2.36E+01**	± 6.25E+00	± 2.22E-08	± 1.45E-15
OLPSO-L	1.15E+01	3.67E+03	8.58E+01	6.05E+00	2.69E+00	2.05E+03
	± 6.68E-01	± 2.03E+05	± 3.30E+02	± 3.01E+00	± 2.29E+00	± 3.38E+07
OLPSO-G	9.44E+00	4.88E+02	7.19E+00	9.69E-03	1.64E-02	1.25E-02
	± 2.97E-01	± 4.07E+04	± 7.38E+00	± 1.45E-05	± 1.68E-04	± 1.56E-03
CMPSO	**8.77E+00**	**1.57E+02**	4.50E+00	**2.85E-08**	**0.00E+00**	**3.02E-17**
	± 1.67E-01	**± 7.86E+05**	± 4.86E+02	**± 0.00E+00**	**± 0.00E+00**	**± 4.16E-54**
	F_{13}	F_{14}	F_{15}	F_{16}	F_{17}	F_{18}
PSO*w*	2.78E-02	8.44E+03	2.63E+02	2.09E+01	2.53E+02	5.22E+09
	± 1.58E-02	± 5.05E+05	±6.11E+03	± 8.11E-03	± 3.70E+04	± 1.80E+19
CLPSO	1.46E-06	8.58E+03	2.31E+02	2.02E+01	2.70E+00	2.50E+02
	± 3.38E-12	± 2.29E+05	± 1.36E+03	± 4.17E-03	± 3.08E+00	± 5.61E+03
OLPSO-L	3.58E+04	8.70E+03	2.56E+02	2.06E+01	7.76E+01	1.91E+08
	± 8.69E+09	± 8.65E+04	± 1.24E+03	± 4.42E-01	± 1.14E+03	± 3.43E+16
OLPSO-G	2.44E-03	8.50E+03	1.46E+02	1.98E+01	1.24E+00	2.66E+02
	± 1.81E-05	± 2.72E+05	± 1.74E+03	± 1.22E+01	± 8.53E-02	± 1.08E+05
CMPSO	**2.88E-17**	**7.84E+03**	**5.41E+00**	**6.42E+00**	**0.00E+00**	**1.70E+01**
	± 1.71E-53	**± 1.72E+05**	**± 9.56E+02**	**± 8.62E+01**	**± 0.00E+00**	**± 2.08E+01**
	F_{19}	F_{20}	F_{21}	F_{22}	F_{23}	F_{24}
PSO*w*	1.53E+02	2.00E+02	2.58E+02	3.41E+02	5.68E+02	2.22E+02
	± 1.58E+03	± 1.38E+04	± 1.27E+04	± 1.93E+04	± 2.66E+04	± 2.88E+04
CLPSO	**1.79E+00**	3.22E+01	9.93E+01	1.73E+02	3.49E+02	3.33E+01
	± 1.36E+01	± 4.23E+03	± 1.23E+04	± 1.02E+04	± 7.40E+03	± 6.07E+03
OLPSO-L	1.19E+02	8.76E+00	5.96E+01	2.04E+02	3.61E+02	1.98E+01
	± 6.19E+02	± 5.76E+02	± 2.00E+03	± 4.41E+03	± 1.82E+03	± 1.92E+02
OLPSO-G	9.50E+00	4.80E+01	4.60E+01	1.66E+02	3.33E+02	4.87E+01
	± 1.51E+01	± 4.09E+03	± 5.69E+03	± 4.58E+03	± 4.66E+03	± 4.76E+03
CMPSO	9.48E+01	**1.13E-22**	**1.34E+01**	**1.38E+02**	**3.03E+02**	**1.38E+01**
	± 1.82E+02	**± 9.71E-44**	**± 9.18E+01**	**± 4.23E+02**	**± 1.43E+02**	**± 2.82E+01**
	F_{25}	F_{26}	F_{27}	F_{28}	F_{29}	
PSO*w*	8.48E+02	5.37E+02	9.88E+02	1.15E+03	1.07E+03	
	±1.82E+04	±4.19E+04	±1.84E+04	±4.31E+04	±1.26E+04	
CLPSO	6.83E+02	**1.00E+02**	7.64E+02	5.62E+02	2.22E+02	
	±3.37E+04	**±1.09E+04**	±1.86E+04	±4.22E+04	±1.34E+04	
OLPSO-L	5.35E+02	1.41E+02	7.37E+02	5.88E+02	2.53E+02	
	±3.82E+03	±2.33E+03	±1.74E+04	±6.00E+04	±1.09E+04	
OLPSO-G	7.07E+02	1.49E+02	7.23E+02	6.32E+02	2.80E+02	
	±3.78E+04	±2.74E+04	±3.29E+04	±1.13E+05	±3.08E+04	
CMPSO	**5.05E+02**	1.70E+02	**6.14E+02**	**4.44E+02**	**2.00E+02**	
	±5.10E+02	±9.86E+02	**±3.40E+03**	**±6.32E+03**	**±2.13E-25**	



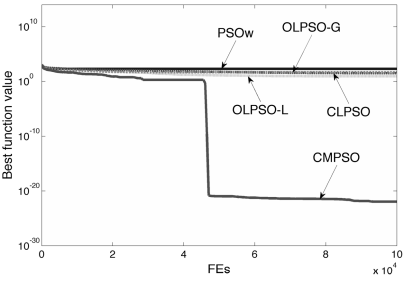
(a) Benchmark function $\boldsymbol{F}_1$



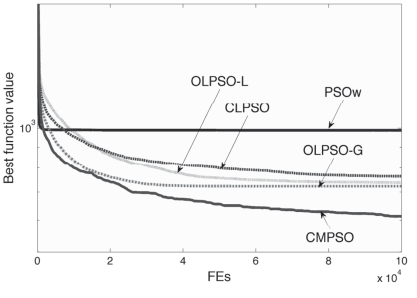
(b) Benchmark function $\boldsymbol{F}_8$



(c) Benchmark function $\boldsymbol{F}_{14}$



(d) Benchmark function $\boldsymbol{F}_{20}$



(e) Benchmark function $\boldsymbol{F}_{27}$

Fig. 3. Average convergence traces over 50 runs

4 Conclusion

A novel coevolutionary memetic particle swarm optimizer (CMPSO) was proposed in this paper. CMPSO defines a single solution position **P** and five coevolutionary memes (CMs) characterized with different local search strategies and computational budget are introduced in optimized the solution. CMPSO co-evolves both CMs and **P** in the way that CMs are updated like PSO particles but with specific rules, and **P** is updated by undergoing the individual refinement defined in each CM. By performing all the individual learning processes on the same solution vector **P**, the confliction of

Effectiveness and *Attainableness* is alleviated. Experimental results on 7 unimodal and 22 multimodal benchmark functions demonstrate that the CMPSO attains better performance than other state-of-the-art PSO variances. CMPSO is capable of attaining higher convergence speed wile preventing the premature convergence.

References

1. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: IEEE International Conference on Neural Network, Australia, pp. 1942–1948 (1995)
2. Liang, J.J., Qin, A.K., Suganthan, P.N., et al.: Comprehensive Learning Particle Swarm Optimizer for Global Optimization of Multimodal Functions. IEEE Transactions on Evolutionary Computation 10(3), 281–295 (2006)
3. Zhan, Z.H., Zhang, J., Li, Y., et al.: Orthogonal Learning Particle Swarm Optimization. IEEE Transactions on Evolutionary Computation 15(6), 832–847 (2010)
4. De Oca, M.A.M., Aydin, D., Stützle, T.: An Incremental Particle Swarm for Large-Scale Optimization Problems: An Example of Tuning-in-the-loop (Re)Design of Optimization Algorithms. Soft Computing 15, 2233–2255 (2011)
5. Yang, Z.Y., Tang, K., Yao, X.: Scalability of Generalized Adaptive Differential Evolution for Large-Scale Continuous Optimization. Soft Computing 15, 2141–2155 (2011)
6. Davidon, W.: Variable Metric Method for Minimization. SIAM Journal on Optimization 1(1), 1–17 (1991)
7. Schwefel, H.P.: Evolution and Optimum Seeking: the Sixth Generation. John Wiley & Sons, USA (1993)
8. Gao, Y., Xie, S.L.: Chaos Particle Swarm Optimization Algorithm. Computer Science 31(8), 13–15 (2004)
9. Enriquez, N., Sabot, C.: Random Walks in a Dirichlet Environment. Electronic Journal of Probability 11(31), 802–817 (2006)
10. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by Simulated Annealing. Science 220(4598), 671–680 (1983)
11. Nguyen, Q.H., Ong, Y.S., Lim, M.H.: Non-genetic Transmission of Memes by Diffusion. In: Annual Conference on Genetic and Evolutionary Computation, USA, pp. 1017–1024 (2008)
12. Dawkins, R.: The Selfish Gene, 2nd edn. Oxford University Press, UK (1989)
13. Yao, X., Liu, Y., Lin, G.M.: Evolutionary Programming Made Faster. IEEE Transactions on Evolutionary Computation 3(2), 82–102 (1999)
14. Liang, J.J., Suganthan, P.N., Deb, K.: Novel Composition Test Functions for Numerical Global Optimization. In: IEEE Swarm Intelligence Symposium, USA, pp. 68–75 (2005)
15. Suganthan, P.N., Hansen, N., Liang, J.J., et al.: Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-parameter Optimization. In: IEEE Congress on Evolutionary Computation, UK (2005)

Handling Multi-optimization with Gender-Hierarchy Based Particle Swarm Optimizer

Wei Wei[1], Weihui Zhang[2], Yuan Jiang[1], and Hao Li[2]

[1] China Jiliang University, Hangzhou 310032, Zhejiang, China
weiw@cjlu.edu.cn
[2] Department of Computer Science of Zhejiang University of Technology, Hangzhou 310024, Zhejiang, China
zwh@zjut.edu.cn

Abstract. In this study, we present a novel particle swarm optimizer, called Gender-Hierarchy Based Particle Swarm Optimizer (GH-PSO), to handle multi-objective optimization problems. By employing the concepts of gender and hierarchy to particles, both the exploration ability and the exploitation skill are extended. In order to maintain an uniform distribution of non-dominated solutions, a novel proposal, called Rectilinear Distance based Selection and Replacement (RDSR), is also proposed. The proposed algorithm is validated by using several benchmark functions and metrics. The results show that the proposed algorithm outperforms over MOPSO, NSGA-II and PAES-II.

Keywords: Gender, Hierarchy, Particle Swarm Optimizer, Multi-objective Optimization, Rectilinear Distance.

1 Introduction

As one of the most successful paradigms of Swarm Intelligence (SI), Particle Swarm Optimization (PSO), has been used widely in many practical optimization scenarios since it was proposed firstly in 1995, and the extended PSOs also show their competitive abilities on multi-objective optimization problems (MOP)[1,2,3,4,5,6,7]. In this paper, we will present a proposal, called GH-PSO, which is designed to handle multi-objective optimization problems with a small iteration because, in practice, the objective function evaluations are a time-consuming and computational resource consuming task. We revise and improve our previous study which was reported in [8] to deal with multi-objective optimization problems. A repository (also called Archive) incorporates to keep the non-dominated solutions which are found in runs, and in order to maintain an uniform distributed non-dominated solutions, a novel mechanism for the most crowded non-dominated solutions replacement, called rectilinear distance (also be named as Manhattan distance) based replacement approach, is also presented in this study.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 101–108, 2012.

GH-PSO is validated using several test functions and compared against NSGA-II, PAES-II and MOPSO on multi-objective optimization since these algorithms represent the state-of-the-art of the PSOs.

The remainder of this paper is organized as follows. In Sect. 2, we review some basic concepts of PSO. And, we present our proposed algorithm in Sect. 3. In Sect. 4 and Sect. 5, the experimental results, discussions and conclusions are presented respectively.

2 Basic Concepts

2.1 A Brief Description of Particle Swarm Optimization

A basic version PSO, gBest PSO, it was proposed firstly by Kennedy and Eberhart with the following equations:

$$v_i^{k+1} = \omega * v_i^k + \phi_1 * (g^k - x_i^k) + \phi_2 * (l_i^k - x_i^k) \tag{1}$$

$$x_i^{k+1} = x_i^k + v_i^{k+1} \tag{2}$$

where

$$\phi_1 = \gamma_1 * a_g, \phi_2 = \gamma_2 * a_t, \gamma_1, \gamma_2 \rightarrow U(0,1) \in R \tag{3}$$

$$V = [\nu_1, \nu_2, \ldots, \nu_n], X = [x_1, x_2, \ldots, x_n \tag{4}$$

Concerning about the gBEST PSO, we observer that it consists of two components: social part $(g^k - x_i^k)$ and recognition part $(l_i^k - x_i^k)$. In the social part, all particles learn from the best particle, and thus shift to a fittest position, which is a metaphor as human beings do. In the recognition part, the particle learns from its own best previous position, thus the particle moves towards the best position experienced so far in the search space. It is a self-study procedure. After several iterations, then, the PSO obtains the optimal solutions or exits.

3 Description of the Proposed Approach

3.1 Gender-Hierarchy Based PSO

Before presenting our algorithm, some terms used by our algorithm will be stated as following. Gender usually is used to describe the exploration and exploitation ability [8]. In the initial phase, each particle is randomly assigned a gender. It is given to indicate search ability of the female and the male respectively. Hierarchy, In human beings and other sophisticated societies, they all have a hierarchy. the hierarchy depicts the social rank, the more higher rank it have and the more power is obtained in society. Each particle has a hierarchical level, also called grade. In our proposed algorithm, it can be observed that the social part and the recognition part are both modified in order to accelerate the convergence by multiplying a new coefficient, $\omega_{\texttt{particle\_i}}$. Thereby, we can choose the different values of c_1 and c_2 to keep the balance between the impact of gender and the impact of grade.

3.2 Gender-Hierarchy Based PSO for Multi-objective Optimization

Using GH-PSO to handle a multi-objective optimization problem, an external repository is incorporated to maintain the non-dominated solutions found in generations. In order to maintain an uniform distribution non-dominated solution set, a metric, called rectilinear distance (usually be called as Manhattan distance), is employed to GH-PSO to maintain the spread of non-dominated solutions. Now that, we will present our proposed algorithm Gender-Hierarchy Based PSO for Multi-Objective Optimization (GH-PSO) in details. In fact, we also can apply the punishment policy [8], which is a different proposal used here, to the particles when the particles move out of the search boundaries.

A. The Proposed Algorithm: The algorithm of GH-PSO is the following.

1. Initialize the populations of GH-PSO, Pop.
2. Initialize the personal best of each particle in Pop.
3. For i = 0 to MAX_PARTICLES_NUMBER
4. Pbests[i] = Pop[i]
5. End For
6. Evaluate the objective function values of the particles in Pop.
7. Maintain the particles which are the non-dominated found in Pop and store them into the external repository REP.
8. While not finished
9. For each particle in Pop
10. Designate a global guider GBEST randomly from the external repository.
11. Calculate $\omega_{particle}$, ω_{si}, and $X_{average}$.
12. Update the velocity and position of particle.
13. If a particle flies out of the boundaries the it is reset to the range of [-MAX_POS,MAX_POS] again and its velocity is set to the oppsite direction by multiplying -1 so that the particle searches the oppsite space.
14. End For
15. Use RDSR approach to update the non-dominated solutions in external repository.
16. Update the personal best of each particle in Pop with PBESTS[i] = Pop[i] if the current PBESTS dominates the position stored in its memory.
17. If the terminal conditions are satisfied, then set the finished condition to true.
18. End of While.
19. Output the optimal solution, Non-dominated solutions found.

B. External Repository: Compared to single objective optimization problems, one of the main goals in MOP is to obtain a well distributed solutions throughout the whole Pareto front. Therefore, a mechanism, called Rectilinear Distance based Selection and Replacement (RDSR), is employed to obtain a well distributed non-dominated solutions. The metric, rectilinear distance, is

used to describe the sum of the (absolute) differences of their coordinates between two points. The definition of rectilinear distance is presented formally as: $RD(P,Q) = \|P - Q\|_1 = \sum_{i=1}^{n} |p_i - q_i|$. where, P $=(p_1, p_2, \cdots, p_n)$ and Q$=(q_1, q_2, \cdots, q_n)$. And, from the definition of spacing, we observe that the spacing metric uses statistical rectilinear distance to evaluate the distribution of the non-dominated solutions. Therefore, we propose a novel approach, which is based on the metric of rectilinear distance, so as to obtain a well distributed non-dominated solutions throughout the whole Pareto front.

1. If $Reposity.size < MAX\_REPOSITY\_SIZE$ then
2. Add the new found non-dominated solution to repository.
3. Else
4. For i=0 to num_of_nondominated_solutions.
5. Calculate the rectilinear distance, RD_i
6. End for
7. Sort the rectilinear distance, RD_i, in ascending order.
8. Calculate the rectilinear distance of the new found non-dominated solution, RD_{new}.
9. Select a non-dominated solution which has the smallest value of rectilinear distance in repository, $RD_{smallest}$
10. If $RD_{new} > RD_{smallest}$ then
11. Replace a non-dominated solution which has the smallest value of rectilinear distance in repository with the new found non-dominated solution.
12. End If
13. End If

4 Experiments and Discussions

4.1 Experimental Settings

In the following experiments, NSGA-II, PAES-II and MOPSO, are employed so as to assess and validate the performance of GH-PSO [2,11,12]. The crossover rate, mutation rate and the depth of adaptive grid adopted the same value used in [2,11,12]. All of the algorithms maintained a population size of 100 and an archive size of 100. In all the following experiments, we report the results obtained from conducting 30 independent runs of each algorithm compared. The algorithms all conducted on Freebsd 7.0 system platform with GUN C++ compiler 4.2.1 and 1G RAM.

4.2 Performance Metrics

When we optimize a multi-objective problem, three aspects of the algorithm should be take into consideration. Firstly, the distance between the true Pareto front and the front generated by the proposed algorithm; secondly, the distribution of the front we produced; lastly, the number of the obtained non-dominated solutions, which does not belong to the true Pareto set. Therefore, we choose Generation Distance (GD), Spacing (SP), and Error ration (ER) as our performance metrics.

4.3 Experimental Results

Test Function 1: SCH2. Test function SCH2 is proposed by Schaffer [9]. SCH2 has two objective functions, and the decision variable of SCH2 is at the range of [-5,10]. The Pareto front of SCH2 is a dis-continuous front, and its contain two branches. One branch the decision variables is located in the range of [1, 2]; the another branch the decision variable is at the interval of [4, 5].

$$SCH2 : min \begin{cases} f_1(X) = g(X) \\ f_2(X) = (x-5)^2, otherwise \end{cases} \tag{5}$$

where, $g(X) = -x \mid (x \leq 1), x-2 \mid (1 < x \leq 3), 4-x \mid (3 < x \leq 4), x-4 \mid (x > 4)$

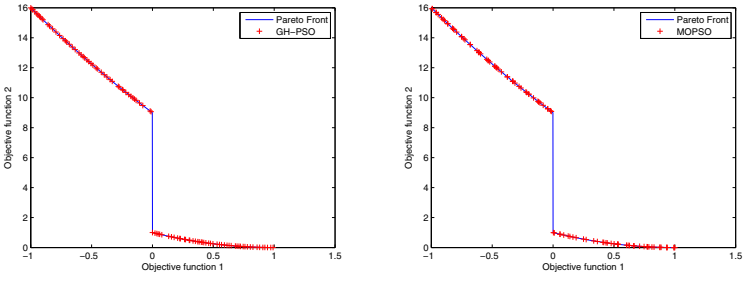


Fig. 1. Pareto fronts for the SCH2 test function

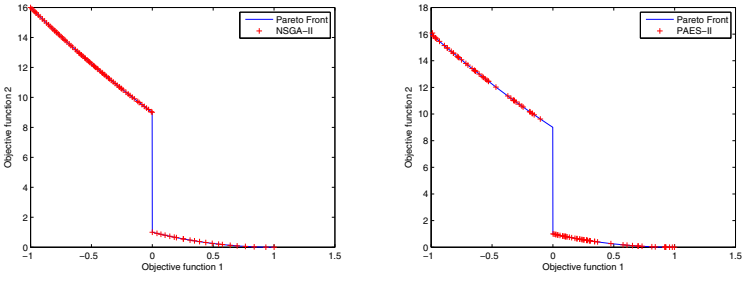


Fig. 2. Pareto fronts for the SCH2 test function

In this case, with respect to GD, the SP and the ER under consideration, the GH-PSO runs the first place. The results of error ratio shown in Tab. 1 state that the GH-PSO out-performs all others in three algorithms on spacing metric because the GH-PSO approximates closer to the true Pareto front than that of the others did. Although the GH-PSO falls slightly behind the MOPSO on the standard deviation metric, in Tab. 2, the GH-PSO is in leading place with respect to the average generation distance.

Table 1. The results of the ER of the SCH2

ER	GH-MOPS	MOPSO	NSGA-II	PAES-II
Beset	1.00000E-02	1.00000E-02	3.00000E-02	7.14286E-02
Worst	1.00000E-01	7.00000E-02	1.00000E-01	2.30769E-01
Average	5.96667E-02	5.96667E-02	6.36667E-02	1.13919E-01
Median	6.00000E-02	7.00000E-02	6.00000E-02	7.14286E-02
Std. Dev.	2.54929E-02	1.51621E-02	1.88827E-02	7.04631E-02

Table 2. The results of the GD of the SCH2

GD	GH-MOPS	MOPSO	NSGA-II	PAES-II
Beset	2.10338E-04	2.37592E-04	2.79421E-04	1.69986E-04
Worst	3.04970E-04	2.46704E-04	3.50361E-04	6.47535E-03
Average	2.39720E-04	2.44127E-04	3.05900E-04	2.97332E-03
Median	2.37280E-04	2.45577E-04	3.02141E-04	1.69986E-03
Std. Dev.	3.83869E-06	3.63869E-06	1.62701E-05	2.11180E-03

Table 3. The results of the SP of the SCH2

SP	GH-MOPS	MOPSO	NSGA-II	PAES-II
Beset	2.04568E-02	5.44174E-02	2.24919E-02	5.77608E-01
Worst	3.06099E-02	5.76094E-02	3.10069E-02	7.42050E-01
Average	2.59036E-02	5.63695E-02	2.72376E-02	6.21459E-01
Median	2.56108E-02	5.62590E-02	2.75936E-02	5.77608E-01
Std. Dev.	2.75954E-03	1.22825E-03	2.15617E-03	7.27189E-02

G. Test Function 2: KUR. Our second multi-objective test function was proposed by Kursawe [10]. It has three decision variables and two objective functions, all of the decision variables are at the interval of [-5, 5]. The true Pareto front of KUR is a dis-continuous and non-convex line.

$$KUR: Min \begin{cases} f_1(X) = \sum_{i=1}^{n-1}(-10exp(-0.2\sqrt{x_i^2 + x_{i+1}^2})) \\ f_2(X) = \sum_{i=1}^{n}(|x_i|^{0.8} + 5sin(x_i)^3), n = 3 \end{cases} \tag{6}$$

In Fig. 3 and Fig. 4, the graphical results state that only the GH-PSO and the MOPSO cover the whole true Pareto front. The Pareto front produced by the NSGA-II only is a part of the true Pareto front. The Pareto front obtained by the NSGA-II reflects that the NSGA-II only explores the center part of the search space, the margin of the search space does not be explored. The PAES-II meets the same issue on KUR function. Our statements are confirmed by the numerical results shown in Tab. 4 and Tab. 5. With respect to ER and GD, the GH-PSO is in the first place followed by the MOPSO. Although the NSGA-II also has an uniform Pareto front distribution and a smaller generation distance, in fact, the Pareto front produced by the NSGA-II does not distribute throughout the entire true Pareto front. Therefore, we draw the conclusion that the GH-PSO has the best performance on KUR function.

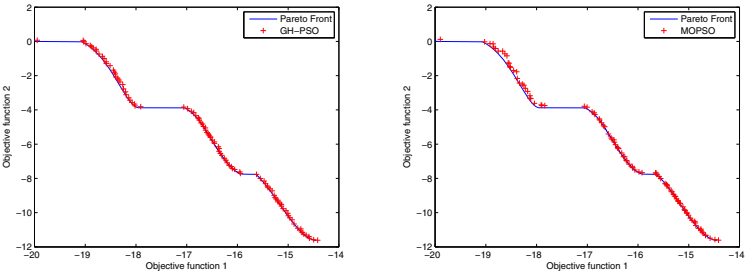


Fig. 3. Pareto fronts for the KUR test function

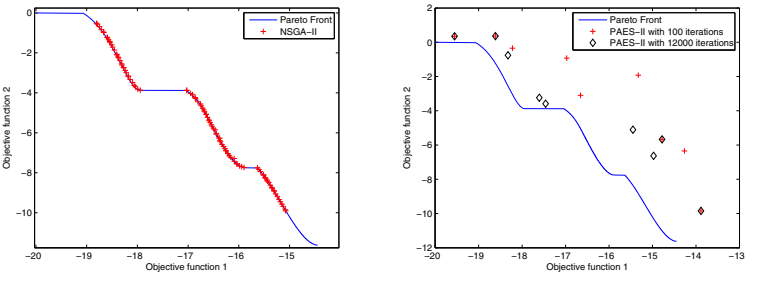


Fig. 4. Pareto fronts for the KUR test function

Table 4. The results of the ER of the KUR

ER	GH-MOPS	MOPSO	NSGA-II	PAES-II
Beset	7.90000E-01	7.60000E-01	6.40000E-01	1.00000E+00
Worst	9.50000E-01	9.00000E-01	8.40000E-01	1.00000E+00
Average	8.84646E-01	8.33000E-01	7.51667E-01	1.00000E+00
Median	8.90000E-01	8.30000E-01	7.600000E-01	1.00000E+00
Std. Dev.	3.66424E-02	3.03480E-02	5.45334E-02	0.00000E+00

Table 5. The results of the GD of the KUR

GD	GH-MOPS	MOPSO	NSGA-II	PAES-II
Beset	2.79945E-03	3.15058E-03	1.00850E-03	1.74499E+00
Worst	3.77302E-03	7.39038E-03	8.29315E-03	1.92665E+00
Average	3.22489E-03	5.05637E-03	2.45549E-03	1.75104E+00
Median	3.19809E-03	4.97383E-03	1.92209E-03	1.74499E+00
Std. Dev.	2.19689E-04	1.23526E-03	1.45383E-03	3.26099E-02

Table 6. The results of the SP of the KUR

SP	GH-MOPS	MOPSO	NSGA-II	PAES-II
Beset	7.90221E-02	5.97845E-03	3.85658E-02	7.64514E-02
Worst	1.04721E-01	1.18247E-01	2.03283E-02	2.81594E-01
Average	8.63365E-02	8.83308E-02	7.43422E-02	8.32895E-02
Median	8.61962E-02	9.03928E-02	5.51238E-02	7.64514E-02
Std. Dev.	6.67407E-03	1.93582E-02	4.07817E-2	3.68243E-02

5 Conclusions

In this study, we present a approach, called GH-PSO which is an extended version of our previous work, to deal with MOP with a small iteration number, and the proposed algorithm was validated on benchmark functions. The GH-PSO has competitive performance with respective to the well-known algorithms, MOPSO, NSGA-II and PAES-II. We studies also indicate that the proposed approach is suit for the resource limited circumstance compared to NSGA-II and PAES-II.

Next, to solve constrained multi-objective optimization problems and dynamic multi-objective optimization problems and their parallelization are underway.

References

1. Moore, J., Chapman, R., Dozier, G.: Multiobjective Particle Swarm Optimization. In: ACM-SE 38: Proceedings of the 38th Annual on Southeast Regional Conference, pp. 56–57 (2000)
2. Coello Coello, C.A., Pulido, G.T., Lechuga, M.S.: Handling Multiple Objectives With Particle Swarm Optimization. IEEE Transactions on Evolutionary Computation 8(3), 256–279 (2004)
3. Moore, J., Chapman, R.: Application of Particle Swarm to Multiobjective Optimization. Dept. of Computer Science Software Engineering, Auburn University (1999)
4. Ray, T., Liew, K.M.: A swarm metaphor for multiobjective design optimization. Engineering Optimization 34(2), 141–153 (2002)
5. Hu, X., Eberhart, R.: Multiobjective optimization using dynamic neighborhood particle swarm optimization. In: Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002), vol. 2, pp. 1677–1681 (2002)
6. Li, X.: A Non-dominated Sorting Particle Swarm Optimizer for Multiobjective Optimization. In: Cantú-Paz, E., Foster, J.A., Deb, K., Davis, L., Roy, R., O'Reilly, U.-M., Beyer, H.-G., Kendall, G., Wilson, S.W., Harman, M., Wegener, J., Dasgupta, D., Potter, M.A., Schultz, A., Dowsland, K.A., Jonoska, N., Miller, J., Standish, R.K. (eds.) GECCO 2003. LNCS, vol. 2723, pp. 37–48. Springer, Heidelberg (2003)
7. Parsopoulos, K.E., Vrahatis, M.N.: Particle swarm optimization method in multiobjective problems. In: Proceedings of the 2002 ACM Symposium on Applied Computing, pp. 603–607 (2002)
8. Gao, J., Li, H., Hu, L.: Gender-Hierarchy Particle Swarm Optimizer Based on Punishment. In: Tan, Y., Shi, Y., Tan, K.C. (eds.) ICSI 2010. LNCS, vol. 6145, pp. 94–101. Springer, Heidelberg (2010)
9. Schaffer, J.D.: Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. Ph.D. thesis, Vanderbilt University, Nashville, Tennessee (1984)
10. Kursawe, F.: A Variant of Evolution Strategies for Vector Optimization. In: Schwefel, H.-P., Männer, R. (eds.) PPSN 1990. LNCS, vol. 496, pp. 193–197. Springer, Heidelberg (1991)
11. Knowles, J.D., Corne, D.W.: Approximating the Non-dominated Front using the Pareto Archived Evolution Strategy. Evolutionary Computation 8(2), 149–172 (2000)
12. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6, 182–197 (2002)

The Comparative Study of Different Number of Particles in Clustering Based on Two-Layer Particle Swarm Optimization*

Guoliang Huang, Xinling Shi, and Zhenzhou An

School of Information Science and Engineering, Yunnan University, Kunming 650091, China
hgl_wan.2008@163.com

Abstract. To study how the different number of particles in clustering affect the performance of two-layer particle swarm optimization (TLPSO) that set the global best location in each swarm of the bottom layer to be the position of the particle in the swarm of the top layer, fourteen configurations of the different number of particles are compared. Fourteen benchmark functions, being in seven types with different circumstance, are used in the experiments. The experiments show that the searching ability of the algorithms is related to the number of particles in clustering, which is better with the number of particles transforming from as little as possible to as much as possible in each swarm of the bottom layer when the function dimension is increasing from low to high.

Keywords: Particle swarm optimization, hierarchy, cluster.

1 Introduction

The particle swarm optimization (PSO) algorithm is based on the evolutionary computation technique. And it has been used increasingly as a novel technique for solving complex optimization problems. Many researchers have expanded on the original ideas of PSO, and some improving approaches such as the idea of hierarchy and cluster have been reported. In [1], a dynamically changing branching degree of the tree topology was proposed for solving intractable large parameter optimization problems. In [2], the particles have been clustered in each iteration and the centroid of the cluster was used as an attractor instead of using the position of a single individual. In [3], a new method named MSSE-PSO (master–slave swarms shuffling evolution algorithm based on PSO) was proposed. In [4], the population was partitioned into several sub-swarms, each of which was made to evolve based on the PSO. In [5], the PSO approach that used an adaptive variable population size and periodic partial increasing or declining individuals in the form of ladder function was proposed so that overall performance of the PSO was enhanced. In [6], a two-layer particle swarm optimization (TLPSO) was proposed for unconstrained optimization problems.

* This paper is supported by the National Natural Science Foundation of China No. 61062005.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 109–115, 2012.

The main content of this article discusses the influence on the ability to search function optimization among the configurations of the different number of particles in clustering, and looks for the general rule of the different configurations compared in various benchmark function with all kinds of circumstance. In this article, the main idea of [6] has been used in the updated algorithms for the comparison. Through the experiment, it has come to the conclusion that a good efficiency of searching ability is related to the number of particles transforming from as little as possible to as much as possible in each swarm of the bottom layer when the function dimension is increasing from low to high.

The rest of this article is organized as follows. Section 2 describes the main idea of [6] and three classifications about the fourteen benchmark functions quoted from [7]. In Section 3, the basic process of the updated algorithms idea is presented and the seven more detailed classifications for the function are described for the following main content. In section 4, fourteen configurations of the different number of particles in clustering are compared in the benchmark functions existing in the seven types. Finally, section 5 draws conclusions about the comparison among the fourteen configurations of the different number of particles testing in the seven types.

2 TLPSO and the Benchmark Function

In [6], a two-layer particle swarm optimization (TLPSO) was proposed for unconstrained optimization problems. In the TLPSO approach, there were two layers of the structure: top layer and bottom layer, and *M* swarms of particles and one swarm of particles were generated in the two layers, respectively. Each global best location in each swarm of the bottom layer was set to be the position of the particles in the swarm of the top layer so that the global best location in the swarm of the top layer influenced the particles of each swarm in the bottom layer indirectly. Furthermore, a mutation operation was added into the particles of each swarm in the bottom layer. Consequently, the diversity of the population in the TLPSO increased so that the TLPSO has the ability to avoid trapping into the local optimum.

In [7], fourteen benchmark functions ($f_1 - f_{14}$) were divided into three types, which were simple unimodal function from f_1 to f_3, highly complex multimodal function with many local minima from f_4 to f_9, and multimodal function with few local minima from f_{10} to f_{14}.

3 Algorithms and Classification

3.1 Basic Process of the Updated Algorithms Idea

In the updated algorithms, there will be two layers: the top layer and the bottom layer. The total number of particles is set to be one hundred and twenty, and there will be fourteen different classifications about the total number of particles. The classifications with the fourteen different configurations are shown in Table 1, where

M means the total number of swarms in the bottom layer and *N* stands for the average number of particles in clustering.

Table 1. The number of particles in clustering

M	2	3	4	5	6	8	10	12	15	20	24	30	40	60
N	60	40	30	24	20	15	12	10	8	6	5	4	3	2

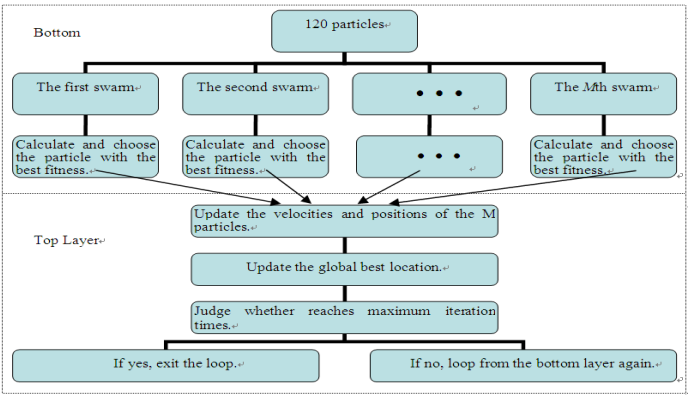


Fig. 1. The flow of the algorithmic thinking

The total number of swarms (*M*) is corresponding to the average number of particles (*N*) in clustering in the bottom layer. According to the fitness contrasting among the particles in the *d*th swarm of the bottom layer, *d*=1,2,…,*M*, the global best location of the *d*th swarm is determined. Then the global best location of each swarm is set to be in the top layer, that is to say the number of particles in the top layer has also been determined, which is *M*. And then, the global best location of the swarm in the top layer will be determined according to the fitness contrasting among *M* particles in the top layer. The specific flow of the algorithmic thinking is shown in Fig.1.

3.2 Computation Cost Analysis among the Fourteen Configurations of the Number of Particles in Clustering

Suppose the computation cost of one particle in the algorithms is *c*, then the total computation cost of the algorithms for one generation is *MNc*+*Mc*, where *MNc* stands for the computation cost of the bottom layer and *Mc* stands for the computation cost of the top layer. Therefore, it can be obtained from the analysis that the computation cost is increasing along with the increasing of the number of swarms, in other words, it corresponds to the decreasing of the number of particles in clustering.

3.3 Detailed Classification and Main Work

In this section, the three types about the fourteen benchmark functions [7] will be expanded to seven types:

- type 1. Unimodal function in two dimensions;
- type 2. Unimodal function in ten dimensions;
- type 3. Unimodal function in one hundred dimensions;
- type 4. Highly complex multimodal function with many local minima in two dimensions;
- type 5. Highly complex multimodal function with many local minima in ten dimensions;
- type 6. Highly complex multimodal function with many local minima in one hundred dimensions;
- type 7. Multimodal function with few local minima in two or four dimensions.

The main content of this article discusses the influence on the ability to search function optimization among the fourteen configurations of the different number of particles in clustering, and looks for the general rule of the fourteen configurations compared in the seven types. The next section will emulate and present the results of the comparison.

4 Experiments

In this section, seven types of the functions are employed to examine the efficiency of searching function optimization with the fourteen configurations of the different number of particles shown in Table 1, respectively. The maximum iteration times and operation cycle times for each function in different dimensions are listed in Table 2.

The simulation results about the fourteen configurations of the different number of particles in clustering testing in the seven types are shown from Fig.2 to Fig.8, respectively. In the figures, the horizontal coordinate stands for the different number of particles in clustering and the vertical coordinate stands for the average value of all the global optimum values except for the maximum and the minimum values.

According to the results comparison shown from Fig.2 to Fig.8, respectively, it comes to the conclusion shown as follows:

In the circumstance of unimodal function, Fig.2 reveals that the searching ability will become better with the less number of particles in clustering in two dimensions and Fig.4 reveals the opposite that the more number of particles in clustering in one hundred dimensions shows better efficiency.

In the circumstance of multimodal function, Fig.5 and Fig.8 reveal the same results that the better searching ability is corresponding to the less number of particles in clustering in two or four dimensions, and Fig.7 reveals the same result as Fig.4 in one hundred dimensions.

As in Fig.3 and Fig.6 simulated from unimodal and multimodal function in ten dimensions, there will be a conjecture that the searching ability will become better

along with the number of particles in clustering transforming from as little as possible to as much as possible.

Table 2. Iteration times and operation cycle times for each function in different dimensions

2 or 4 dimensions			10 dimensions			100 dimensions		
Func-tion	Iteratio-n times	Operatio-n cycle times	Func-tion	Iteratio-n times	Operatio-n cycle times	Func-tion	Iteratio-n times	Operatio-n cycle times
f_1	10	30000	f_1	20	10000	f_1	200	30
	10	30000		80	10000		15000	30
f_2	10	30000	f_2	1000	500	f_2	20000	30
f_3	15	30000	f_3	2200	200	f_3	100000	30
	10	30000		100	10000		15000	30
f_4	10	30000	f_4	20	10000	f_4	300	30
f_5	10	30000	f_5	40	10000	f_5	200	30
	5	30000		20	10000		100	30
f_6	5	30000	f_6	20	10000	f_6	200	30
f_7	5	30000	f_7			f_7		
	5	30000						
f_8	10	30000	f_8			f_8		
f_9	10	30000	f_9			f_9		
f_{10}	10	30000						
f_{11}								
f_{12}								
f_{13}								
f_{14}								

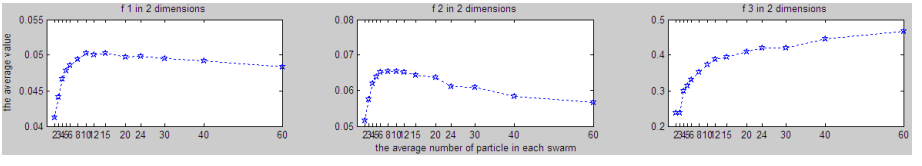


Fig. 2. Fourteen configurations of the different number of particles compared in type 1

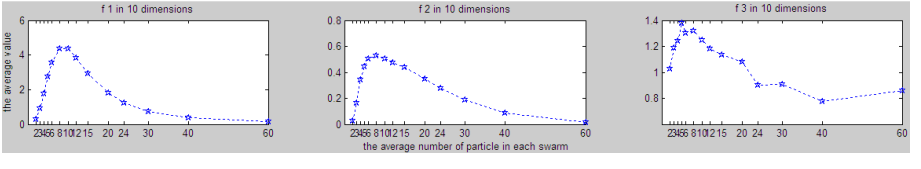


Fig. 3. Fourteen configurations of the different number of particles compared in type 2

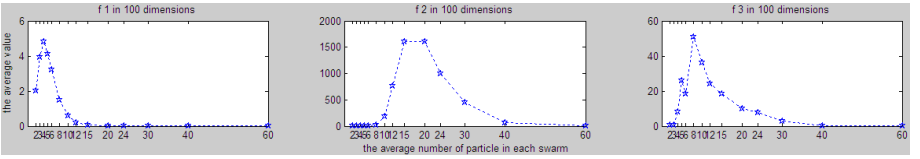


Fig. 4. Fourteen configurations of the different number of particles compared in type 3

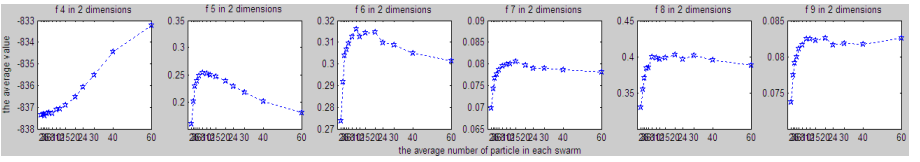


Fig. 5. Fourteen configurations of the different number of particles compared in type 4

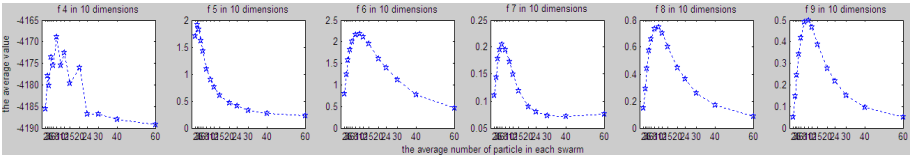


Fig. 6. Fourteen configurations of the different number of particles compared in type 5

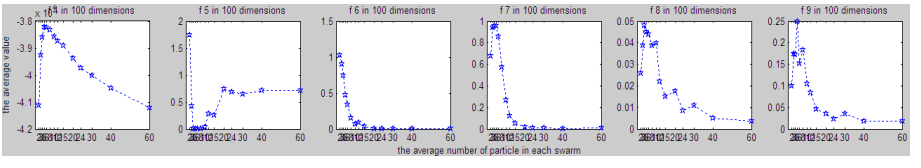


Fig. 7. Fourteen configurations of the different number of particles compared in type 6

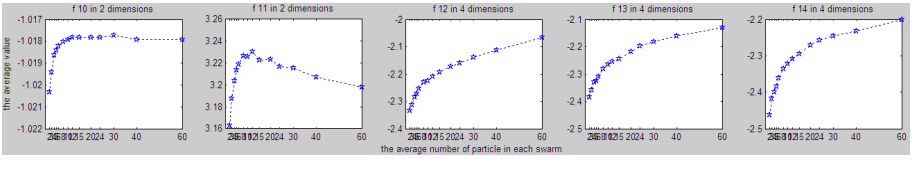


Fig. 8. Fourteen configurations of the different number of particles compared in type 7

5 Conclusion

In this article, fourteen configurations of the different number of particles in clustering have been compared in the fourteen benchmark functions existing in different circumstance, respectively. According to the simulation results of the seven types, it can reach three general conclusions listed as follows:

1. The less number of particles in clustering, the better searching ability for the function in low dimension.

2. The searching ability is better with the number of the particles transforming from as little as possible to as much as possible in clustering when the function dimension is increasing from low to high.
3. The more number of particles in clustering, the better searching ability for the function in high dimension.

References

1. Janson, S., Middendorf, M.: A hierarchical particle swarm optimizer and its adaptive variant. IEEE Transactions on Systems, Man, and Cybernetics- Part B: Cybernetics 35, 1272–1282 (2005)
2. Kennedy, J.: Stereotyping: improving particle swarm performance with cluster analysis. In: Proceedings of the 2000 Congress on Evolutionary Computation, vol. 2, pp. 1507–1512 (2000)
3. Jiang, Y., Liu, C.M., Huang, C.C., Wu, X.N.: Improved particle swarm algorithm for hydrological parameter optimization. App. Math. Comp. 217, 3207–3215 (2010)
4. Jiang, Y., Hu, T., Huang, C.C., Wu, X.: An improved particle swarm optimization algorithm. App. Math. Comp. 193, 231–239 (2007)
5. Chen, D.B., Zhao, C.X.: Particle swarm optimization with adaptive population size and its application. App. Soft. Comp. 9, 39–48 (2009)
6. Chen, C.C.: Two-layer particle swarm optimization for unconstrained optimization problems. App. Soft. Comp. 11, 295–304 (2011)
7. Bratton, D., Kennedy, J.: Defining a Standard for Particle Swarm Optimization. In: Proceedings of the 2007 IEEE Swarm Intelligence Symposium, pp. 120–127. IEEE Press, Honolulu (2007)

Improved Particle Swarm Optimization with Wavelet-Based Mutation Operation

Yubo Tian, Donghui Gao, and Xiaolong Li

School of Electronics and Information, Jiangsu University of Science and Technology, Zhenjiang Jiangsu 212003, China
{Tianyubo,lixiaolong}@just.edu.cn, 476949108@qq.com

Abstract. An improved wavelet-based mutation particle swarm optimization (IWMPSO) algorithm is proposed in this paper in order to overcome the classic PSO's drawbacks such as the premature convergence and the low convergence speed. The IWMPSO introduces a wavelet-based mutation operator first and then the mutated particle replaces a selected particle with a small probability. The numerical experimental results on benchmark test functions show that the performance of the IWMPSO algorithm is superior to that of the other PSOs in references in terms of the convergence precision, convergence rate and stability. Moreover, a pattern synthesis of linear antennas array is implemented successfully using the algorithm. It further demonstrates the effectiveness of the IWMPSO algorithm.

Keywords: particle swarm optimization, wavelet mutation, synthesis.

1 Introduction

The PSO algorithm, which was firstly developed by Kennedy and Eberhart[1], is a kind of evolutionary computational technology based on intelligent behavior of organisms. Its basic idea is originally from artificial life and evolutionary computation[2][3]. As a kind of the general global research algorithm, the PSO is able to solve problems in real number field with few adjusted parameters. It is simple, easy to implement and efficient to compute. Therefore, the method has been widely used in many fields, such as neural network training, function optimization, fuzzy control system, etc [4]. Unlike the other heuristic techniques, the PSO has a flexible and well-balanced mechanism to enhance the global and local exploration abilities. At present, as a robust global optimal method, the PSO is also utilized in electromagnetic field [5][6][7], such as the design of absorbing material, antenna design, solving complex transcendental equations and so forth.

The PSO algorithm is based on the search of all particles and their own experience toward the direction of optimal solution. In the evolutionary process, especially in the late period of evolution, the convergence speed gets slow significantly due to the lack of particle diversity. Moreover, the optimization cannot continue effectively when the algorithm converges to a certain precision. Therefore, the accuracy of algorithm is low. In view of the fact, this study injects mutation operation into the PSO algorithm to improve its performance. Reference [8] proposed a hybrid wavelet-based PSO algorithm with mutation operation. The wavelet theory enhances the PSO in exploring

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 116–124, 2012.

the solution space more effectively. Numerical experiments show the proposed method significantly outperforms the existing methods in terms of convergence speed, solution quality and solution stability. This study proposes an IWMPSO algorithm with wavelet-based mutation operation based on the reference [8]. The IWMPSO applies a wavelet mutation operator to the current best solution of swarm.. After that, the best mutated solution replaces the selected particle with a small probability. The IWMPSO algorithm can increase the search probability of all particles, which means the algorithm can search in global area more effectively.

2 The Particle Swarm Optimization Algorithm

The PSO simulates the behaviors of bird flock[2][3]. Assuming a group of birds is randomly searching food in an area. There is only one piece of food being searched in the area. These birds do not know where the food is but they know how far the food is. So what is the best strategy to find the food? The effective one is to follow the bird that is nearest to the food. The PSO learns from the scenario and uses it to solve the optimization problems. In the PSO, each single solution is a "bird" in the search space. We call it "particle". All of particles have fitness values that are evaluated by the fitness function to be optimized, and have velocities that direct the flying of the particles. The particles are "flown" through the problem space by following the current optimum particles. The PSO is initialized with a group of random particles (solutions) first and then is searched for optima by updating generations. In every iteration, each particle is updated by the following two "the best" values. The first one is the best solution (fitness) it has achieved so far. This value is called pbest. The other that is tracked by the particle swarm optimizer is the best value, which is obtained so far by any particle in the swarm. This value is the global best and called gbest.

After finding the two best values, the particle updates its velocity and position with following formulas:

$$v_{i,d}^{k+1} = \omega \cdot v_{i,d}^{k} + c_1 \cdot rand(\ ) \cdot \left(pbest_{i,d}^{k} - x_{i,d}^{k} \right) + c_2 \cdot rand(\ ) \cdot \left(gbest_{d}^{k} - x_{i,d}^{k} \right) \quad (1)$$

$$x_{i,d}^{k+1} = x_{i,d}^{k} + v_{i,d}^{k+1} \quad (2)$$

where ω is inertia weight that controls the PSO's exploitation ability and exploration ability. c_1 and c_2 are learning factors, and usually $c_1 = c_2 = 2$. $rand(\ )$ is a random number between (0, 1). $v_{i,d}^{k}$ and $x_{i,d}^{k}$ are velocity and position of particle *i* in *d*th dimension and *k*th iteration, respectively, and they are limited to a scope.

The inertia weight in this work adopts linearly decreasing manner. Suppose the scope of inertia weight is $[\omega_{\min}, \omega_{\max}]$, the maximum iteration number is *num*. Therefore, the *k*th inertia weight is given by [9]

$$\omega_k = \omega_{\max} - \frac{\omega_{\max} - \omega_{\min}}{num} \times k \quad (3)$$

3 The PSO with Wavelet-Based Mutation Operation and Its Improvement

The reference [8] proposed a wavelet-based mutation particle swarm optimization (WMPSO) algorithm with a wavelet-based mutation operation that changes the selected particle slightly. Let the mutation probability of the mutation operator is $p \in [0,1]$, whose value is decided by the dimension of particles. Suppose $\boldsymbol{x}_i^k = \left(x_{i1}^k, x_{i2}^k, \cdots, x_{ij}^k\right)$ is the *i*th mutated particle in the *k*th iteration, x_{ij}^k is the particle of the *j*th dimension, and $px_{\max i}$ and $px_{\min i}$ are the upper limitation and lower limitation of the particle in the search area, respectively. Therefore, the mutation function is given by

$$mut(x_{ij}^k) = \begin{cases} x_{ij}^k + \sigma \times \left(px_{\max i} - x_{ij}^k\right), \sigma > 0 \\ x_{ij}^k + \sigma \times \left(x_{ij}^k - px_{\min i}\right), \sigma \le 0 \end{cases} \tag{4}$$

where $mut(x_{ij}^k)$ is x_{ij}^k after mutation operation, and σ is the wavelet value. Here, if the Morlet wavelet is considered, σ is defined as [10]

$$\sigma = \frac{1}{\sqrt{a}} e^{-\left(\frac{\phi}{a}\right)^2 / 2} \cos\left(5\left(\frac{\phi}{a}\right)\right) \tag{5}$$

where the range of ϕ is $[-2.5a, 2.5a]$. The computing function of *a* is given by

$$a = e^{-\ln(g) \times \left(1 - \frac{k}{num}\right)^{\xi_{\omega m}}} + \ln(g) \tag{6}$$

where $\xi_{\omega m}$ is shape parameter of the monotonic increasing function, *g* is the upper limitation of the parameter *a*, *k* is iteration number, and *num* is maximum iteration number.

Now, the wavelet mutation function in (4) is improved and defined by

$$mut(x_{iD}^k) = \begin{cases} gb_D^k + \sigma \times \left(px_{\max i} - gb_D^k\right), \sigma > 0 \\ gb_D^k + \sigma \times \left(gb_D^k - px_{\min i}\right), \sigma \le 0 \end{cases} \tag{7}$$

where gb_D^k is the current best in swarm in *k*th iteration, and the meanings of $px_{\max i}$, $px_{\min i}$, and σ are the same as (4).

The significant difference between function (7) and function (4) is that in function (4), the particle is selected to mutate according to certain probability. After mutation, the mutated particle may lie near the local maximum, and it cannot find the global

maximum. Whereas in function (7), the mutated particle is the current best. After mutation, using the mutated particle replaces a particle selected by certain probability. Therefore, the improved algorithm can search effectively near the best particle. This will lead the particle to the direction of global optimum, and decrease the probability of trapping in local optimum.

4 Numerical Experiments

4.1 Benchmark Test Functions

Some numerical experiments are taken in order to validate the performance of the the IWMPSO. The standard PSO hereafter is named as SPSO. Six benchmark test functions are used, involving Sphere, Rosenbrock, Schwefel, Rastrigrin, Griewank, and Ackly. The expressions of these functions are tabulated in Table 1.

Table 1. Benchmark test functions

Test function	Domain range	Optimal point
$f_1(x)=\sum_{i=1}^{30} x_i^2$	$-100 \le x_i \le 100$	$f_1(\mathbf{0})=0$
$f_2(x)=\sum_{i=1}^{30}\left[100(x_{i+1}-x_i^2)^2+(x_i-1)^2\right]$	$-2.048 \le x_i \le 2.048$	$f_2(\mathbf{1})=0$
$f_3(x)=\sum_{i=1}^{30}\lvert x_i\rvert+\prod_{i=1}^{30}\lvert x_i\rvert$	$-10 \le x_i \le 10$	$f_3(\mathbf{0})=0$
$f_4(x)=\sum_{i=1}^{30}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	$-50 \le x_i \le 50$	$f_4(\mathbf{0})=0$
$f_5(x)=\frac{1}{4000}\sum_{i=1}^{30} x_i^2-\prod_{i=1}^{30}\cos(\frac{x_i}{\sqrt{i}})+1$	$-600 \le x_i \le 600$	$f_5(\mathbf{0})=0$
$f_6(x)=-20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{30} x_i^2}\right)-\exp\left(\frac{1}{30}\sum_{i=1}^{30}\cos 2\pi x_i\right)+20+e$	$-32 \le x_i \le 32$	$f_6(\mathbf{0})=0$

Above six benchmark test functions can be divided into two categories. The first one is the category of the unimodal function including f_1, f_2 and f_3. The second one is the category of the multimodal function including f_4, f_5 and f_6.

4.2 Simulation Results and Analysis

The parameters in this study are as follows. The number of particles is 50, both the learning factors c_1 and c_2 are 2.05, the scope of linearly decreasing inertia weight is [1.2, 0.1], and the mutation probability is 0.1 in the IWMPSO and the WMPSO. In the IWMPSO, $\xi_{\omega m} = 0.5$ and $g = 1000$. And in the WMPSO, $\xi_{\omega m} = 5.0$ and $g = 100000$.

Table 2. Comparison between different PSO methods for benchmark test functions (Rk: 1- best, 3- worst)

Function		Index	IWMPSO	WMPSO	SPSO
Unimodal function	f_1	*Mn*	9.4668e-41	2.2628e-007	1.3578
		Bt	1.5110e-045	9.0028e-013	4.1700e-001
		Sd	1.0164e-041	3.0715e-008	6.8235e-002
		Rk	1	2	3
	f_2	*Mn*	9.5877e-013	3.6921e-001	63.9907
		Bt	5.2954e-016	5.3459e-004	35.1379
		Sd	1.0367e-013	4.9791e-001	1.0195
		Rk	1	2	3
	f_3	*Mn*	1.4485e-018	9.5425e-004	5.2183
		Bt	9.5482e-23	1.1099e-006	2.0290
		Sd	2.0635e-019	1.9951e-005	2.8460e-001
		Rk	1	2	3
Multimodal function	f_4	*Mn*	9.8528e-014	8.8337e-002	2.7785e+002
		Bt	0	4.1768e-005	1.6493e+002
		Sd	7.4109e-15	1.2961e-002	7.4000
		Rk	1	2	3
	f_5	*Mn*	3.9968e-004	3.9304e-003	86.9011
		Bt	0	1.0842e-006	72.4321
		Sd	5.7097e-005	7.0126e-004	1.14326
		Rk	1	2	3
	f_6	*Mn*	2.2204e-014	5.7506e-004	16.9971
		Bt	8.8818e-016	1.1266e-006	1.7414
		Sd	3.0452e-015	7.8779e-005	2.1099
		Rk	1	2	3

The numerical experiments are performed according to above-mentioned parameters. 50 times runs are taken in each function. For all of the benchmark test functions, the computing results are listed in Table 2, where *Mn* presents the best mean fitness in the 50 times runs, *Bt* presents the best fitness in the 50 times runs, and *Sd* presents standard deviation of fitness function in the 50 times runs. *Mn*, *Bt*, and *Sd* are used to show the performance of the algorithms.

From Table 2, it can be seen that to all of the six benchmark test functions, *Mn*, *Bt*, and *Sd* of the IWMPSO are superior to that of the WMPSO and the SPSO. Furthermore, to all of the test functions, the optimal results of the IWMPSO are close to the global optima. Compared to the WMPSO and the SPSO, the IWMPSO leads to the effective evolution direction. Its capability to leave the local optimum is good. Therefore, the convergence and stability of the IWMPSO are better than that of the WMPSO and the SPSO.

Figure 1 shows the mean best fitness in the 50 times runs of different PSO algorithms with different initialized swarm, where the abscissa presents iteration numbers and y-axis presents mean best fitness. The figure implies that the IWMPSO generally get global optima when less than 200 times iterations, and obviously, its convergence speed is better than that of the WMPSO and the SPSO. Except f_5, the convergence accuracy of the IWMPSO is superior to the WMPSO and the SPSO more than 10^{10}. This means the convergence accuracy of the IWMPSO is better than that of the WMPSO and the SPSO, which proves the feasibility and validity of the IWMPSO. In a word, the performance of the IWMPSO is superior to both the WMPSO and the SPSO.

5 Synthesis of Linear Array

5.1 Basic Concept of Synthesis

Synthesis of array is to design the array parameters when expected pattern or main lobe width and side lobe level are given. It is an optimal problem with multi-dimensional non-linear characteristics. Considering an equal distance non-uniform linear array with 2N elements, the distance of elements is half wavelength, and the amplitude of current is symmetry. Suppose the phase of every element is zero, the pattern beam of the array is given by

$$\begin{aligned} F(\theta) &= A_1(e^{j\frac{1}{2}kd\cos\theta} + e^{-j\frac{1}{2}kd\cos\theta}) + A_2(e^{j\frac{3}{2}kd\cos\theta} + e^{-j\frac{3}{2}kd\cos\theta}) + \\ &\quad \cdots + A_N(e^{j\frac{2N-1}{2}kd\cos\theta} + e^{-j\frac{2N-1}{2}kd\cos\theta}) \\ &= 2\sum_{i=1}^{N} A_i \cos\left[(2i-1)\frac{1}{2}kd\cos\theta\right] \end{aligned} \tag{8}$$

where θ is direction angle of incident signal to array axial.

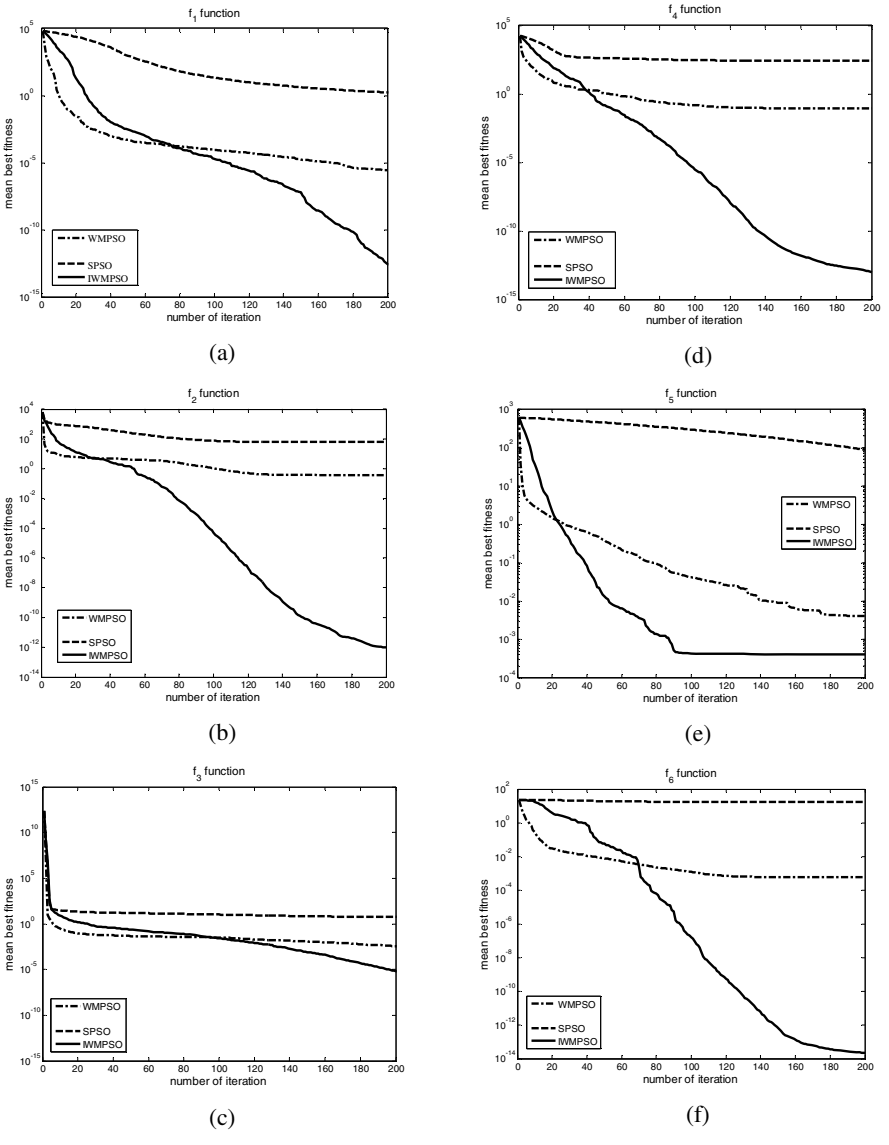


Fig. 1. Optimal processes of different PSO algorithms

The object function is defined by

$$f_1 = \eta(MSLVL - SLVL) + (1-\eta)(MBW - BW)^2 \tag{9}$$

where MSLVL is the maximum side lobe level, SLVL is the designed side lobe level, MBW is the beam width at zero power, BW is the designed beam width at zero power, and η is weight. In this work, η=0.8.

5.2 Example of Synthesis

In this example, the design indexes are 2N=20, SLVL=－40 dB, d=λ/2, beam width at zero power $2\theta_0$=20°, the range of current amplitude is [0,1], and the number of particles is 100. The patterns are drawn in Figure 2 (a) and Figure 2 (b) when the iteration number is 100 times runs and 500 times runs, respectively, and their current amplitudes are listed in column 2 and column 3 of Table 3.

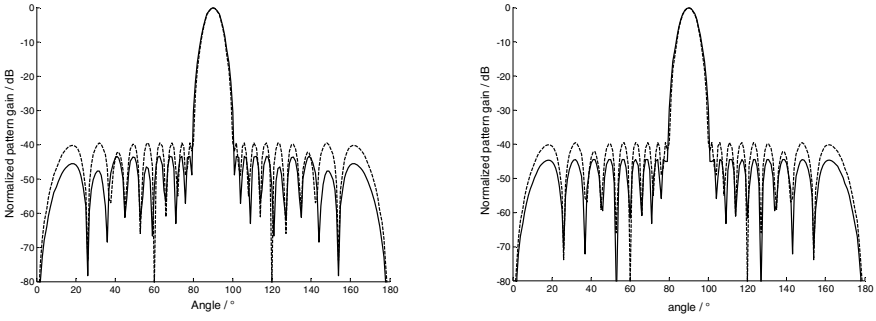


Fig. 2. Pattern after synthesized (Dashed line presents the result of reference [11] whereas solid line presents the result of this work)

Table 3. Current amplitude after optimized

Number of element	100 times runs	500 times runs
1, 20	0.8880	0.9553
2, 19	0.8467	0.9117
3, 18	0.7690	0.8292
4, 17	0.6716	0.7168
5, 16	0.5472	0.5871
6, 15	0.4224	0.4525
7, 14	0.3072	0.3255
8, 13	0.2022	0.2143
9, 12	0.1223	0.1265
10, 11	0.0693	0.0752

Figure 2(a) is the pattern after 100 times runs, beam width is 20° at zero power, and maximum side lobe level is －43.5016 dB. Figure 2(b) is the pattern after 500 times runs, and the beam width is 20° at zero power, the side lobe level is almost equal, the maximum side lobe level is decreased to －44.4797dB. In reference [11], the maximum side lobe level is －39.5996 dB after 1000 times runs, and beam width at zero power is same with this study. However, the maximum side lobe level after 100 times runs in this study is 3.9020dB lower than reference [11]. Also, it turns out in Figure 2 that the optimal result after 500 times runs is obviously not better than that after 100 times runs, which means 100 times runs is almost enough in array synthesis.

6 Conclusion

Targeting at easy premature convergence, trapping local optimum, and low convergence accuracy, this work proposes an improved wavelet-based mutation particle swarm optimization algorithm. The performances of the IWMPSO, WMPSO and SPSO are compared by benchmark test functions. The computing results show that the IWMPSO can increase the diversity of swarm and avoid local optimum. Meanwhile, the IWMPSO algorithm has some advantages, such as high convergence accuracy, good stability, small iteration number, and fast computing speed. Moreover, a pattern synthesis of linear antennas array is implemented successfully using the algorithm. It further proves the feasibility and effectiveness of the IWMPSO algorithm.

Acknowledgement. This work is supported by the Program for Postgraduates Research Innovation in University of Jiangsu Province, the Pre-research foundation of shipping industry of China under grant No. 10J3.5.2 and the Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions.

References

1. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: IEEE Int. Conf. on Neural Networks, pp. 1942–1948. IEEE Press, Piscataway (1995)
2. Zeng, J.C., Jie, J., Cui, Z.H.: Particle swarm optimization. Science Press, Beijing (2004)
3. Clerc, M.: Particle Swarm Optimization. ISTE Publishing Company (2006)
4. Poli, R.: Analysis of the publications on the applications of particle swarm optimization. Journal of Artificial Evolution and Applications (4) (2008)
5. Robinson, J., Rahmat-Samii, Y.: Particle swarm optimization in electromagnetics. IEEE Trans. on Antennas and Propagation 52(2), 397–407 (2004)
6. Mussetta, M., Selleri, S., Pirinoli, P., et al.: Improved Particle Swarm Optimization algorithms for electromagnetic optimization. Journal of Intelligent and Fuzzy Systems 19(1), 75–84 (2008)
7. Tian, Y.: Solving complex transcendental equations based on swarm intelligence. IEEJ Trans. on Electrical and Electronic Engineering 4(6), 755–762 (2009)
8. Ling, S.H., Iu, H.H.C., Chan, K.Y., et al.: Hybrid Particle Swarm Optimization With Wavelet Mutation and Its Industrial Applications. IEEE Trans. on Systems, Man, and Cybernetics – part B: Cybernetics 38(3), 743–763 (2008)
9. Shi, Y., Eberhart, R.: Empirical study of particle swarm optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation, pp. 1945–1950 (1999)
10. Ruch, D.K., Van Fleet, P.J.: Wavelet theory: an elementary approach with applications. Wiley-Interscience (2009)
11. Xiao, L.S., Huang, H., Xia, J.G., et al.: Antennas Beam Pattern Synthesis Based on Neighborhood Particle Swarm Optimization. Communications Technology 42(9), 52–53+ 71 (2009)

Elastic Boundary for Particle Swarm Optimization

Yuhong Chi[1,2], Fuchun Sun[1], Langfan Jiang[2], Chunming Yu[2], and Ping Zhang[3]

[1] Tsinghua National Laboratory for Information Science and Technology,
Department of Computer Science and Technology,
Tsinghua University, Beijing 100084, China
[2] Unit 65053, PLA, Dalian 116113, China
[3] Unit 65044, PLA, Dalian 116113, China
chiyh06@mails.tsinghua.edu.cn

Abstract. Standard particle swarm optimization (PSO) introduced in 2007, here called 2007-sPSO, is chosen as a starting algorithm in this paper. To solve the problems of the swarm's velocity slowing down towards zero and stagnant phenomena in the later evolutionary process of 2007-sPSO, elastic boundary for PSO (EBPSO) is proposed, where search space boundary is not fixed, but adapted to the condition whether the swarm is flying inside the current elastic search space or not. When some particles are stagnant, they are activated to speed up in the range of the current elastic boundary, and personal cognition is cleared. Experimental results show that EBPSO improves the optimization performance of 2007-sPSO, and performs better than comparison algorithms.

Keywords: particle swarm optimization, boundary, search space, personal cognition, optimization performance.

1 Introduction

Particle swarm optimization (PSO) [1] proposed in 1995 by Kennedy and Eberhart is a powerful evolutionary optimization algorithm which is inspired by social behaviors of fish schooling and bird flocking. In almost two decades, the fast developments and performance improvements of PSO in different ways have been achieved, although little work is reported on search space boundary control in PSO algorithm. EI-Abd [2] introduced a PSO-Bounds method for building a probabilistic model of the promising regions in the search space; Galan [3] presented floating boundaries for the search space, where floating boundaries' positions are moved depending on the number of particle hits; in CSV-PSO algorithm [4], the ranges of both search space and velocity of the swarm are contracted dynamically with the evolution of PSO algorithm; and Kitayama presented ARPSO [5] algorithm, where search domain range is determined by the mean and standard deviation of each design variable, and both of the best position and the side constraints are considered. The above mentioned studies are several representatives of formal reports, and are verified to be effective.

In this study, a novel elastic boundary for PSO (EBPSO) algorithm is proposed, where the elastic boundary is pressed, extended, or reset according to the range of the

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 125–132, 2012.

swarm's distribution. In EBPSO, the search space is not fixed, and all particles fly in a promising space surrounded by the elastic boundary. The experimental results illustrate the proposed EBPSO algorithm is superior to comparison algorithms.

2 Stagnation Phenomena in 2007-sPSO Algorithm

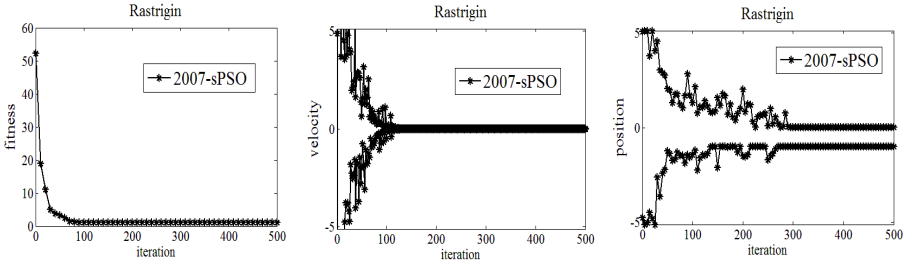
In this study, 2007-sPSO is chosen as a starting reference algorithm, and its mathematical description is given as follows:

$$v_{t+1}=\omega v_t+\tilde{c}_1(p_t-x_t)+\tilde{c}_2(n_t-x_t) \tag{1}$$

$$x_{t+1}=x_t+v_{t+1} \tag{2}$$

Where t is current time step, x_t and v_t are the position and velocity of particle i at t, respectively; p_t is personal best position of particle i, and n_t is its local best position found by its neighbors; $\tilde{c}_1$, $\tilde{c}_2$ are random number drawn from the uniform distribution on [0, c], c=1.193, and ω=0.721. The values given in 2007-sPSO come from more complete analysis, and more features about 2007-sPSO can be found in [6], [7]. However, like any other optimization algorithm, 2007-sPSO sometimes works well on a complex problem and sometimes not so well on another.

Fig. 1 is an example of the evolutionary process of 2007-sPSO on function Rastrigin, where (a), (b), and (c) are curves of fitness value, velocity and position of the swarm, respectively. It is clear that the convergence speed is fast as shown in Fig.1 (a), and the swarm's velocity drops quickly, and the swarm gathers together as shown in Fig.1 (b) and (c). The efficient optimization occurs at the beginning of evolutionary process. In the later of the evolutionary process, observed Fig. 1(b) and (c), the swarm sinks into some area, and the velocity is slowing down towards zero, and their positions are still running to steadiness. The global best of 2007-sPSO hasn't updated for several steps as shown in Fig. 1(a). This phase is named as stagnation [8].



(a) convergence curve (b) velocity (c) position

Fig. 1. An example for the evolutionary process of 2007-sPSO

In PSO, particles change their positions in search space to test whether the new position is better. Therefore stagnation doesn't benefit for the optimization performance.

3 The Proposed EBPSO Algorithm

It seems that, as a rule, search space is always fixed through the whole evolutionary process. When particles gather into a mass, most of search space is given up. If the optimum solution is just there, it is no doubt that the amazing optimization will be achieved. But unlucky the optimum often hides outside some area where particles mass. Therefore, it is desirable to determine a valid search range for particles.

For this purpose Elastic boundary for PSO algorithm (EBPSO) is proposed in this paper, which is simple and easy to understand. Fig. 2 is the pesudocode of EBPSO algorithm, and step 2-4 are the same processes as 2007-sPSO.

```
1.  for each iteration t
2.     calculate the fitness
3.     Update the best value;
4.     Update the position and velocity;
5.     for each dimension d
6.        if ER_l(d) <
7.           reset the left elastic boundary EB_l(d);
8.        else
9.           if O_l(d) =1
10.             Outspread EB_l(d);
11.          else
12.             Constrict EB_l(d);
13.          end if
14.       end if
15.       if ER_r(d) <
16.          reset the right elastic boundary EB_r(d);
17.       else
18.          if O_r(d) =1
19.             Outspread EB_r(d);
20.          else
21.             Constrict EB_r(d);
22.          end if
23.       end if
24.     end for
25. end for
26. activate velocity;
27. clear personal cognition;
```

Fig. 2. Pseudocode for EBPSO algorithm

The main characters of EBPSO algorithm is described as follows:

1. Track the Swarm

The particles' distribution is recorded by checking whether some particle is flying outside the current elastic boundary $[EB_l(d), EB_r(d)]$ on dimension d using (3) and (4).

If $O_l(d)$=0, it means that all particles fly inside the left elastic search space on dimension d, or else $O_l(d)$ =1. Similarly, we can get the value of $O_r(d)$ using (4), $[EB_l, \ EB_r]^D$ is initialized to the predefined search space [xmin, xmax] ^{D}.

$$O_l(d) = \begin{cases} 1, & \text{if } x_i(d) < \mathrm{EB}_l(d) \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

$$O_r(d) = \begin{cases} 1, & \text{if } x_i(d) > \mathrm{EB}_r(d) \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

2. EB Strategy

Elastic boundary (EB) strategy is a new way to adjust the search boundary on dimension at each iteration step. The scope between the left and right elastic boundary is elastic region, named ER in short. In each dimension, ER(d) is divided into two segments by the global best. For convenient, let ER_l =| g_t-xmin | denotes the left part, and ER_r=| xmax-g_t | denotes the right part. In Fig. 2 step 9-13 controls the left elastic boundary EB_l , which is calculated on dimension as below:

$$\mathrm{EB}_l(d) = g_t(d) - \mathrm{ER}_l(d) \times r_b \tag{5}$$

Step 18-22 controls the right elastic boundary EB_r on dimension as below:

$$\mathrm{EB}_r(d) = g_t(d) + \mathrm{ER}_r(d) \times r_b \tag{6}$$

Where g_t is the global best, $r_b \in$[a, b] is scale factor with a<1 and b>1. when $r_b \in$[1,b], EB is pushed away from the global best, so ER is expanded; if $r_b \in$ [a, 1], EB is pulled nearer to the global best, and ER is reduced. In general, let a<b-1, so that the speed of approach is slower than that of rebound off the swarm. EB strategy gives the swarm enough search scope, and doesn't tighten and disturb the swarm behaviors.

The above operations on boundary control are under the condition of ER is more than the threshold ε. if false, i.e. ER_l <ε or ER_r <ε, then EB_l or EB_r is reset randomly in the range of [g_t, xmin] or [xmax, g_t] using step 7 or 16.

3. Activate the Swarm

When EB is reset, we activate some particles' velocities by adding random values. Four different methods for setting velocity are given as follows:

- Uniform distribution, set v=U(EB_l- EB_r, EB_r-EB_l);
- Hammersley, set v=U(EB_l, EB_r)-x;
- improved Hammersley, set v=0.5*(U(EB_l, EB_r)-x);
- distance, set v=x'-x, where x' is another particle's position, chosen at random.

Based on the experience results obtained from different methods, it shows that improved Hammersley method has better performance in general.

4. Clear Personal Cognition

For example, as shown in Fig. 3, the point $f_{\min}$ is the actual optimum. Based on the principle of PSO algorithm, particle i may move to the point x_i' under the action of

three force, so it flies from the right to the left of the optimum. If the personal best position of particle i is just x_i, without its personal cognition p_i, it may move closer to the optimum point. To alleviate the effect of personal cognition and improve the ability to escape from stubborn local optima, some particles' personal cognition is clear in EBPSO algorithm when the current boundary is reset.

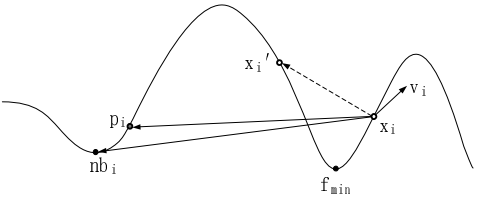


Fig. 3. Effect of cognition to particle's movement

4 Experiment Evaluation

4.1 Experiment Setup

Table 1 shows four benchmark functions [9] employed in this section, which are widely used in evaluating performance of PSO. The source codes for these functions are consulted from http://www.ntu.edu.sg/home/EPNSugan/. To validate the proposed EBPSO algorithm, the performance of EBPSO is compared with PSO-Bounds [2], CSV-PSO [4], ARPSO [5], and 2007-sPSO [6], where the first three are selected as comparisons based on the related work about search space boundary control, and their parameters setting can be found in the corresponding literatures. The last is used as a starting reference algorithm of our study. The parameters of EBPSO are set the same as 2007-sPSO. For each comparison test, the following values are used: problem dimension D is 30 and swarm size ps is 50; the number of independent runs is 100, and each run with the maximum of 10000 iteration steps. A run is terminated if either the required accuracy or the maximum of iteration steps is reached.

Table 1. Four CEC'2008 benchmark functions employed in this study

Name	Function	Bounds	Global Optimum	
shifted Sphere	$f_1(x)=\sum_{i=1}^{D} z_i^2 - 450$	$[-100,100]^D$	$x^*=o$	$f_1(x^*)=-450$
shifted Griewank	$f_2(x)=\frac{1}{4000}\sum_{i=1}^{D} z_i^2 - \prod_{i=1}^{D}\cos(\frac{z_i}{\sqrt{i}})+1-180$	$[-600,600]^D$	$x^*=o$	$f_2(x^*)=-180$
shifted Rastrigin	$f_3(x)=\sum_{i=1}^{D}(z_i^2-10\cos(2\pi z_i)+10)-330$	$[-5.12, 5.12]^D$	$x^*=o$	$f_3(x^*)=-330$
shifted Rosenbrock	$f_4(x)=\sum_{i=1}^{D-1}(100(z_{i+1}-z_i^2)^2+(z_i-1)^2)+390$	$[-50,50]^D$	$x^*=o$	$f_4(x^*)=390$

4.2 Experimental Results

Convergence curves between EBPSO and other comparison algorithms for four test functions are illustrated in Fig. 4. It can be seen that PSO-Bounds and CSV-PSO are easy to trap into local optima, and they have similar optimization performance. EBPSO, 2007-sPSO, and ARPSO have better ability to escape from local optima than PSO-Bounds and CSV-PSO, especially, EBPSO algorithm achieves outstanding performance for multimodal function Rastrigin. The convergence speed of EBPSO and 2007-sPSO is faster than the others.

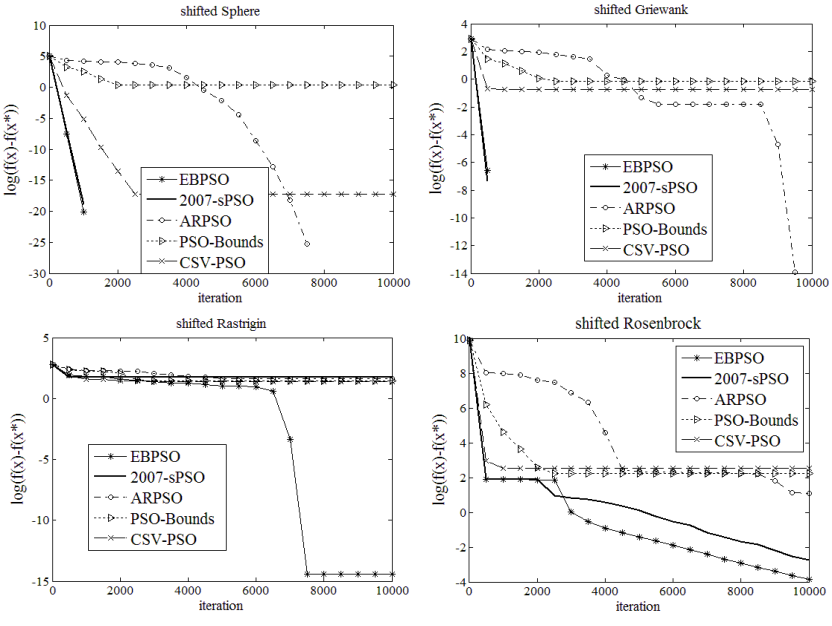


Fig. 4. Convergence curves between EBPSO and other comparison algorithms

For fair comparison, we use box-plots to evaluate the optimization performance between EBPSO and the other PSO algorithms, as shown in Fig. 5, where y-axis is fitness value. Box-plots offer much of information in a compact way. On each box, the box itself contains the middle 50% of 100 optimization results obtained by each comparison algorithm, the bottom and top of the box are the lower and upper quartiles, and the line in the box indicates the median value of 100 optimization results, the whiskers extend to a maximum of 1.5 times the inter-quartile range, and any data not included between the whiskers is plotted as an outlier with a cross.

In Fig. 5, for function shifted Rastrigin, it is clear that PSO-Bounds performs better than 2007-sPSO, ARPSO, and CSV-PSO, except of this, it is worse than the rest comparison algorithms. It is worth noting that the proposed EBPSO algorithm is obviously superior to all of comparison algorithms for function Rastrigin. Though 2007-sPSO and ARPSO are significantly inferior to EBPSO for function Rastrigin, their optimization abilities are similar to EBPSO for the other test functions.

For further comparison between EBPSO, 2007-sPSO and ARPSO, the success rates and mean values over 100 independent runs for EBPSO, 2007-sPSO and ARPSO are presented in Fig. 6 and Table 2, respectively.

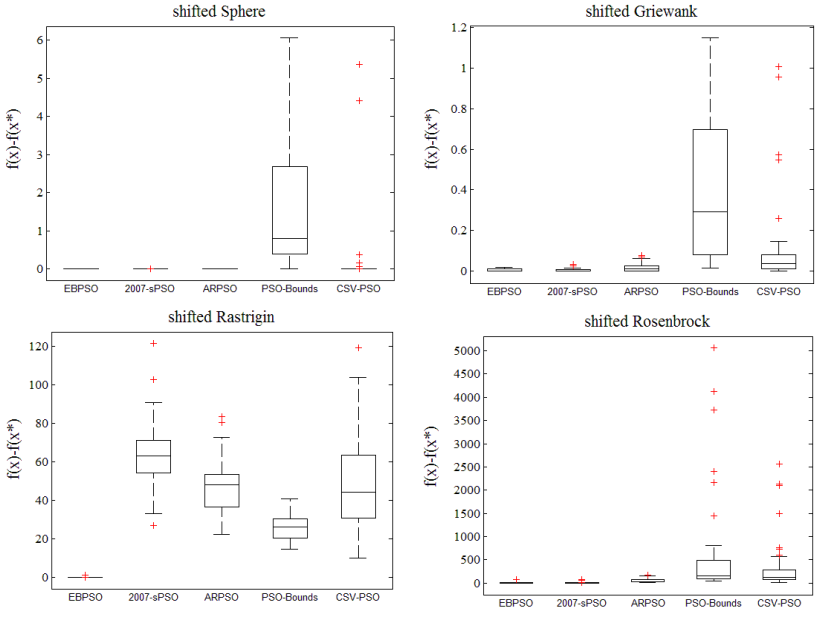


Fig. 5. Boxplots of EBPSO and the other comparison algorithms

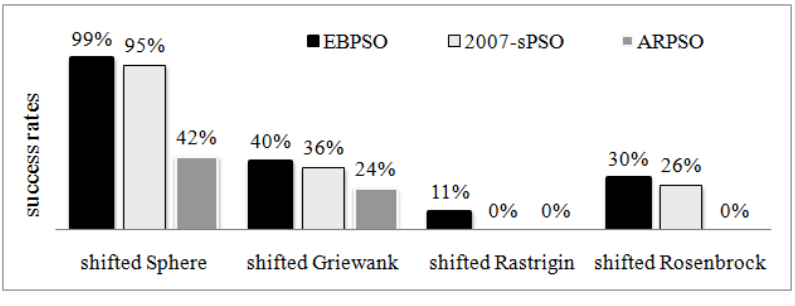


Fig. 6. Success rates for EBPSO and comparison algorithms

Table 2. Mean values for EBPSO and comparison algorithms

	shifted Sphere	shifted Griewank	shifted Rastrigin	shifted Rosenbrock
EBPSO	5.05E-31	5.26E-03	3.98E-01	2.85E+00
2007-sPSO	1.77E-30	5.49E-03	6.08E+01	3.93E+00
ARPSO	1.09E-28	1.84E-02	4.92E+01	5.12E+01

In Fig. 6, the success rate of ARPSO is zero for both of function Rastrigin and Rosenbrock, that is, ARPSO can't obtain the predefined accurate solution for the two

test functions. Though 2007-sPSO has also zero success rate and the mean value of 2007-sPSO is inferior to ARPSO for function Rastrigin as shown in Table 2, the success rate and mean value of 2007-sPSO is better than ARPSO for most of test functions. In particularly, the 2007-sPSO algorithm has 26% success rate, however the success rate of ARPSO is zero. So that 2007-sPSO is an admiring optimization algorithm. Among these comparison algorithms, the performance of 2007-sPSO is similar to the proposed EBPSO algorithm, although they are different. From Fig. 4 to Fig. 6, and Table 2, they all prove that EBPSO performs better than 2007-sPSO.

In conclusion, the optimization performance of EBPSO is more stable, and performs better than other comparison algorithms, especially for multimodal function Rastrigin, EBPSO achieves amazing performance.

5 Conclusion and Future Work

Aim at the problems of swarm's velocity falling towards zero and stagnant phenomena in the later evolutionary process of 2007-sPSO, we proposed EBPSO to adjust the search space boundary and alleviate the local minima. Experiments indicate that PSO equipped with elastic boundary provides stable convergence and a better probability of success. In future, we will study a more efficient method to deal with the violate particles instead of the simple way used in this paper.

References

1. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: IEEE International Conference on Neural Networks, Perth, Australia, pp. 1942–1948 (1995)
2. EI-Abd, M., Kamel, M.S.: Particle Swarm Optimization with Varying Bounds. In: IEEE Congress on Evolutionary Computation, Singapore, pp. 4757–4761 (2007)
3. Galan, A.Y., Boryskina, O.P., Sauleau, R., Boriskin, A.V.: Particle Swarm Optimization Algorithm with Moving Boundaries as a Powerful Tool for Exploration Research. In: 5th European Conference on Antennas and Propagation (EUCAP), Rome, pp. 1961–1964 (2011)
4. Chen, B.R., Feng, X.: Particle Swarm Optimization with Contracted Ranges of Both Search Space and Velocity. Journal of Northeastern University (Natural Science) 26(5), 488–491 (2005) (in Chinese)
5. Kitayama, S., Yamazaki, K., Arakawa, M.: Adaptive Range Particle Swarm Optimization. Optimization and Engineering 10(4), 575–597 (2009)
6. Clerc, M.: From Theory to Practice in Particle Swarm Optimization. In: Panigrahi, B.K., Shi, Y., Lim, M.-H. (eds.) Handbook of Swarm Intelligence. ALO, vol. 8, pp. 3–36. Springer, Heidelberg (2011)
7. Clerc, M.: Standard Particle Swarm Optimization, `http://clerc.maurice.free.fr/pso/`
8. Poli, R.: Mean and Varinace of the Sampling Distribution of Particle Swarm Optimizers During Stagnation. IEEE Transactions on Evolutionary Computation 13, 712–721 (2009)
9. Tang, K., Yao, X., Suganthan, P.N., MacNish, C., Chen, Y.P., Chen, C.M., et al.: Benchmark Functions for the CEC 2008 Special Session and Competition on Large Scale Global Optimization. Tech. Rep. No. NCL-TR-2007012, Hefei, China (2007)

Optimization Locations of Wind Turbines with the Particle Swarm Optimization

Ming-Tang Tsai and Szu-Wzi Wu

Department of Electrical Engineering, Cheng-Shiu University, Kaohsiung, Taiwan
{Ming-Tang Tsai,tsaymt}@csu.edu.tw

Abstract. In this paper, a new algorithm is presented for the locations of wind turbine in the distribution systems. Technical constraints such as feeder capacity limits, bus voltage, and load balance are considered. The Particle Swarm Optimization(PSO) is applied to solve this problem. To enhance the performance of the new algorithm, a load flow program with Equivalent Current Injection (ECI) is used to analyze the load flow of distribution systems. Based on ECI load flow model, a constant Jacobian matrix is determined to improve the existing power-based model by using the Norton Equivalent Theorem. Example of IEEE 69-bus system is adopted to illustrate the efficiency and feasible of the proposed algorithm. Test results show that with proper site selections of wind turbines can be used to reduce system losses and maintain the voltage profile.

Keywords: Particle Swarm Optimization, Equivalent Current Injection, Wind Turbine, Distribution System.

1 Introduction

Distribution system planner must ensure that there is adequate substation capacity and feeder capacity to meet the load growth within the planning horizon year. In the past few years, the utilities has faced many challenges in a competitive market due to increased investment cost and required the high level of reliability in a system. They have the obligation for customers to supple the service reliability by planning, operation, construction, and maintenance. Due to advance in small wind energy technologies, utilities began to integrate the Wind Turbines(WTs) in the distribution system for reducing the system loss and improving service reliability. WTs are mostly installed in demand system and directly connected to distribution networks. Appropriate location of the WTs will become an important problem[1-5].

Optimal locations of the WTs are gaining interests in the electricity industries. Some import factors such as rating, location, and operating power factor have to be carefully considered in the planning process. A load flow algorithm is illustrated that inappropriating locations of WTs may lead to greater system loss[6]. [7] shows the optimal placement and sizing of capacitor in a distribution system to reduce system loss. Both WTs and capacitors may reduce power loss and improve voltage profiles but overall efficiency can be improved using WTs only. [8] used Genetic Algorithm(GA) for the placement of WTs to reduce the system loss. Some approaches

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 133–141, 2012.

are introduced for placement of WTs on the basis of loss reduction[9-12]. Tabu Search(TS) is used to determine the location and size of the WTs by minimizing the system loss[13]. [14] used the Liner Programming(LP) to find optimal location of embedded generation in distribution networks. However, the penetration of WTs in the distribution system is increasing, and it is important to place the WTs in such an optimal way that it will reduce system loss and improve the voltage profiles.

In this paper, a Particle Swarm Optimization(PSO) based approach is presented to optimally incorporate WTs into a distribution system. The proposed algorithm combines PSO with load flow algorithm to find the best combination of locations. To enhance the performance of the new algorithm, a load flow model with Equivalent Current Injection (ECI) [15] is used to analyze the load flow. The objective function is to minimize the distribution network loss while satisfying the operational constraints. This problem is formulated as a mixed-integer non-linear optimization problem. The IEEE 69-bus distribution system[16] is used to validate the proposed method. Numerical results are also provided to show its effectiveness.

2 Mathematical Formulation

The objective function can be expressed as

$$\text{Min} \quad P_{loss} = \frac{1}{2}\sum_{i=1}^{NB}\sum_{j=1}^{NB}\text{Re}\left[Y_{ij}\right]\left[\left|V_i\right|^2 + \left|V_j\right|^2 - 2\left|V_i\right|\left|V_j\right|\cos\theta_{ij}\right] \tag{1}$$

Where,

P_{loss} :the total line loss

Y_{ij} :the admittance of branch $i-j$

Re :the real part of complex quantity

NB :the total number of branches in the system

V_i :The voltage of $i-th$ bus

$\theta_{ij} = \theta_i - \theta_j$:the angle of voltage

From the Equation (1), the line loss could be reduced by lowering the branch currents in the distribution network. In order to reduce the current in certain parts of the network, WTs is introduced to the distribution network.

The constraints considered are described as follows

1. The equality constraints are the load flow equations in the radial distribution system as follows.

$$P_{i,sch} = P_{wind,i} - P_{di} = \left|V_i\right|\sum_{j=1}^{NB}\left|V_j\right|\left|Y_{ij}\right|\cos\left(\theta_i - \theta_j - \delta_{ij}\right) \tag{2}$$

$$Q_{i,sch} = Q_{wind,i} - Q_{di} = \left|V_i\right|\sum_{j=1}^{NB}\left|V_j\right|\left|Y_{ij}\right|\sin\left(\theta_i - \theta_j - \delta_{ij}\right) \tag{3}$$

2. The inequality constraints are the voltage limits imposed on the radial distribution system.

$$|V_i^{\min}| \le |V_i| \le |V_i^{\max}| \tag{4}$$

3. Line flow constraints from bus i to bus j.

$$S_{ij} \le S_{ij}^{\max} \tag{5}$$

4. The inequality constraints with the WTs real power output

$$P_{wind,i}^{\min} \le P_{wind,i} \le P_{wind,i}^{\max} \tag{6}$$

5. Pitch angle limit of WTs at bus

$$\theta_i^{\min} \le \theta \le \theta_i^{\max} \tag{7}$$

δ_{ij} : the angle of branch i-j element of the admittance

S_{ij} :The line flow in the branch i-j

$S_{ij}^{\max}$:the upper line flow in the branch i-j

$P_{wind,i}^{\min}$, $P_{wind,i}^{\max}$:the lower and upper real power generation of WT at i-th bus

$P_{wind,i}$:the real power generation for WT at i-th bus

$Q_{wind,i}$:the reactive power generation of WT at i-th bus

P_{di} :the real power demand at i-th bus

Q_{di} :the reactive power demand at i–th bus

The turbine mechanical power of the turbines is described by Equations (8)(9)[17].

$$P_{wind} = 0.5\rho A C_p V_{wind}^3 \tag{8}$$

$$\lambda = \frac{\omega * \gamma}{V_{wind}} \tag{9}$$

Where ρ =air density, A=rotor swept area, C_p =power coefficient function, V_{wind} = wind speed. In this paper, C_p are assigned as follows.

$$\text{Cp}=0.5716\left(\frac{116}{\lambda i}-116*0.4-5\right)e^{-\left(\frac{21}{\lambda}-21*0.035\right)}+0.018\lambda \tag{10}$$

Where,

$$\lambda_i = \frac{(\lambda+0.08\theta)(\theta^3+1)}{(\theta^3+1-0.035*\lambda-0.0028*\theta)}$$

3 Load Flow Model with Equivalent Current Injection

Two major load flow techniques used in the industrial application are Gauss-Seidel and Newton-Raphson based algorithms. The Gauss-Seidel algorithm is a slow convergence and uses a full matrix which directly defines the problem to be solved and can not be altered. The Newton-Raphson algorithm is a gradient technique where the line parameters are stored in the Jacobian matrix. [15] presented a bi-factorized complex Y-admittance matrix Gauss-Seidel method which is based on the Equivalent Current Injection, and the power components can be modeled in the Y matrix or converted into ECI. In this paper, the load flow with Newton-Raphson method is proposed based on ECI.

For the power-based Newton-Raphson method, the mismatch function can be written in the rectangular form as

$$\begin{bmatrix} \Delta P_i \\ \\ \Delta Q_i \end{bmatrix} = \begin{bmatrix} \dfrac{\partial P_i}{\partial e_i} & \dfrac{\partial P_i}{\partial f_i} \\ \\ \dfrac{\partial Q_i}{\partial e_i} & \dfrac{\partial Q_i}{\partial f_i} \end{bmatrix} \begin{bmatrix} \Delta e_i \\ \\ \Delta f_i \end{bmatrix} \tag{11}$$

Where, $\Delta P_i = P_{i,sch} - P_{i,cal}$, $\Delta Q_i = Q_{i,sch} - Q_{i,cal}$

$P_{i,sch} = P_{wind,i} - P_{di}$, which is the net real power at i-th bus, includes the real power demand (P_{di}) and WT's power generation ($P_{wind,i}$). $P_{i,cal}$ is the real power, which is calculated by load flow analysis. $Q_{i,sch} = Q_{wind,i} - Q_{di}$, which is the net reactive power at i-th bus, includes the reactive power demand (Q_{di}) and WT's reactive power generation ($Q_{wind,i}$). $Q_{i,cal}$ is the reactive power, which is calculated by load flow analysis. The reactive power generation of WTs is calculated based on the pre-specified power factor. The Jacobian matrix is given by

$$J = \begin{bmatrix} \dfrac{\partial P_i}{\partial e_i} & \dfrac{\partial P_i}{\partial f_i} \\ \\ \dfrac{\partial Q_i}{\partial e_i} & \dfrac{\partial Q_i}{\partial f_i} \end{bmatrix} \tag{12}$$

The ECI-based load flow uses current instead of power. The mismatch function can be re-written by

$$\begin{bmatrix} \Delta I^r \\ \\ \Delta I^i \end{bmatrix} = \begin{bmatrix} \dfrac{\partial I^r}{\partial e} & \dfrac{\partial I^r}{\partial f} \\ \\ \dfrac{\partial I^i}{\partial e} & \dfrac{\partial I^i}{\partial f} \end{bmatrix} \begin{bmatrix} \Delta e \\ \\ \Delta f \end{bmatrix} \tag{13}$$

$\Delta I = I^{eqv} - I^{cal} = \Delta I^r + j\Delta I^i$ and $\Delta V = \Delta e + j\Delta f$ are the real and imaginary components of currents and voltages, respectively. I^{cal} is obtained from load flow analysis. I^{eqv} is given by

$$I^{eqv} = \left(\frac{P + jQ}{V} \right)^* = \mathrm{Re}(I^{eqv}) + j\,\mathrm{Im}(I^{eqv}) \tag{14}$$

P , Q , and V are the constant real power, imaginary power, and voltage at a specified bus. P and Q are also the net power at a specified bus, which include the WT's power generations and load demands.

From the equation (13), a constant Jacobian matrix can be obtained, which has the same matrix dimension as the based Newton-Raphson algorithm. The constant Jacobian matrix can be written by

$$J = \begin{bmatrix} G & -B \\ B & G \end{bmatrix} \tag{15}$$

Where, G and B are the conductance matrix.

4 The Proposed Methodology

In a PSO system, Birds (particles) flocking optimizes a certain objective function. Each particle knows its current optimal position (*pbest*), which is analogy of personal experiences of each particle. Each particle also knows the current global optimal position(*gbest*) among all particles. PSO can have several solutions at the same time, and particles have a cooperative relationship for sharing messages. Through specific equations, each particle adjusts its position and determines the search direction according to its search memory. In other words, it tries to reach compatibility between local search and global search. The search memory of a particle is the objective function and the optimum position found by the particle.

In this paper, PSO with Constriction Factor(PSO-CF)[18] was selected to trace the *pbest* and *gbest*. PSO-CF is used a constriction factor to control the trajectory of particles without considering the velocity of particles. There are more possibilities to promote the convergent rapid and the searching performance. Using the PSO-CF, the velocity can be represented in the PSO algorithm. Using the Equation (16), a certain velocity can be calculated due to the position of individuals gradually close to *pbest* and *gbest* . The current position can be modified by Equation (17).

$$V_{i,d}^{j+1} = K \times [\ V_{i,d}^{j} + c_1 \times rand(0,1) \times (pbest_{i,d}^{j} - P_{i,d}^{j}) + c_2 \times rand(0,1) \times (gbest^{j} - P_{i,d}^{j})\ ] \tag{16}$$

$$P_{i,d}^{j+1} = P_{i,d}^{j} + V_{i,d}^{j} \tag{17}$$

Where,

$$K = \frac{2}{\left|2 - c - \sqrt{c^2 - 4c}\right|} \quad , \quad c = c_1 + c_2 \quad , \quad c > 4$$

c_1 , c_2 : acceleration constant, In this paper, $c_1 = c_2 = 2.05$

$rand(0,1)$: uniform random value with a range of [0,1]

$P_{i,d}^{j}$: dimension d of the position of particle i at iteration j

$V_{i,d}^{j}$: dimension d of the velocity of particle i at iteration j

$pbest_{i,d}^{j}$: dimension d of the own best position of particle i at iteration j

$gbest^{j}$: dimension d of the best particle in the swarm at iteration j

The proposed methodology can be summarized in the following steps.

(a) Randomly initialize 30 particle(WTs) with feasible position in the system buses.
(b) Randomly assign velocities (V_{wind}) and pitch angles(θ) of WTs to each particle. By using Equation (11), the power output of WTs can be calculated. If the power output is less 100KW, the velocities and pitch angles of WTs are re-generated.
(c) Perform the load flow model with ECI to calculate the system losses.
(d) Determine the best particle dependent upon the system losses of entire particles.
(e) Update velocity and position vectors according to (16) and (17) for each particle.
(f) The terminating condition is maximal number of iterations. If the preset target is not yet attained, then go back to Step (a) and repeat operation.

5 Case Study

The proposed algorithm is applied to solve the 69-bus distribution system[16]. The numerical computations were performed using the Matlab language on a PIV-2.6GHZ computer with 512MB RAM. The PSO parameter used in this paper is 30 particles. 500 generations is set as the stopping criteria. The power output of WTs ranges from 100KW to 200KW dependent upon the pitch angle and wind speed. The number of WTs are located in the buses to find the optimal system losses. Table 1 shows the simulated results. It is clear from the obtained results that the number of WTs has significantly improved the system loss. The loss reduction ranges from 0% to 40.98% due to the WTs are added in the distribution system. Figure 1 shows the voltage profiles before and after the WTs installed. From the Figure 1, it can be shown that the voltage profile is clearly improved after the WTs installed, becoming almost satisfy the voltage limits. Figure 2 is the convergent characteristics of the proposed method. The convergent generation is about 250 generation.

6 Conclusion

This paper presented a PSO for the locations of WTs in the distribution systems in order to minimize the system losses. Technical constraints such as feeder capacity limits, bus voltage profile, and load balance are considered. To enhance the performance of the new approach, a load flow model with ECI is used to analyze the load flow of distribution systems. The effectiveness of the proposed algorithm is tested on IEEE 69-bus distribution system. Results show that incorporating the WTs in the distribution system can reduce the system losses and maintain the voltage profile. In addition, the results show that the different locations of WTs could also produce multi-solutions to achieve the real global or nearly global solution.

Table 1. The simulated results

The No. of WTs	Location Bus	The output of WTs (KW)	Loss (KW)	Reduction (%)
0	***	***	86	0
1	62	131	79.4	7.67
2	61,69	262	76.6	10.93
3	61,62,63	393	67.535	21.47
4	24,61,63,64	523	64.2	25.38
5	55,59,61,63,64	654	62.1	27.79
6	43,53,57,61,63,64	786	61.4	28.60
7	8,59,61,62,63,68,69	915	58	32.56
8	15,18,19,24,25,27,61,63	1042	56.4	34.42
9	12,20,37,42,58,61,62,64,65	1179	53.4	37.91
10	17,27,54,56,57,58,61,62,63,66	1309	50.761	40.98

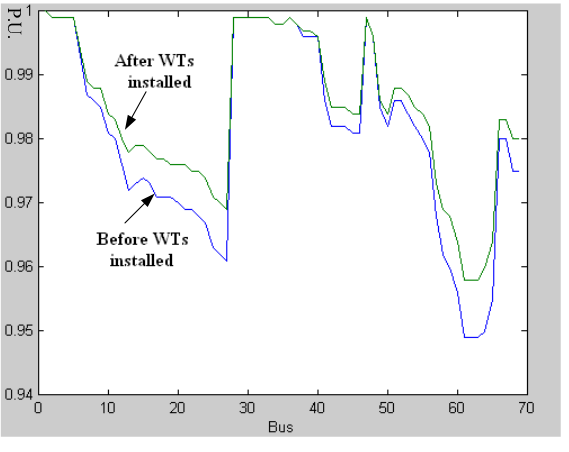


Fig. 1. The voltage profiles before and after the WTs installed

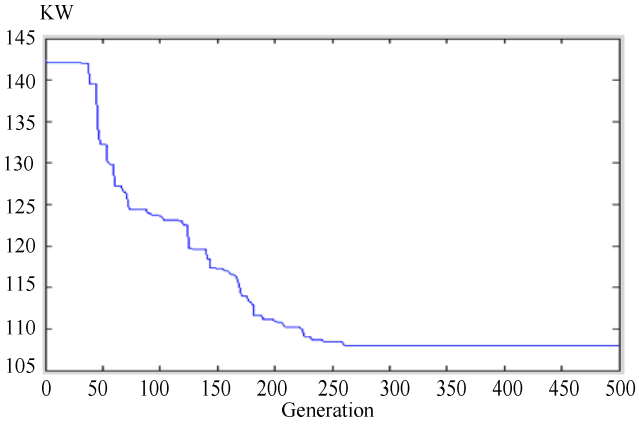


Fig. 2. The convergent characteristics of the proposed method

References

1. Yasin, Z.M., Rahman, T.K.A., Musirin, I., Rahim, S.R.A.: Optimal sizing of distributed generation by using quantum-inspired evolutionary programming. In: 2010 4th International Power Engineering and Optimization Conference (PEOCO), pp. 468–473 (2010)
2. Celli, G., Ghiani, E., Mocci, S., Pilo, F.: A Multi-objective approach to maximize the presentation of distributed generation in distribution networks. In: International Conference on Probabilistic Methods Applied to Power Systems, pp. 1–6 (2006)
3. Ghosh, S., Ghoshal, S.P.: Two analytical approaches for optimal placement of distributed generation unit in power systems. In: International Conference on Power Systems, pp. 1–6 (2009)
4. Singh, A.K., Parida, S.K.: Optimal placement of DGs using MINLP in deregulated electricity market. In: 2010 Proceedings of the International Conference on Energy and Sustainable Development: Issues and Strategies (ESD), pp. 1–7 (2010)
5. AlHajri, M.F., AlRashidi, M.R., El-Hawary, M.E.: Hybrid Particle Swarm Optimization Approach for Optimal Distribution Generation Sizing and Allocation in Distribution Systems. In: Canadian Conference on Electrical and Computer Engineering, pp. 1290–1293 (2007)
6. Yan, G.G., Yuan, T.F., Zhang, Z.Q., Mu, G., Zhang, C.X., Xu, F.: Optimal allocation of wind turbine with DFIG for minimizing network losses in distribution systems using sensitivity analysis method. In: International Conference on Sustainable Power Generation and Supply, pp. 1–5 (2009)
7. Rau, N.S., Wan, Y.H.: Optimum location of resources in distributed planning. IEEE Trans. On Power Systems 9(4), 2014–2020 (1994)
8. Grainger, J.J., Lee, S.H.: Optimum size and location of shunt capacitors for reduction of losses. IEEE Trans. on Power Apparatus and Systems PAS 100(3), 1105–1118 (1981)
9. El-Ela, A.A., Allam, S.M., Shatla, M.M.: Maximal optimal benefits of distributed generation using genetic algorithms. Electric Power Systems Research 80(7), 869–877 (2010)

10. Tuba, G.M., Hakan, H.: An analytical method for the sizing and siting of distributed generators in radial systems. Electric Power Systems Research 79(6), 912–918 (2009)
11. Sudipta, G., Ghoshal, S.P., Saradindu, G.: Optimal sizing and placement of distributed generation in a network. International Journal of Electrical Power and Energy Systems 32, 849–856 (2010)
12. Andrew, K., Haiyang, Z.: Optimization of wind turbine energy and power factor with an evolutionary computation algorithm. Energy 35(3), 1324–1332 (2010)
13. Kumar, A., Gao, W.: Optimal distributed generation location using mixed integer non-linear programming in hybrid electricity markets. IET Generation, Transmission, and Distribution 4(2), 281–298 (2010)
14. Nara, K., Hayashi, Y., Ikeda, K., Ashizawa, T.: Application of tabu search to optimal placement of distributed generators. IEEE PES Winter Meeting 2, 918–923 (2001)
15. Keana, A., Malley, M.: Optimal allocation of embedded generation on distributed networks. IEEE Trans. on Power Systems 20(3), 2014–2020 (2005)
16. Lin, W.M., Teng, J.H.: Phase-decoupled load flow method for radial and weakly-meshed distribution networks. IEE Proceedings-Generation, Transmission and Distribution 143(1), 39–42 (1996)
17. Baran, M.E., Wu, F.F.: Optimal capacitor placement on radial distribution systems. IEEE Transactions on Power Delivery 4(1) (1989)
18. Hui, J., Bakhshai, A.: A new adaptive control algorithm for maximum power point tracking for wind energy conversion systems. In: IEEE Power Electronics Specialists Conference, pp. 4003–4007 (2008)
19. Shi, Y., Eberhart, R.C.: A modified particle swarm optimizer. In: Proceedings of the IEEE International Conference on Evolutionary Computation Anchorage, pp. 69–73 (1998)

A PSO-Based Algorithm for Load Balancing in Virtual Machines of Cloud Computing Environment

Zhanghui Liu and Xiaoli Wang

College of Mathematics and Computer Sciences, Fuzhou University. Fuzhou, China
yuhcaolong@126.com, fzugwz@163.com

Abstract. It is possible for IT service providers to provide computing resources in an pay-per-use way in Cloud Computing environments. At the same time, terminal users can also get satisfying services conveniently. But if we take only execution time into consideration when scheduling the cloud resources, it may occur serious load imbalance problem between Virtual Machines (VMs) in Cloud Computing environments. In addition to solve this problem, a new task scheduling model is proposed in this paper. In the model, we optimize the task execution time in view of both the task running time and the system resource utilization. Based on the model, a Particle Swarm Optimization (PSO) – based algorithm is proposed. In our algorithm, we improved the standard PSO, and introduce a simple mutation mechanism and a self-adapting inertia weight method by classifying the fitness values. In the end of this paper, the global search performance and convergence rate of our adaptive algorithm are validated by the results of the comparative experiments.

Keywords: Cloud Computing, VMs, Load Balancing, Task Scheduling, PSO.

1 Introduction

Nowadays, Cloud Computing has become a very popular commercial computing paradigm. Then the whole Cloud Computing system can provide services to users with virtual machine as the resources unit [1-2]. For users, all of the bottom resources are transparent. In theory, every job submitted by terminal users owned an independent virtual machine. The computing cells and memory cells for the jobs are in the situations of mutual isolation [3]. In Cloud Computing environment, each physical host can load one or more virtual machines, so that you can ensure users' applications run independently. So, the task scheduling in Cloud Computing happens between the virtual machines actually. Keep load balances of the VMs is the ongoing work in Cloud Computing systems.

It is an NP-hard combinational optimization problem to establish the mappings between jobs submitted by terminal users and dynamical resources encapsulated in the virtual machines. For such problem, researchers have put forward a variety of static, dynamic and mixed scheduling strategies [4-9]. Static scheduling algorithms are: ISH algorithm [10], MCP algorithm and ETF algorithm [10]. All of these algorithms are based on BNP (Bounded Number Processors), and very suitable for

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 142–147, 2012.

high performance networks in distributed environment. But the application requirements in Cloud Computing VMs are more complicated, and service costs are also required according to usages amount, those algorithms can not pay key roles. Currently, the swarm intelligence algorithms are well used for resolving these kinds of problems[11-12]. PSO is a global search optimization technique proposed by Kennedy and Eberhart in 1995[13]. But when the problem continues to expand there scales, Simple heuristic algorithm seem to be not so effective [14]. In this paper, a kind of improved PSO algorithm to solve the problem of virtual machines load balance is proposed, so as to establish corresponding relations between the tasks and the virtual machines effectively. Aiming at finding an optimal or nearly optimal scheduling solution, not only makes the task execution time the shortest, and can make the virtual machines of resource utilization the highest. The simulation results show that the algorithm has fast convergence speed, high efficiency, and has practical application significance.

2 Problem Description and Task Scheduling Model

In VMs of Cloud Computing environments, the jobs submitted by terminal users can be classified two kinds, which are independent ones and interrelated ones. The interrelated jobs can be divided into small separate tasks that can run without interferences, so we just study how to balance the workload of VMs with independent tasks. The objectives of our model are to achieve the minimum execution time of tasks and the maximum VMs resource utilization.

For simplicity, we assume that the virtual machine is resource unit of Cloud Computing environment. And for different virtual machines, the resource demands of any task are the same. The models can be represented as follows.

There are m virtual machines which are interconnected by network. The VMs can be represented by the set $V = (v_1, v_2, \cdots, v_m)$, in which v_i means the maximum resource capacity that virtual machine i can provide, where $i \in [1, m]$.

2.1 Task Model

There is a task sequence $T = (t_1, t_2, \cdots, t_n)$, in which t_j means the task which is number j, where $j \in [1, n]$, and n is the length of this sequence. Task model is defined as $t_j(timeNeed, resourceNeed)$, in which $timeNeed$ denotes the task's execution time, and $resourceNeed$ denotes the resource requirements of the task.

2.2 Objective Model

We use one-zero matrix A to represent the mapping relationships between tasks and VMs. There has $\sum_{i=1}^{m} a_{i,j} = 1$, and $i \in [1, m], j \in [1, n]$.

Which is can be described:

$$A = \begin{bmatrix} a_{1,1}, a_{1,2}, \cdots, a_{1,n} \\ a_{2,1}, a_{2,2}, \cdots, a_{2,n} \\ \vdots \quad \vdots \quad \vdots \\ a_{m,1}, a_{m,2}, \cdots, a_{m,n} \end{bmatrix}.$$

Based on the above models, we have the equations as follows.

$$\begin{cases} VTime = \max\limits_{i=1}^{m} \left[\sum\limits_{j=1}^{n} (a_{i,j} \times t_j.timeNeed) \right] \\ VRutilization = \sum\limits_{i=1}^{m} \left(\dfrac{\sum\limits_{j=1}^{n} a_{i,j} \times t_j.resourceNeed}{v_i} \right) \end{cases}. \tag{1}$$

where we use $VTime$ to denote the execution time of VMs for executing all of the tasks, and $VRutilization$ to denote the resource utilization of VMs during the process of running the tasks. So, the objective function of the task scheduling model is:

$$\begin{cases} Min(VTime) \\ Max(VRutilization) \end{cases}. \tag{2}$$

3 Optimization Algorithm Description

Resource allocations and scheduling strategies are combined to realize the mappings from tasks to VMs. In this paper, we introduce mutation operator and self-adaptation of inertia weight to the standard PSO algorithm aiming at virtual machines assignment for the user tasks.

3.1 Fitness Function

In order to measure how well the particle's position, the fitness function can be defined:

$$f = Min\left(\frac{VTime}{VRutilization}\right). \tag{3}$$

3.2 Define the Positions and Velocities of Particles

Suppose there is a N-dimension particle $X = (x_1, x_2, \cdots, x_n)$, where $x_i (i \in [1, n])$ is the serial number of virtual machine on which the number i task is processed. At the same time, an N-dimension velocity $V = (v_1, v_2, \cdots, v_n)$ is defined, where $v_i (i \in [1, n])$ means the velocity of x_i. And

$$\begin{cases} x_i \in Z^+ \ \&\ \&1 \le x_i \le m \\ v_i \in [-v\max, v\max] \end{cases}. \tag{4}$$

where $v\max$ denotes the maximum velocity component of particle. In this paper, we set the $v\max = m$ [15]. But, it always leads to precocity. We introduce a simple mutation mechanism: When overflow occurs, the position will get random value from the solution space.

3.3 Self-adapting Inertia Weight and Updating Positions and Velocities

$$\begin{cases} w = \begin{cases} 0.2, (\left|\dfrac{fi - fg}{\max(fi, fg)}\right| < 0.2) \\ 0.8, (\left|\dfrac{fi - fg}{\max(fi, fg)}\right| > 0.8) \\ 1 - 0.2 \cdot e^{-(fi-fg)^2 \cdot rand()}, else \end{cases} \\ vx_i^{k+1} = w \cdot vx_i^k + c_1 \cdot rand() \cdot (p_i^k - x_i^k) + c_2 \cdot rand() \cdot (fg^k - x_i^k) \\ vx_i^{k+1} = \begin{cases} vx_i^{k+1}, (\left|vx_i^{k+1}\right| \le v\max) \\ v\max, else \end{cases} \\ x_i^{k+1} = \begin{cases} \lfloor x_i^k + vx_i^{k+1} \rfloor, (1 \le x_i^k + vx_i^{k+1} \le m) \\ random\_value \in [1, m], else \end{cases} \end{cases}. \tag{5}$$

where, there has: w inertia weight, f_i the fitness value of particle i, p_i the previous best fitness value of particle i, fg the global best fitness value, x_i the position of particle i, vx_i the velocity of particle i, c_1, c_2 the coefficients which are set as 2.05 in this paper.

4 Experimental Results and Analysis

The optimization process is simulated by MATLAB in this paper, where has 80 virtual machines and 120 tasks, and with 200 iterations.

To validate the improvement of our algorithm(MAPSO), we compared it with non-adapting standard PSO which has the invariable inertia weight. Fig1 shows that both convergence speed and robustness of MAPSO algorithm are better than those of SPSO.

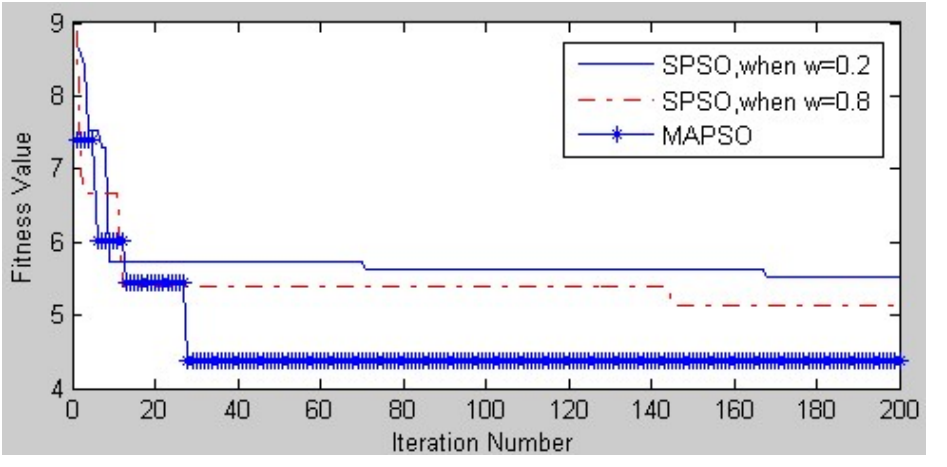


Fig. 1. Comparison of fitness values SPSO and MAPSO

5 Conclusions

Nowadays, from both scheduling flexibility and application scale, there is much work should be done on the past studies. this paper works to solve the load balancing problem in VMs of Cloud Computing environment. Based on the development of virtualization and distributed technology, we put forward MAPSO tasks scheduling algorithm by improving standard PSO. In addition, because of this experiment with simulation environment, some specific questions need to be overcome in specific actual cloud environment, such as restrictions from bandwidth, problems in job decomposition, energy costs of cloud datacenters etc.

Acknowledgments. This work was supported in part by the National Natural Science Foundation of China under Grant No. 61103175, the Key Project Development Foundation of Education Committee of Fujian province under Grand No. JA11011 and JK2010001, the Technology Innovation Platform Project of Fujian Province under Grant No.2009J1007, the project development foundation of Fuzhou University under Grand No. 2010-XQ-21 and XRC-1037.

References

1. Virtualization and Cloud Computing Group.: Virtualization and Cloud Computing, pp.110–114. Publishing House of Electronics Industry, Beijing (2009) (in Chinese)
2. Hu, J., Gu, J., Sun, G., et al.: A Scheduling Strategy on Load Balancing of Virtual Machine Resources in Cloud Computing Environment. In: 3rd International Symposium on Parallel Architectures, Algorithms and Programming, Dalian, Liaoning, China, pp. 89–96 (2010)
3. Fang, Y., Wang, F., Ge, J.: A Task Scheduling Algorithm Based on Load Balancing in Cloud Computing. In: Wang, F.L., Gong, Z., Luo, X., Lei, J. (eds.) WISM 2010. LNCS, vol. 6318, pp. 271–277. Springer, Heidelberg (2010)
4. Paton, N.W., de Aragao, M.A.T., Lee, K., Fernandes, A.A.A.: Optimizing Utility in Cloud Computing through Automatic Workload Execution. IEEE Data Eng. Bull. 32, 51–58 (2009)
5. Li, L.: An Optimistic Differentiated Service Job Scheduling System for Cloud Computing Service Users and Providers. In: Third International Conference on Multimedia and Ubiquitous Engineering, Qingdao, China, pp. 295–299 (2009)
6. Wei, G., Athanasios, V.V., Yao, Z., et al.: A game-theoretic method of fair resource allocation for Cloud Computing Services. The Journal of SuperComputing 2, 252–269 (2009)
7. Martin, R., David, L., Taleb-Bendiab, A.: A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing. In: 2010 IEEE 24th International Conference on Advanced Information Netwoking and Applications Workshops, Perth, Australia, pp. 551–556 (2010)
8. Zhang, B., Gao, J., Ai, J.: Cloud Loading Balance Algorithm. In: 2nd International Conference on Information and Engineering, ICISE 2010, Hangzhou, China, pp. 5001–5004 (2010) (in Chinese)
9. Laura, G., David, I., Varun, M., et al.: Harnessing Virtual Machine Resource Control for Job Management. In: The 1st Workshop on System-level Virtualization for High Performance Computing, Lisbon, Portugal (2007)
10. Kwok, Y.-K., Ahmad, I.: Static scheduling algorithms for allocating directed task graphs to multiprocessors. ACM Computing Surveys 4, 406–471 (2009)
11. Ji, Y.-M., Wang, R.-C.: Study on PSO algorithm in solving grid task scheduling. Journal on Communications 10, 60–66 (2007) (in Chinese)
12. Pandey, S., Wu, L., Guru, S., et al.: A Particle Swarm Optimization (PSO)-based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments. In: 24th IEEE International Conference on Advanced Information Networking and Applications, Perth, Australia, pp. 400–407 (2010)
13. James, K., Russell, E.: Particle Swarm Optimization. In: Proceedings of Neural Networks 1995, Perth, Australia, pp. 1942–1948 (1995)
14. Zhou, H.-R., Zheng, P.-E.: Optimization for parrel multi-machine scheduling based on hierarchial genetic algorithm. Computer Applications, 2273–2275 (2007) (in Chinese)
15. Zhou, C., Gao, H.-B., Gao, L., et al: Particle Swarm Optimization (PSO) Algorithm. Application Research of Computers, pp. 7–11 (2003) (in Chinese)

Training ANFIS Parameters with a Quantum-behaved Particle Swarm Optimization Algorithm

Xiufang Lin[1], Jun Sun[1], Vasile Palade[2], Wei Fang[1], Xiaojun Wu[1], and Wenbo Xu[1]

[1] Key Laboratory of Advanced Control for Light Industry (Ministry of China), No. 1800, Lihu Avenue, Wuxi, Jiangsu 214122, China
[2] Department of Computer Science, University of Oxford, Parks Road, Oxford, OX1 3QD, United Kingdom
xiufang0@gmail.com, {sunjun_wx,wxfangwei,xwb_sytu}@hotmail.com, vasile.palade@cs.ox.ac.uk, wu_xiaojun@yahoo.cn

Abstract. This paper proposes a novel method for training the parameters of an adaptive network based fuzzy inference system (ANFIS). Different from previous approaches, which emphasized on the use of gradient descent (GD) methods, we employ a method based on. Quantum-behaved Particle Swarm Optimization (QPSO) for training the parameters of an ANFIS. The ANFIS trained by the proposed method is applied to nonlinear system modeling and chaotic prediction. The simulation results show that the ANFIS-QPSO method performs much better than the original ANFIS and the ANFIS-PSO method.

Keywords: Particle swarm optimization, quantum-behaved particle swarm Optimization, training algorithm, evolutionary fuzzy systems.

1 Introduction

Fuzzy systems (FSs) have been successfully applied in many areas, such as system modeling and control. To ease the design and improve system performance, many neural or statistical learning approaches that automatically generate fuzzy rules have been proposed [5]. Also, evolutionary fuzzy systems, which use evolutionary algorithms to design fuzzy systems, have been a research focus in recent years. The adaptive network based fuzzy inference system (ANFIS) [4] is a popular one of the representatives. The TSK [9] is a fuzzy system with crisp functions and has been found to be efficient complex applications [1]. It has been proved that with proper number of rules, a TSK system could approximate every plant. As such, the TSK systems are widely used in the ANFIS, which has the advantage of good applicability as it can be interpreted as local linearization modeling and conventional linear techniques for state estimation and control.

The ANFIS has both the advantages of neural networks and fuzzy systems. However, training of ANFIS parameters is one of the main issues encountered when it is applied to the real-word applications. The most of the training methods for ANFIS are based on gradient descent (GD) approach, and calculation of gradient in each step is tractable since the chain rule used may cause many local minima of the problem. The gradient method is known as a local search method and its performance generally

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 148–155, 2012.

depends on initial value of parameters, which makes it difficult to find the global optimal learning rate. Here, we propose a method which trains the parameters using quantum-behaved particle swarm optimization (QPSO). The QPSO algorithm is a variant of the particle swarm optimization (PSO) method and was inspired by quantum mechanics [10, 11]. In terms of convergence properties, QPSO is very different from PSO in that it has been proved to be global convergent [2, 12]. Many empirical studies show that QPSO has stronger global search ability when solving many continuous optimization problems.

The rest of the paper is organized as follows: In Section 2, we provide a review of ANFIS. In Section 3, the PSO and QPSO algorithms are reviewed. The proposed method is descried in Section 4. Section 5 presents how to use the proposed method for nonlinear system modeling and chaotic prediction. The paper is concluded in Section 6.

2 ANFIS Modeling

2.1 Overview of ANFIS Architecture

This section introduces the basics of ANFIS network architecture. A detailed coverage of ANFIS can be found in [4]. The ANFIS network is a neuro-fuzzy network that was proposed by Jang in 1993. For simplicity, the above mentioned system is supposed to have two inputs and one output. The rule base contains two TSK fuzzy if-then rules. The TSK fuzzy model was proposed by Takagi, Sugeno and Kang [9] in an effort to formalize a systematic approach to generate fuzzy rules from an input-output data set. A typical TSK fuzzy model with two rules may be stated as:

Rule 1: if x is A_1 and y is B_1 then $f_1 = p_1 x + q_1 y + r_1$.

Rule 2: if x is A_2 and y is B_2 then $f_2 = p_2 x + q_2 y + r_2$.

where x and y are the inputs of ANFIS, A_i and B_i are the fuzzy sets, and f_i is a first order polynomial and represents the outputs of the first order TSK fuzzy inference system. In the above rules, p_i , q_i and r_i are the parameters set, referred to as the consequent parameters.

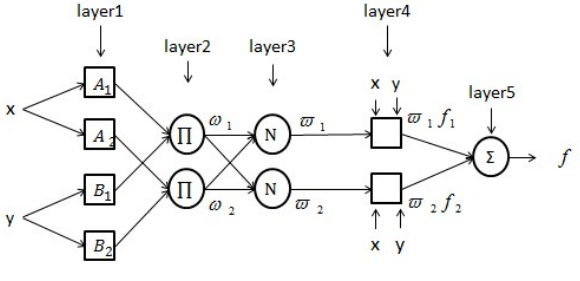


Fig. 1. The ANFIS architecture [2]

The architecture of ANFIS is shown in Fig. 1, and the nodes' function in each layer is described below.

Layer 1: This layer contains adaptive nodes with node functions described as:

$$O_i^1 = \mu_{A_i}(x) \tag{1}$$

where x is the input to node i, and A_i is the linguistic label (*small* , *large*, etc.) associated with this node function. In other words, O_i^1 is the membership function of A_i, and it specifies the degree to which the given $\boldsymbol{x}$ satisfies the quantifier A_i. Usually we choose $\mu_{A_i}(x)$ to be bell-shaped with maximum equal to 1 and minimum equal to 0, such as

$$\mu(x) = \frac{1}{1 + (\frac{x - c_i}{a_i})^{2b_i}} \tag{2}$$

where a_i, b_i and c_i are the parameter set. The bell shaped functions vary as the values of this parameter are changing. These parameters are called the premise parameters.

Layer 2: Every node in this layer is a circle node labeled $\prod$ which multiplies the incoming signals and sends the product out. For instance,

$$O_i^2 = \omega_i = \mu_{Ai}(x) \cdot \mu_{Bi}(y) \qquad for \quad i = 1,2 \tag{3}$$

Each node output represents the firing strength of a rule.

Layer 3: Every node in this layer is a fixed node, marked by a circle and labeled *N*, with the node function to normalize the firing strength by calculating the ratio of the *i*th node firing strength to the sum of all rules' firing strength.

$$O_i^3 = \varpi_i = \frac{\omega_i}{\sum \omega_i} = \frac{\omega_1}{\omega_1 + \omega_2} \qquad for \quad i = 1,2 \tag{4}$$

Layer 4: Every node in this layer is an adaptive node, marked by a square, with the node function:

$$O_i^4 = \varpi_i \cdot f_i = \varpi_i (p_i x + q_i y + r_i) \qquad for \quad i = 1,2 \tag{5}$$

Layer 5: Every node in this layer is a fixed node, the overall output can be expressed as a linear combination of the consequent parameters.

$$\begin{aligned} O_i^5 &= f_{out} = \sum_i \varpi_i \cdot f_i = \varpi_1 f_1 + \varpi_2 f_2 = \frac{\omega_1}{\omega_1 + \omega_2} f_1 = \frac{\omega_2}{\omega_1 + \omega_2} f_2 \\ &= (\varpi_1 x) p_1 + (\varpi_1 y) q_1 + (\varpi_1) r_1 + (\varpi_2 x) p_2 + (\varpi_2 y) q_2 + (\varpi_2) r_2 \end{aligned} \tag{6}$$

2.2 Hybrid Learning Algorithm

It can be seen that there are two modifiable parameter sets, $\{a_i, b_i, c_i\}$ called the premise parameters, and $\{p_i, q_i, r_i\}$ called the consequent parameters. The aim of the training algorithm for this architecture is to tune the above two parameter sets to make the ANFIS output fit the training data. Each epoch of this hybrid learning procedure is composed of two passes: a forward pass and a backward pass. In the forward pass, premise parameters are fixed and the least squares estimation (LSE) is applied to identify consequent parameters. When the optimal parameters are found, the backward pass starts with the consequent parameters fixed, the error rate of output node back-propagates from output end towards the input and the premise parameters are updated by using the gradient descent (GD) method.

Popular methods update the premise parameters by using GD or Kalman filtering and appear to be prone to trap into the local optima. In this paper, we employ the QPSO algorithm to train the parameters of the ANFIS for the purpose of obtaining the global optimal solution.

3 The PSO and the QPSO

3.1 Particle Swarm Optimization

In the original PSO proposed by Kennedy and Eberhart [7], each particle flies in a *D*-dimensional space in light of its own historical experience and of other particles in the swarm. The position of the *i*-th particle is represented as $X_i = (x_{i1}, x_{i2}...x_{iD})$. Each particle maintains a memory of its previous best position $P_i = (p_{i1}, p_{i2}...p_{iD})$, known as the personal best position. The best personal position among all the particles in the population is represented as $G = (G_1, G_2...G_D)$ and called the global best position. The velocity of each particle is represented as $V_i = (v_{i1}, v_{i2}...v_{iD})$. In each iteration, the velocity along each dimension is adjusted according to equation (7), and a new position of the particle is determined using that velocity as shown by equation (8).

$$V_{id} = wv_{id} + c_1 r_1 (P_{id} - x_{id}) + c_2 r_2 (G_d - x_{id}) \tag{7}$$

$$x_{id} = x_{id} + v_{id} \tag{8}$$

The first part of equation (7) represents the inertia of the previous velocity; the second part is the cognition part and it tells us about the personal thinking of the particle; the third part represents the cooperation among particles and is therefore named as the social component; c_1, c_2 are known as acceleration constants, r_1, r_2 are uniformly generated random numbers between 0 and 1; *w* is the inertia weight and is described in [3, 8].

3.2 Quantum-Behaved Particle Swarm Optimization

In the quantum model of PSO, the state of a particle is depicted by the wave function $\psi(x,t)$, instead of the position and the velocity. The dynamic behavior of the particle is very different from that of the particle in traditional PSO systems in that the exact values of *x* and *v* cannot be determined simultaneously. We can only learn the probability of the particle's appearing in position *x* from the probability density function $|\psi(x,t)|^2$, the form of which depends on the potential field the particle lies in. The particles in QPSO move according to the following iterative equation:

$$X_{id}(t+1) = \bar{P}_{id} \pm \alpha \cdot |C_d(t) - X_{id}(t)| \cdot \ln(1/u_{id}) \qquad u_{id}(t) \sim U(0,1) \tag{9}$$

where

$$\bar{P}_{id}(t) = \varphi_d(t) \cdot P_{id}(t) + [1 - \varphi_d(t)] \cdot G_d(t) \qquad \varphi_d(t) \sim U(0,1) \tag{10}$$

$$C_d(t) = \frac{1}{M} \sum_{i=1}^{M} P_{id}(t) \tag{11}$$

$C_d(t)$ is defined as the mean value of all particles' personal best position, u and φ are random number distributed uniformly on [0,1], respectively, and α called Contraction- Expansion Coefficient, is the only parameter in QPSO algorithm.

4 ANFIS Parameter Learning by QPSO

This section presents how to employ the QPSO algorithm for updating the ANFIS parameters. The ANFIS has two types of parameters which need training, that is, the premise parameters $\{a_i, b_i, c_i\}$ and the conclusion parameters $\{p_i, q_i, r_i\}$. The premise parameters are updated by the QPSO algorithm and the least squares estimation (LSE) is applied to identify the consequent parameters. The fitness is defined as the root mean squared error (RMSE) between the actual output and the desired output, which can be described by:

$$fitness = \sqrt{\frac{\sum_{i=1}^{n} (f(i) - f_0(i))^2}{n}}$$

Where $f(i)$ is the actual output and $f_0(i)$ is the desired output, *n* is the number of the output.

The ANFIS-QPSO algorithm is outlined below:

Step1: Initialize particles with random position; set the best position of each particle as $P_i(0) = X_i(0)$. and set *t*=0.

Step2: Set the position of each particle $X_i(0)$ as the premise parameters $\{a_i, b_i, c_i\}$ and identify consequent parameters $\{p_i, q_i, r_i\}$ with LSE. Then, calculate the fitness value of each particle and set each particle's personal best position as $P_i(0) = X_i(0)$.

Step3: Find out the mean value of all particles' personal best position $C_d(t)$ by using equation (11).

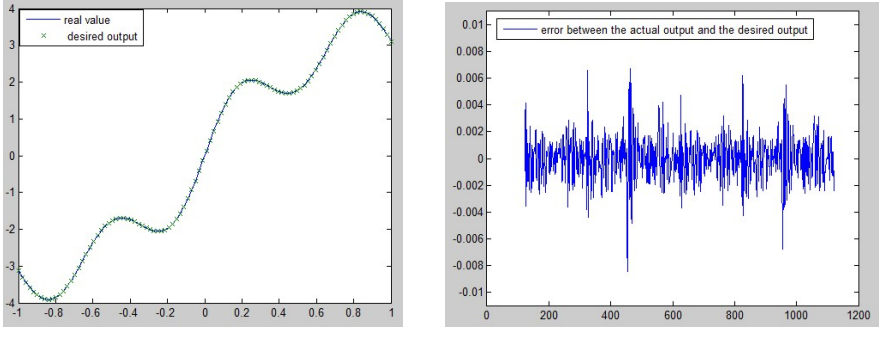
Step4: For every particle $i(1 \le i \le M)$ implement steps 5 to 7.

Step5: Calculate each particle's fitness value $fitness(X_i(t))$, and then compare it with the fitness of its personal best position, $fitness(P_i(t-1))$. If $fitness(X_i(t)) < fitness(P_i(t-1))$, then $P_i(t) = X_i(t)$; otherwise $P_i(t) = P_i(t-1)$.

Step6: Compare the fitness value of each particle's personal best position $fitness(P_i(t))$ with that of the global previous best position, $fitness(G(t-1))$. If $fitness(P(t)_i) < fitness(G(t-1))$, then $G(t) = P_i(t)$; otherwise $G(t) = G(t-1)$.

Step7: Update the position of each particle $X_i(t)$ by using equations (9) to (10).

Step8: If the termination condition is met, exit; otherwise go to Step 2, and set t=t+1.



(a) (b)

Fig. 2. Errors between the actual output and the desired output by using ANFIS-QPSO (a) the results of example 1. (b) the results of example 3.

5 Simulations

To investigate the efficiency of the proposed method, three examples are tested. The first two examples are on the identification of nonlinear systems, and the 3rd one is on the prediction of future values of a chaotic time series. The test results are compared with the results of the original ANFIS and the ANFIS-PSO method. For the PSO and QPSO, 50 particles were used with each run lasting for 10 training epochs, as listed in Table 1.

Example 1: Nonlinear single-input-single-output (SISO) System modeling [6].

In this example, the nonlinear plant described by:

$$y = \sin(\pi x) + 0.8\sin(3\pi x) + 0.2\sin(5\pi x) \tag{12}$$

where x is the only input, and we choose 100 input data randomly generated between -1 and 1. Fig. 2(a) shows the results using the ANFIS-PSO for identification.

Example 2: Nonlinear multiple-input-single-outpu(MISO)System modeling [6].

$$\begin{aligned} y = {} & \sin(\pi x_1) + 0.8\sin(3\pi x_1) + 0.2\sin(5\pi x_1) + \\ & 0.6\sin(2\pi x_1) + 0.6\sin(4\pi x_2) + 0.1\sin(5\pi x_2) + \\ & 0.2\sin(3\pi x_2) + 0.3\sin(2\pi x_3) + 0.5\sin(\pi x_3) \end{aligned} \tag{13}$$

Equation (13) describes a three input nonlinear function. From the grid points of the range $[-1,1]\times[-1,1]\times[-1,1]$ within the input space of the above function, 500 data pairs were generated.

Example 3: Prediction of future values of a chaotic time series.

$$x(t) = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \tag{14}$$

Equation (14) is also known as the chaotic Mackey-Glass differential delay equation. The initial conditions for $x(0)$ and τ are 1.2 and 17 respectively. We use four past input data for this prediction. And the fuzzy system is generated as:

$$[x(t\text{-}18), x(t\text{-}12), x(t\text{-}6), x(t): x(t+6)].$$

When t was varying from 118 to 1117, we generated 1000 data pairs for our data, and applied the first 500 data pairs for training, the last 500 data pairs for prediction. Fig.2(b) shows the errors between actual output and desired output by using ANFIS-QPSO.

In Table 1, we summarize the related parameters for the training/checking data. In order to investigate the performance statistically, the proposed method was run for 10 times independently. The RMSE for training (T-RMES) and checking(C-RMES) data of ANFIS-QPSO method are listed in Table2, and the RMSE of the original ANFIS and the ANFIS-PSO method are also listed for performance comparison. From the simulation results, we can see that ANFIS-PSO performs much better than the original ANFIS. But the method we proposed ANFIS-QPSO is more effective.

Table 1. Parameters of the 3 examples

Example	No. of Inputs	No. of MFs for each input	Training epochs	Population size in PSO/QPSO	No. of Training/ Testing data
1	1	5	10	50	50/50
2	3	5	10	50	250/250
3	4	2	10	50	500/500

Table 2. Simulation results for the 3 examples

Example	ANFIS		ANFIS-PSO		ANFIS-QPSO	
	T-RMES	C-RMES	T-RMES	C-RMES	T-RMES	C-RMES
1	1.665e-2	1.675e-2	9.796e-3	9.905e-3	6.301e-3	7.620e-3
2	3.351e-3	3.351e-3	1.711e-3	1.716e-3	1.052e-3	1.055e-3
3	2.550e-3	2.502e-3	2.073e-3	2.043e-3	1.870e-3	1.777e-3

6 Conclusion

In this paper, we proposed a novel method for training the parameters of an ANFIS network by using the quantum-behaved particle swarm optimization (QPSO) for updating the premise parameters and the LSE approach for updating the conclusion part parameters.

The effectiveness of the proposed ANFIS-QPSO method was verified by applying it to nonlinear system identification and to the prediction of a chaotic system. The simulation results show that the proposed ANFIS-QPSO method has better performance than the ANFIS-PSO and the ANFIS trained with the gradient decent method, due to the stronger global search ability of the QPSO algorithm.

References

1. Alcala, R., Casillas, J., Cordon, O., Herrera, F.: Learning TSK rule- based system from approximate ones by mean of MOGUL methodology. Grandauni of spain (2000)
2. Van den Bergh, F.: An Analysis of Particle Swarm Optimizers. PhD Thesis. University of Pretoria (2001)
3. Clerc, M., Kennedy, J.: The Particle Swarm: Explosion, Stability, and Convergence in a Multi-dimensional Complex Space. IEEE Transactions on Evolutionary Computation 6(1), 58–73 (2002)
4. Jang, J.-S.R.: ANFIS: Adaptive-Network-Based Fuzzy Inference System. IEEE Trans. Sys., Man and Cybernetics 23(3), 665–685 (1993)
5. Juang, C.F.: A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms. IEEE Trans. Fuzzy Syst. 10, 155–170 (2002)
6. Juang, C.-F., Hisao, C.-M., Hsu, C.-H.: Hierarchical Cluster-Based Multispecies Particle-Swarm Optimization for Fuzzy-System Optimization. IEEE Trans. Fuzzy Syst. 10(1), 14–26 (2010)
7. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proc. IEEE 1995 International Conference on Neural Networks, IV, Piscataway, NJ, pp. 1942–1948 (1995)
8. Shi, Y., Eberhart, R.: Empirical Study of Particle Swarm Optimization. In: Proc. 1999 Congress on Evolutionary Computation, Piscataway, NJ, pp. 1945–1950 (1999)
9. Sugeno, M., Kang, G.T.: Structure identification of fuzzy model. Fuzzy Sets and Systems 28(1), 15–33 (1998)
10. Sun, J., Feng, B., Xu, W.-B.: Particle Swarm Optimization with Particles Having Quantum Behavior. In: Proc. 2004 Congress on Evolutionary Computation, Piscataway, NJ, pp. 325–331 (2004)
11. Sun, J., Xu, W.-B., Feng, B.: A Global Search Strategy of Quantum-behaved Particle Swarm Optimization. In: Proc. 2004 IEEE Conference on Cybernetics and Intelligent Systems, Singapore, pp. 111–115 (2004)
12. Sun, J., Wu, X., Palade, V., Fang, W., Lai, C.H., Xu, W.: Convergence Analysis and Improvements of Quantum-behaved Particle Swarm Optimization. Information Sciences 193, 81–103 (2012)

Research on Improved Model of Loans Portfolio Optimization Based on Adaptive Particle Swarm Optimization Algorithm

Ying Sun and Yuelin Gao

Research Institute of Information and System Science,
Beifang University of Nationalities, YinChuan, 750021, China
nxsunying@126.com

Abstract. The paper establishes a decision-making model of the commercial bank's loans portfolio optimization based on complex risk weight in view of the loan enterprise's credit graduation situation and so on. It is more similar with the actual operation. In order to solve this model that is a non-linear 0-1 fractional integer programming question, we present a adaptive particle swarm optimization (APSO) algorithm. It is shown with the numerical result that this algorithm is effective for solving commercial bank's loans portfolio decision-making problem. The algorithm can solve the middle-scale question and the given model is reasonable.

Keywords: Loans portfolio optimization, Risk weight, Non-linear 0-1 fractional programming, Adaptive particle swarm optimization (APSO).

1 Introduction

With the development of market economy, the enterprise's risk is gradually increased during the business process. The largest corporate creditors –bank, its risk will increase. For the bank, an effective loan decisions can reduce risk and increase income in a way [1]. The credit portfolio decision-making discussed in the paper is based on the modern property portfolio theory and selects a group of appropriate loan objects from the multitudinous application loan objects by comprehensively considering loan income-risk and enterprise's credit graduation information as well as makes use of the optimization techniques.

At present, for the domestic and foreign commercial bank, the choice between risk and income under the indefinite investment condition is approximately divided into two kinds[2]: One kind makes use of control principle to choose; Two kind makes use of Sharp Index method to choose. However two methods both exist in certain limitations, therefore to cause people to seek one more feasible method to studies it. According to the decision-making model of loan portfolio optimization[3]and[4] based on the unit risk income biggest principle and fully considering of enterprise's credit graduation situation, the paper gives a decision-making model of loans portfolio optimization based on composite risk weight.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 156–163, 2012.

The proposed model is essentially a non-linear 0-1fraction integer programming problem. The problem can be easily solved in small scale by the enumeration method. But with the scale increasing, its computation quantity presents index growth and the general traditional methods will be unable to solve it. So in the paper, we propose an adaptive particle swarm optimization. The simulation experiment indicates that the algorithm has a very good effect to solve the decision-making problem of loans portfolio optimization, no matter seeking the superior ability, or the computational speed and the stability property.

2 Decision-Making Principle of Loans Portfolio Optimization

Commercial bank's management objective is the enhancement of economic efficiency. Therefore, in view of loans portfolio, we need consider the following several basic principles[3]and[5]:

(1) Comprehensive risk bearing capacity principle. One, the loan decision-making must make bank withstand risk reasonably. Two, the loan decision-making must consider the size of loans portfolio risk.
(2) The smallest principle of loan surplus resources. 0-1integer programming question is a good modelling method to enable the resources to be used fully.
(3) Comparability and uniform principle. Using the total net present value method may realize this principle well.
(4) Single loan limitation principle. The bank provides the single loan limit amount to avoid more money loan to a single enterprise.

3 A New Decision-Making Model of Loans Portfolio Optimization

Suppose that m is the application loan enterprise's number; $TPNV_i (i = 0,\ldots,m)$ is the total net present value of ith enterprise's newly built project; X_j is a 0-1 variable, $X_i = 0$ expresses ith loan enterprise is not selected, $X_i = 1$ expresses the loan enterprise is selected; $TPNV$ is the total benefit of loans portfolio. According to the risk recognition situation of the bank in the actual operation, the risk size is mainly related with the loan object (expressed by the enterprise's credit rank) and the loan way, and the loan deadline as well as the loan shape. Its risk weight follows:

ith risk weigh of loan object = credit rank coefficient $\times$ way risk coefficient $\times$ deadline risk coefficient $\times$ shape risk coefficient, Namely:

$$W_i = \prod_{j=1}^{4} W_{ij}$$

Suppose σ for the standard deviation of loans portfolio, and it weighs the total risk of loans portfolio. $Cov(W_i TNPV_i X_i, W_j TNPV_j X_j)$ presents the covariance between ith loan object and jth loan object, namely their portfolio risk ($X_i = 0$, namely as

Table 1. Loan risk adjustment weight

Credit rank	W_{i1}	loan way	W_{i2}	Loan deadline	W_{i3}	loan shape	W_{i4}
AAA	0.1	national debt pawns	0.05	≤0.25	1.00	normal	1.0
AA	0..2	this bank business regular deposit slip pawns	0.1	>0.25 and≤0.5	1.05	attention	1.2
A	0.3	A level financial organ guarantee	0.2	>0.5and≤1	1.10	secondary	1.8
BBB	0.5	foreign capital or Chinese-foreign joint venture bank credit	0.3	>1 and≤3	1.3	Suspicious	2.2
BB	0.6	state-owned commercial bank guarantees	0.4	>3 and≤5	1.40	Loss	2.5
B	0.6	house and other building mortgage	0.6	>5	1.60		
CCC	0.8	he domestic market customer guarantee	0.7				
CC	1	company share mortgage above A level	0.8				
C	1	special-purpose mechanical device mortgage	0.9				
D	1	credit loan	1.0				

Note: the data in table 1 come from Reference [5],[6],[7]

the ith loan object is not selected, the covariance is 0 between it and jth loan object).Therefore the total risk of loans portfolio is:

$$\sigma = |\sum_{i=1}^{m}\sum_{j=1}^{m} Cov(W_i TNPV_i X_i, W_j TNPV_j X_j)|^{1/2}$$
$$= |\sum_{i=1}^{m}\sum_{j=1}^{m} W_i X_i W_j X_j Cov(TNPV_i, TNPV_j)|^{1/2}$$

The total expected income of loans portfolio is:

$$TNPV = \sum_{i=1}^{m} TNPV_i X_i$$

then, the objective function is:

$$\max \quad \frac{TNPV}{\sigma}$$

Let the used loan cash L; In order to satisfy the principle (2), suppose the lowest bank 's loan amount is L_a; L_i is the loan money sum applied by the ith loan object. Then the restraint function follows as:

$$L_a \le \sum_{i=1}^{m} L_i X_i \le L$$

Therefore, the loans portfolio optimization decision-making model is:

$$\begin{aligned} &\max \quad \frac{TNPV}{\sigma} \\ &\text{s.t.} \quad L_a \le \sum_{i=1}^{m} L_i X_i \le L \\ &\qquad X_i = \begin{cases} 0, & i^{\text{th}} \text{ company is not selected} \\ 1, & i^{\text{th}} \text{ company is selected} \end{cases} \end{aligned} \tag{1}$$

4 The Description of Adaptive Particle Swarm Optimization Algorithm

Suppose that N is the number of a particle swarm, m is the number of enterprise, so let m be the dimension of binary code, $x_i = (x_{i1}, \cdots, x_{im})$ is noted as the current position of the i^{th} particle of swarm. Different position x_i corresponding to different individual fitness function f_i (fitness function is objective function in this paper)that related to optimized objective function.

In view of the upper and lower bound's constraints of the model, for the loans portfolio that surpass the upper or the lower bound's constraint, for principle (4), we will make the following transformation and turn them into the feasible solution. （ⅰ）When the total loans surpass the redundancy loan cash L, we make $x_{ij_1} = 0$, x_{ij_1} express the loan enterprise with biggest amount. If it till surpass, we make $x_{ij_2} = 0$, x_{ij_2} express the loan enterprise with second big amount, and so no, until it satisfies L;（ⅱ）When the total loans have not achieved the lowest loan amount L_a ,we make $x_{ik_1} = 1$, x_{ik_1} express the loan enterprise with smallest amount in the enterprises that have not been loaned. If it till not achieve, we make $x_{ik_2} = 1$, x_{ik_2} express the loan enterprise with second small amount, and so no, until it satisfies L_a. Do this, we may avoid massive loans concentrate in few enterprises and make loans disperse to many enterprises. It conforms to the bank's requirement of diversification of risk.

Each particle's position vector x_i corresponds with a loan portfolio, for i^{th} particle of swarm, $x_i = [x_{i1}, x_{i2}, \cdots, x_{1m}]$ $(x_{ij} \in \{0,1\}, \quad i = 1,2,\cdots,N, j = 1,2,\cdots,m)$ is particle's position vector, and $x_{pi} = [x_{pi1} x_{pi2}, \cdots x_{pim}]$ is noted as the best position by which it has ever visited. g is noted as the index of best particle among the particles in the population. $x_g = [x_{g1}, x_{g2}, \cdots, x_{gm}]$ is noted as the best position by which the swarm have ever visited, the rate of the velocity is represented as $v_i = [v_{i1}, v_{i2}, \cdots v_{im}]$. In discrete PSO algorithm, the particles are manipulated according to the equation:

$$v_{ij}^{(t+1)} = \omega^{(t)} v_{ij}^{(t)} + c_1 r_{1ij}^{(t)} (x_{pij}^{(t)} - x_{ij}^{(t)}) + c_2 r_{2ij}^{(t)} (x_{gj}^{(t)} - x_{ij}^{(t)}) \tag{2}$$

$$x_{ij}^{(t+1)} = \begin{cases} 0, \rho \le sig(v_{ij}^{(t+1)}) \\ 1, \rho > sig(v_{ij}^{(t+1)}) \end{cases} \tag{3}$$

Where t expresses the t^{th} iteration; $r_{1ij}^{(t)}$ and $r_{2ij}^{(t)}$ are random numbers uniformly distributed in the range $[0,1]$; c_1 and c_2 are positive constants, called the cognitive and social parameter respectively, both equal to [0,2] in general cases; $sig(\cdot)$ expresses Sigmoid function, in this paper, let $sig(x) = 1/[1+\exp(-x)]$;let the upper limit of the velocity of particle be $v_{\max} = 6$, $-v_{\max} \le v_{ij}^{(t)} \le v_{\max}$,so $0.0025 \le sig(v_{ij}^{(t)}) \le 0.9975$; $\omega^{(t)}$ is called inertia weight, we will use a kind of dynamically changing inertia weight.

A kind of dynamically changing inertia weight [7]. The inertia weight is decided by the fitness value aggregation degree and space position aggregation degree of the particle swarm. The inertia weight ω should increase along with the fitness value aggregation degree increase, and reduce along with the space position aggregation degree reduce, It may express:

$$\omega = \omega_{ini} + s\omega_s - h\omega_h \tag{4}$$

Where ω_{ini} is the initial solution of ω, $\omega_{ini} = 1$ in general cases; the fitness value aggregation degree $s = \bar{\sigma}^2 = \frac{\sigma_t^2}{\sigma_{\max}^2}$, $\sigma_t^2 = \sum_{i=1}^{N} (f_i^t - f_{avg}^t)^2$, $\sigma_{\max}^2 = \max_{1 \le j \le t} \{\sigma_j^2\}$; the space position aggregation degree $h = \bar{D} = \frac{D_E^t}{\max_{0 \le k \le t}(D_E^t)}$, $D_E^t = \max_{1 \le a,b \le N} \sqrt{\sum_{j=1}^{m} (x_{aj}^{(t)} - x_{bj}^{(t)})}$, $s \in [0,1], h \in [0,1]$, therefore $\omega \in [\omega_{ini} - \omega_h, \omega_{ini} - \omega_h]$.

Algorithm 1 Adaptive Particle Swarm Optimization (APSO)

Step1 Initialize a population of particles X_N with random position vector x_i and the velocity vector v_i, and inspect every position vector whether satisfies in the constraint conditions, if not, we can use the transformation (ⅰ)、(ⅱ)to change it, and make it satisfy the constraint conditions.

Step2 Calculate the particle's fitness, let $x_{pi} = [x_{pi1} x_{pi2}, \cdots, x_{pim}]$ is the i^{th} particle's optimal position, $x_g = [x_{g1}, x_{g2}, \cdots, x_{gm}]$ is the global optimal position.

Step3 Calculate the dynamically changing inertia weight ω based on equation (4).

Step4 Renew the position and velocities of particles based on equation (2) (3).

Step5 Inspect the new position vector whether satisfies in the constraint conditions if not, we can use the transformation (ⅰ)、(ⅱ)to change it, and make it satisfy the constraint conditions.

Step6 Calculate the particle's fitness and renew every particle's optimal position and the global optimal position.

Step7 (Termination examination) If the termination criterion is satisfied, namely, satisfy the iterations or the error band, then output he global optimal position and its fitness value. Otherwise, Loop to step 3.

5 Numerical Test and Analysis

5.1 Case

L is noted as the loan cash of some bank newly built project, the loan cash L is 3 million Yuan, the lowest loan task L_b is 2.7 million. There are ten enterprises applying for the new loan and the loan money sum is 70% of its total investment. Moreover, each enterprise loan project is the feasible plan after. Table 2 shows additional information. Now determine the bank decision-making of loans portfolio, so as to decide to provide the loan for which enterprises.

Table 2. Plan selected of loans portfolio

project	1	2	3	4	5	6	7	8	9	10
total investment	50	40	57	45	80	37.5	90	30	35	16
bank investment	35	28	39.9	31.5	56	26.25	63	21	24.5	11.2
	77.18	65.22	95.40	96.19	102.45	76.77	146.76	79.67	82.09	39.76
Total net	47.18	45.22	5.40	76.19	12.45	6.77	126.76	39.67	30.09	27.76
	17.18	25.22	-24.60	46.19	-47.55	-21.23	16.76	9.67	18.09	-0.24
mean total net	47.18	45.22	25.40	72.86	22.45	20.77	96.76	43.01	43.42	22.42

Table 3. Covariance matrix of total net present value *TNPV*

	1	2	3	4	5	6	7	8	9	10
1	600	4000	12000	500	1500	980	1300	700	640	400
2	400	266.7	800	333.3	1000	653.3	866.7	466.7	426.7	266.7
3	1200	800	2600	966.7	3100	2100	2300	1433.3	1411.7	746.7
4	500	333.3	966.7	422.2	1233.3	793.3	1133.3	577.8	511.1	342.2
5	1500	1000	3100	1233.3	3800	2520	3100	1766.7	1666.7	973.3
6	980	653.3	2100	793.3	2520	1698.7	1913.3	1166.7	1138.7	616
7	1300	866.7	2300	1133.3	3100	1913.3	3266.7	1466.7	1186.7	946.7
8	700	466.7	1433.3	577.8	1766.7	1166.7	1466.7	822.2	768.9	457.8
9	640	426.7	1411.7	511.1	1666.7	1138.7	1186.7	768.9	771.6	391.1
10	400	266.7	746.7	342.2	973.3	616	946.7	457.8	391.1	280.9

By using the Matlab soft to compile this procedure and solve to the above question, we may obtain the optimal within the accepted time, this example's result is:

$$(X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8, X_9, X_{10}) = (1,1,0,1,1,0,1,1,1,1)$$

The total loan money sum is: L=270.2 ten thousand Yuan. The unit risk income rate considering of synthesis risk achieves maximum value: $TNPV / \sigma = 0.0534$.

5.2 Analysis

In order to carry on the further confirmation to the algorithm 4.1, based on the prime data of loans portfolio's plan to be prepared to choose in table 2, we form the different loans portfolio which enterprise scale is from 20 to 80 and separately carry on the operation to it, this result likes table 4:

Table 4. Operation results of different loans scales

Scale	Optimal solution	Loan money sum	Particle number	Iteration times	Computing time (second)
10	1,1,0,1,1,0,1,1,1,1	270.2000	30	1000	20
20	1,1,1,1,1,0,1,1,1,1,1,1,1,0,0,1,0,1,1	543.2000	50	1000	50
30	1,1,1,1,0,0,1,1,1,1,1,1,1,0,0,1,1,1,1, 1,1,1,1,1,0,1,1,1,1	818.3000	70	2000	154
40	1,1,1,1,0,0,1,1,1,1,1,1,0,1,1,0,1,1,1,1, 1,1,1,1,1,0,1,1,1,1,1,1,1,0,0,1,1,1,1	1088.5000	90	2000	288
50	1,1,1,1,0,0,1,0,1,0,1,1,1,1,1,0,1,1,1,1, 1,1,1,1,1,0,1,1,1,1,1,1,1,0,0,1,1,1,1, 1,1,1,1,0,0,1,1,1,1	1350.3000	110	3000	649
60	1,1,1,1,0,0,1,1,1,0,1,1,1,1,1,1,0,1,1,1,1, 1,1,1,1,0,0,1,1,1,1,1,1,1,1,0,0,1,1,1,1, 1,1,1,1,1,0,1,1,1,1,1,1,1,1,0,0 ,1,1,1,1	1623.0000	130	3000	927

6 Conclusion

Through the actual computation process to the example, we know that the paper gives a better model and a feasible algorithm to solve the question, namely:

1. Consider the risk coefficient. On the one hand, because at present our country commercial bank mainly considers risk weight and income size in processing loan business. The prime model considers the relations between income and risk and only uses income variance to express the risk and neglects risk weight, for example: the original method does not consider the original loan situation to the loan enterprise and neglects the loan mortgage situation and so on. Therefore these reasons inevitably cause the commercial bank to make the wrong appraisal to loans. On the other hand, the consideration of risk weight may cause our research to unify well with commercial bank's actual operation and applies it more conveniently.

2. It is known through the analysis of the upper case that the proposed algorithm is feasible and computational time can be accepted and the algorithm can adapt to the middle or large-scale question. At the same time, in order to confirm its stability, we operate 50 times to the identical group data and obtain the optimal solution every time. As a result, when we use the algorithm to solve the loans portfolio optimisation question, we gain good effect no matter in the solution time or in the stability.

In summary, the decision-making model of loans portfolio optimisation based on composite risk weight and the hybrid genetic algorithm with greedy transformation proposed in the paper may help the commercial bank to carry on the quota macro-scientific policy-making in the loan business.

Acknowledgment. The work is supported by The National Natural Science Foundation of China (60962006) and the Foundation of Beifang University of Nationalities (2010Y036).

References

1. Chi, G., Qin, X., Zhu, Z.: Decision-making model of loans portfolio optimization based on principle of maximum earnings per risk. Control and Decision 15, 469–472 (2000) (in Chinese)
2. Jang, Z., Lin, X.: Financial Management. The Publishing House of Economic Science, Beijing (1994) (in Chinese)
3. Yang, Z.: Optimization model of loan portfolio based on return per unit of risk. Science-Technology and Management 12, 104–107 (2010) (in Chinese)
4. Gao, Y., Sun, Y.: A decision-making model of loans portfolio optimization based on composite risk weight. In: International Conference on Management Innovation, vol. 1, pp. 514–518. P.R.China, Shanghai (2007)
5. Jiang, L.-M.: Decision-Making of Loans Portfolio Optimization Based on Princip1e of Maximum Risk Return. Application of Statistics and Management 11, 84–88 (2005) (in Chinese)
6. Bank of China: Ministry of Education. Credit Load of Bank of China (1999) (in Chinese)
7. Duan, Y., Gao, Y., Li, J.: A new adaptive particle swarm optimization algorithm with dynamically changing inertia weigh. Intelligent Information Management Systems and Technologies 2, 245–255 (2006)

High-Dimension Optimization Problems Using Specified Particle Swarm Optimization

Penchen Chou

Department of Electrical Engineering, DaYeh University, Changhwa, Taiwan
choup@tcts.seed.net.tw

Abstract. Particle Swarm Optimization (PSO), proposed by Dr. J. Kennedy and Professor R. Eberhart in 1995, attracts many attentions to solve for a lot of real uni-modal/multi-modal optimization problems. Due to its simplicity of parameter-setting and computational efficiency, PSO becomes one of the most popular algorithms for optimization search. Since 1995, many researchers provide different algorithms to set parameters for convergence, explosion and exploitation potential of PSO. Most of the proposed methods are to find a general PSO (called Standard PSO, SPSO) for most of the benchmark problems. However, those may not be suitable to a specified problem, for example, Shaffer or Rosenbrock problems; especially the dimension of the problem is high. On the contrary, with to the difficult problem such as, Rosenbrock, a more proper specified PSO is needed for this high-dimension problem. Therefore, for each problem after more understanding the characteristic of the problem, a SPecified PSO (SPPSO) is proposed. Apply this idea to 5 benchmark problems, such as sphere, quatric, Rosenbrock, Griewank, and Rastrigin functions, four different SPPSO algorithms are proposed with good results in the end.

Keywords: Genetic algorithms, particle swarm optimization, mutation, optimization.

1 Introduction

Since 1955, J. Kennedy and professor R. Eberhart introduced the brand new optimization idea called Particle Swarm Optimization (PSO); optimization algorithms are more advanced and efficient than that before [1], [2]. Take an example, Genetic Algorithm (GA) is another optimization one based on evolution principle with high computational burden [6]. Stagnation to the local optimum is frequently occurred in the GA optimization search.

Adjustments on parameters of SPSO (Standard PSO) are under studying by many researchers all around the world since correct parameters setting is essential to PSO efficiency. As mentioned in [7], [8], many Improved PSOs (IPSOs) have been promoted to enhance the capability of original PSO, but fail to find the optimal solution when the dimensionality of the problem is high. In [7], [8], arithmetic mutation borrowed from GA with low mutation rate can improve efficiency of SPSO. In [10], forced mutation can further solve Shaffer problems in medium-high dimension. In this paper, it is found that Rosenbrock function is the most time-consuming and difficult problem if the PSO

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 164–172, 2012.

algorithm is not properly prepared. The rest four functions—Sphere, Quatric, Griewank, and Rastrigin ones [11] are more easily to solve with less parameters to be determined.

In section 2, SPSO equations are outlined and explained first with Rosenbrock function. After that in section 3, SPPSO (SPecified PSO, suggested in the article) and other problems are discussed. Sample examples with this five functions and results are outlined in section 4. The last section is the conclusions.

2 The General Formulae for PSO Algorithm

The formulae recommended by [1] and the others are listed in equation (1) and (2).

$$V_i(k+1) = \omega * V_i(k) + c_1 * \varphi_1 * \left(P_i(Pbest) - P_i(k)\right) + c_2 * \varphi_2 * (P(Gbest) - P_i(k)) \ . \quad (1)$$

$$P_i(k+1) = \omega_p * P_i(k) + V_i(k+1) \ . \quad (2)$$

Where V stands for velocity with dimension D, P stands for particle/position, P(Pbest) is the best parameter in the i-th generation, and P(Gbest) is the best solution particle array from beginning to the present generation so far. φ_1 and φ_2 are two uniform random numbers from 0 to 1. i is the generation number and k is the time step. In the original SPSO, $c_1=c_2=2$, $\omega=1$, $\omega_p=1$, or ω is decreasing from 0.9 to 0.4 linearly with iteration (generation) number or $\omega=0.5*(1+rand)$ or $c_1=c_2=2.05$, $\chi=0.7298$ [14]. P has its own lower and upper bound. V_{max} can be the same as P_{max}. In [7], [8], the best settings of parameters for modified PSO with a mutation rate of 0.0035 are $c_1=c_2=1.9$, $\omega=0.763$, $\omega_p=1$. Many other modified PSO algorithms can be found in [3], [4], [5], [9], [11], [12], [13].

It is interesting to find that for medium-high dimension Rosenbrock problems, mutation attached to SPSO is not a good choice in order to improve the shortcomings of SPSO. However, when the dimension is over 300, the GA mutation is required then. With a special mutation mechanism called one-variable mutation, SPPSO can lead the optimization process to jump out the local optima.

Rosenbrock benchmark optimization problem is described first in eq. (3).

$$f(X) = \sum_{i=1}^{D-1}[100(x_{i+1} - x_i)^2 + (1 - x_i)^2] \quad (3)$$

Where $X=[x_1, x_2,\ldots, x_D]$, $D \geqq 2$. Because x_i and x_{i+1} are tied together in values, each x_i must be closed enough to one; otherwise, f(X) cannot reach the minimum value. So for an arbitrary X value, the tendency of x_{i+1} must approach x_i in order to minimize f(X). The steps to revise next $\triangle X$ are many but $\triangle X$ is not allowed too large. The objective is to minimize f(X) in a large range of $-100 \leq x_i \leq 100$, i=1,2,…,D. The global minimum is zero at $x_i = 1 \text{ for } i = 1,2,\ldots,D$. For a high-dimension Rosenbrock, it has a narrow valley from the perceived local optima to the global optimum [12]. Fig. 1 shows the landscape near the global minimum with local minima around.

For high-dimension Rosenbrock problems, some phenomina must be investigated vastly before the determination of a proper algorithm for them. It is found that if there is no mutation added to PSO, most of the time, the optimization process will be stucked if

only PSO parameters are presented in PSO algorithm. For this reason, a particular mutation called one-variable mutation mechanism has been selected in SPPSO.

One-tenth of total population particle (position) are selected to do the mutation randomly. Only one variable can undergo the mutation process with a random value in the allowable ranges is taken place the original one. The mutation rate is fixed to 0.1. One-variable mutation is activated every 500 generations and lasted for 100 generations. This mutation has the opportunity to let positions jump out of the stuck points to newly located positions. It has been tested to find out that it is necessary for the algorithm to regain the activity to adjust mild ΔX to the global optimum. Besides that in the very first pass, ω=0.5, ω_p=1. For this setting, locations for Gbest and Pbest are appropriately distributed in the high-dimension solution space. Fitness Evaluations (FEs) value can go down quickly from a high value to a moderate one in a short time. After some e.g., 1,000 generations, we set ω=1, ω_p=0.5 (in the second pass or phase). In that way, with one variable mutation together, the best FE value (RED) can go down possibly.

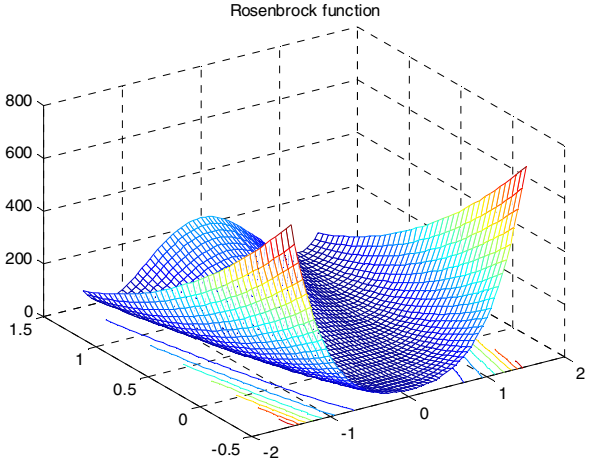


Fig. 1. Rosenbrock function plot near global optimum

In order to compare the results using SPPSO in this paper to [9], the results in [9] and the following statements are quoted here for Rosenbrock problems:

When solving the 10-D problems, the population size is set at ten and the maximum fitness evaluations (FEs) is set at 30,000 (generations). When solving the 30-dimensional (30-D) problems, the population size is set at 40 and the maximum FEs is set at 200,000. All experiments will run 30 times. The mean values and standard deviations of all results are presented in Table 1 as well. CLPSO and FDR are cited and referred in [9].

The range for this problem is arranged into two.

Range #1, $-2.048 \le x_i \le 2.048$,
Range #2, $-100 \le x_i \le 100$.

Besides that the population size is increased to 100, the maximum FEs is set at 200,000 too for dimension D≥10, for the Range #2 case.

Detailed SPPSO algorithm will be included in section 3.

Table 1. Optimization results of Rosenbrock function without shift

Number of variables (dimension n), [ave. FEs], (Success Rate) and ranges	Algorithm	Mean (standard deviation)
10, [30,000], Range #1	CLPSO	2.460(1.70)
10, [30,000], Range #1	PSO-cf	0.698(1.46)
10, [722], Range #1	SPPSO	8.54e-7(1.78e-7)
10, [723], (30/30),Range #2	SPPSO	8.75e-7(1.20e-7)
30, [200,000], Range#1	CLPSO	21 (2.98)
30, [200,000], Range #1	FDR	5.39 (1.76)
30, [14,220(6,193)], (30/30), Range #1	SPPSO	9.98e-7(4.120e-09)
30, [4,678(1,555)], (30/30), Range #2	SPPSO	9.88e-7(1.31e-08)
50, [17,133(6,272)], (30/30), Range #2	SPPSO	9.98e-7(2.63e-09)
100, [29,362(4,512)], (30/30), Range #2	SPPSO	9.54e-7(1.23e-07)
200, [58,803(9,890)], (30/30), Range #2	SPPSO	9.78e-7(1.14e-07)
400, [1.17e+5(2.06e+4)], (28/30), Range #2	SPPSO	9.99e-7(1.70e-09)

From Table 1, the suggested PSO (SPPSO) shows its great capability to find the optimal solution of high-dimension Rosenbrock problems with better results. Using SPPSO, the dimension of Rosenbrock function can be as high as 400. The utimate FEs value or goal is 1e-6. It [11] has included another five high-dimension optimization problems with dimension D= 30, 100, 400 respectively. This will be discussed in the next section. Algorithm so defined above is called SPPSO in this paper.

3 Sample Examples and Related SPPSO Algorithms

In order to compare the performances of SPPSO, five more benchmark problems in [11] are chosen. Algorithm #1 is used for Sphere and Quatric functions, algorithm #2 is for Rastrigin and Griewank functions, algorithm #3 is for Rosenbrock function only. Algorithm #4 is used for shifted Sphere function. The resolution or goal for each problem is set to 1e-6 (10^{-6}). One run will be treated as failed if the goal is not matched.

The basic sequences for SPPSO algorithm are

(1). Set c_1 and c_2 to 1.9 respectively.

(2). Renew ω and ω_p if necessary.
(3). Update vel using eq. (1).
(4). Check vel with position bounds [P_{min}, P_{max}].
(5). Update par using eq. (2).
(6). Do one-variable mutation process.
(7). Check par with position bounds [P_{min}, P_{max}].
(8). Fitness evaluation and data recording including Pbest and Gbest update.

Algorithm #1 for Sphere and Quatric function:

(1). ω=0.763, c_1=1.9, c_2=1.9, and ω_p=0. Use eq. (1).
(2). When the final vel is totally updated already, check it with position bounds. If vel is outside of position bounds, randomize a new vel in bounds to replace the outbound vel instead.
(3). Use eq. (2) to update par first. Use one-variable mutation process to do some disturbances to partial par.

One-variable mutation process:

10% of total population is chosen for mutation. Only one random variable is selected for mutation in the single par vector. This new variable value will be closed to Gbest location with a small tolerance.

(4). When the final updated par is obtained, check it with its own position bounds. If par is outside of bounds, randomize a new par in bounds to replace the outbound par. Also set the related vel to zero.

Algorithm #2 for Rastrigin and Griewank function:

(1). ω=0.763, c_1=1.9, c_2=1.9.
For every 10 iterations set ω_p=0, otherwise ω_p=rand. (Note that rand is a uniform random number between 0 and 1).
(2). The same as (2) in Algorithm #1.
(3). The same as (3) in Algorithm #1.
(4). The same as (4) in Algorithm #1.

Algorithm #3 for Rosenbrock function:

(1). c_1 is decreasing from 2.4 to 1.5, c_2=3.8-c1.
(2). In the first phase, ω=1, ω_p=0.5.
In the second phase, ω=0.5, ω_p=1.
Phase is changed after one-third of total generations (FEs).
(3). The same as (2) in Algorithm #1.
(4). The same as (3) in Algorithm #1.
(5). The same as (4) in Algorithm #1.

Algorithm #4 for shifted-Sphere function:

(1). In the first pass, ω=0.5, c_1=1.9, c_2=1.9, ω_p=1. After some proper generations, the second pass starts with ω=1, and ω_p=0.5.
(2). The same as (2) in Algorithm #1.
(3). The same as (3) in Algorithm #1.
(4). The same as (4) in Algorithm #1

Table 2 specifies the search range and initialization range as in [11]. It is recited here for convenience.

Table 2. Search range and initialization range

Function	Search range	Range of initial population
Sphere	$[-50,50]^D$	$[25,40]^D$, D=Dimension
Quatric	$[-20,20]^D$	$[10,16]^D$, SR=Success Ratio
Rosenbrock	$[-100,100]^D$	$[50,80]^D$
Griewank	$[-600,600]^D$	$[300,500]^D$
Rastrigin	$[-5.12,5.12]^D$	$[1,4.5]^D$

Table 3. Sphere function result using SPPSO (population size=50)

Dimension	FEs Mean & Std. Dev	Fitness value Mean & Std. Dev Goal=1e-6 /10^{-6}	SR
30	2.776000e+001 (6.531973e-001)	7.436449e-007 (1.172610e-007)	100/100
100	3.188000e+001 (3.265986e-001)	7.178680e-007 (1.033474e-007)	100/100
400	3.500000e+001 (0.000000e+00)	7.099101e-007 (3.224467e-008)	100/100
1,000	3.700000e+001 (0.000000e+00)	6.430027e-007 (1.763238e-008)	100/100

Table 3 lists the simulation results of testing Sphere function with different dimensions. The results are very good.

Table 4 lists the simulation results of testing Quatric function with different dimensions. The results are also very good.

Table 5 lists the simulation results of testing Rosenbrock function with different dimensions. SPPSO algorithm proposed is good at finding the true optimum of Rosenbrock function but with handful FEs and relatively long simulation time. However, good results can be expected if the setting of parameters and one-variable mechanism work properly. For a dimension of 400 for example, it will consume about half an hour (depends on computer system used, PC) to reach the desired goal.

Table 4. Quatric function result (population size=50)

Dim.	FEs	Fitness value Goal=1e-6	SR
30	1.681000e+001 (4.860685e-001)	5.745019e-007 (1.889986e-007)	100/100
100	1.685000e+001 (4.351941e-001)	5.695947e-007 (1.851827e-007)	100/100
400	1.693000e+001 (4.083720e-001)	5.673559e-007 (1.971024e-007)	100/100
1,000	2.458000e+001 (4.960450e-001)	6.108190e-007 (3.072196e-007)	100/100

Table 5. Rosenbrock function result (population size=100)

Dim.	FEs	Fitness value Goal=1e-6	SR
30	1.649143e+004 (7.292718e+003)	9.157303e-007 (2.255262e-007)	100/100
100	2.927773e+004 (5.153857e+003)	9.228321e-007 (2.138867e-007)	100/100
400	1.087893e+005 (2.797335e+004)	9.003438e-007 (2.123913e-007)	30/30
600	1.762305e+005 (2.038327e+004)	9.996882e-007 (3.687322e-010)	6/6, 1.33 hours
1000	3.418955e+005 (2.202567e+004)	9.993987e-007 (7.226745e-010)	2/2, 5.81hours

Table 6 lists the simulation results of testing Griewank function with different dimensions. The results are good.

If there is an arbitrary shift of the desired Sphere function, does the Algorithm #1 still working or not. The answer is no. Algorithm #1 is required to be modified and becomes Algorithm #4 as shown above. From Table 7, two things can be investigated:

(1). Average FEs is significantly increased for each run.

(2). Dimension is decreased slightly since more execution time is required for each run. Parameter settings might not be the best so far.

Algorithm #1 is okay to the Sphere function without any shift. With shift, the SPPSO will be more complicated and the finding of global optimum of each run becomes difficult if parameters are not setting correctly. Comparing Table 3 to Table 7, it seems that complicated SPPSO is required to overcome the difficulty of the shift of the final optimum point in Sphere function.

Table 8 lists the simulation results of testing Rastrigin function without shift with different dimensions. Although there are a bunch of local minima everywhere inside, SPPSO algorithm is easy to trace the global optimum in relatively few generations.

Table 6. Griewank function result (population size=50)

Dim.	FEs	Fitness value Goal=1e-6	SR
30	5.438000e+001 (1.845758e+001)	5.443579e-007 (2.820364e-007)	100/100
100	5.054000e+001 (2.132596e+001)	5.198004e-007 (2.854481e-007)	100/100
400	5.470000e+001 (4.068765e+001)	5.772870e-007 (2.776276e-007)	100/100
1000	7.281000e+001 (9.976859e+001)	5.105688e-007 (2.682491e-007)	100/100

4 Conclusions

Finding a standard PSO (SPSO, for example, SPSO-2011 in [12]) for all benchmark optimization functions is not an easy task. Even such an algorithm exists; the

resolution for solving each problem varies with each characteristic of problem. On the other hand, specified SPPSO established for each problem after some investigation on its property of the problem will be a good approach to find optimum of each function with different dimension.

Table 7. Result of Sphere function with arbitrary shift

Dim./ Population Size	FEs	Fitness value Goal=1e-6	SR
5/50	2.301700e+002 (2.717733e+001)	7.137398e-007 (2.088383e-007)	100/100
10/50	3.558000e+002 (1.968138e+002)	8.511339e-007 (1.351048e-007)	100/100
30/50	3.197041e+003 (6.421664e+003)	9.441585e-007 (5.186717e-008)	97/100
50/50	7.531540e+003 (8.194308e+003)	9.588646e-007 (4.822870e-008)	100/100
100/100	1.185577e+004 (2.430991e+003)	9.772421e-007 (2.943015e-008)	92/100
200/100	3.127266e+004 (5.668451e+003)	9.825248e-007 (3.316641e-008)	98/100
300/100	4.057310e+004 (1.288369e+004)	9.737880e-007 (4.292352e-008)	10/10
500/100	6.276250e+004 (7.023671e+003)	9.958272e-007 (4.405973e-009)	10/10
600/100	7.200833e+004 (1.616333e+004)	9.986374e-007 (1.244765e-009)	9/10
700/100	9.488929e+004 (1.189994e+004)	9.985731e-007 (1.758570e-009)	7/10

Table 8. Rastrigin function result (population size=50)

Dim.	FEs	Fitness value Goal=1e-6	SR
30	8.385000e+001 (1.573934e+002)	5.511544e-007 (2.726850e-007)	100/100
100	5.997000e+001 (1.493451e+001)	5.586667e-007 (2.668969e-007)	100/100
400	6.603000e+001 (2.138746e+001)	5.158390e-007 (2.851966e-007)	100/100
1000	8.120000e+001 (6.027596e+001)	5.661843e-007 (2.767283e-007)	100/100

Without any shift on some function such as Sphere, Rastrigin, and Rosenbrock, the test is good for high dimension. In [11], high-dimension (e.g., 30, 100, and 400) benchmark problems and rotated PSO was suggested to solve five functions such as, Sphere, Quatric, Rosenbrock, Griewank and Rastrigin functions. Frankly speaking, the only difficult problem is the Rosenbrock function, the resolution was about 1e+3

with D=400, FEs=2.5e+4 in [11]. With enough FEs, SPPSO algorithm can match the goal of 1e-6 using specified two-phase ω and ω_p setting plus one-variable mutation mechanism. Different PSO parameter–setting for different property of each optimization problem might be a truly way to solve specified problems in the future instead of finding a generalized PSO (SPSO, for example) for all problems. As for functions with shifts, the original SPPSO won't work easily for each problem. Due to the time constraint of the paper, only shifted Sphere function is completed under investigation with some partial result as shown in Table 7. The success rate decreases with increasing dimension, the execution time is more than expected. Further study is apparently required in the future for shifted optimization problems.

References

1. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: Proceedings of the IEEE International Joint Conference on Neural Networks, pp. 1942–1948 (1995)
2. Engelbrecht, A.: Computational intelligence, 2nd edn. John Wiley & Sons, West Sussex (2007)
3. Shi, Y., Eberhart, R.: A Modified Particle Swarm Optimizer. In: The 1998 IEEE International Conference on Evolutionary Computation Proceedings, IEEE World Congress on Computational Intelligence, pp. 69–73 (1998)
4. Cai, X., Cui, Z., Zeng, J., Tan, Y.: Individual Parameter Selection Strategy for Particle Swarm Optimization: Particle Swarm Optimization. InTech Education and Publishing (2009)
5. Vesterstrom, J., Thomsen, R.: A Comparative Study of Differential Evolution, Particle Swarm Optimization, and Evolutionary Algorithms on Numerical Benchmark Problems. In: Proc. IEEE Congress Evolutionary Computation, Portland, OR, June 20-23, pp. 1980–1987 (2004)
6. Chou, P.: Principles and Applications of Genetic Algorithms. ChiamHwa Book Company, Taipei (2006) (in Chinese)
7. Chou, P.: Improved Particle Swarm Optimization with Mutation. In: MSI-2009 IASTED International Conference, Beijing, China (2009)
8. Chou, P.: A Proposal for Improved Particle Swarm Intelligence. In: IEEE-3CA International Conference 2010, Tainan, Taiwan (2010)
9. Liang, J., Qin, A., Suganthan, P., Baskar, S.: Comprehensive Learning Particle Swarm Optimizer for Global Optimization of Multimodal Functions. IEEE Transactions on Evolutionary Computation 10(3), 281–295 (2006)
10. Chou, P., Lian, J.: Enforced Mutation to Enhancing the Capability of Particle Swarm Optimization Algorithms. In: The Second International Conference on Swarm Intelligence, Chongqiang, China, pp. 12–15 (June 2011)
11. Hatanaka, T., Korenaga, T., Kondo, N., Uosaki, N.: Search Performance for PSO in High Dimensional Spaces Particle Swarm Optimization: InTech open (2009)
12. Clerc, M.: A Mini-benchmark, `http://clercmaurice.free.fr/pso/`
13. Yeh, Y.: A Simple Hybrid Particle Swarm Optimization: Advances in Evolutionary Algorithms. I-Tech Education and Publishing, Vienna (2008)
14. Kennedy, J., Clerc, M.: The Particle Swarm: Explosion, Stability and Convergence in a Multidimensional Complex Space. IEEE Transactions on Evolutionary Computation 6(1), 58–73 (2002)

A Novel Simple Candidate Set Method for Symmetric TSP and Its Application in MAX-MIN Ant System

Miao Deng[1], Jihong Zhang[2], Yongsheng Liang[2], Guangming Lin[2], and Wei Liu[2]

[1] School of Information Engineering, Shenzhen University, Shenzhen, China
[2] Shenzhen Key Lab of Visual Media Processing and Transmission,
Shenzhen Institute of Information Technology, Shenzhen, China
deng_miao@live.cn, lingm@sziit.com.cn

Abstract. Traveling Salesman Problem (TSP) is a kind of typical NP problem and has been extensively researched in combinatorial optimization. For solving it more effectively, candidate set is used in many algorithms in order to limit the selecting range when choosing next city to move, such as in Ant Systems, or to initialize a local optimum solution, such as in Lin-Kernighan Heuristic (LKH) algorithm. A novel simple method for generating candidate set is proposed in this paper and applied into MAX-MIN Ant System (MMAS) for symmetric TSP problem. Experimental results show that it has better performance than other Ant Systems including MMAS. Moreover, this method can be used in other algorithms for symmetric TSP problem.

Keywords: Symmetric TSP, Ant Colony Optimization, MMAS, Candidate Set.

1 Introduction

The TSP is to find an optimal solution which has the shortest tour over n given cities and each city can only exist in the tour once. It is a well-known NP-hard problem and extensively studied in combinatorial optimization. Many methods have been presented in these years such as Genetic Algorithm (GA), Neural Network, Particle Swarm Optimization (PSO), Ant Colony Optimization Algorithm (ACO)[1], Lin-Kernighan Heuristic Algorithm (LKH), etc. ACO is an efficient method for solving TSP, which is based on the pheromone communication by ants. When using ACO, a problem may occur that the unwilling long routes may turn out to lay useless pheromone, so a Give-up Ant System (GAS) is proposed in [2], it lets the ant only can move to the cities under a fixed radius near current city. But it will not work well in the case that some cities distribute very densely while other some distribute very sparsely.

Another problem of ACO is that its computational complex since it contains much float computation when calculating the probability, so candidate set methods are employed. Most commonly used is nearest-neighbors (NN) method[5], which always limits an ant moves to several nearest cities around it. Recently, [10] presented that a dynamic size of candidate set should be used according to the number of cities. But these methods can't fit the situation of somewhere dense and other somewhere sparse well.

In the LKH algorithm, a candidate set method called α-measure is employed to generate a local optimum at start[7]. Given the cost of a minimum 1-tree, the α-value

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 173–181, 2012.

of an edge is the increase of the cost of a minimum 1-tree. The α-value provides a good estimate of the edge's chance to belong to an optimum tour, but using this candidate set as a searching strategy is not inferred in LKH. There are several other candidate set methods including: 1). Delaunay distance[11] , 2). Quadrant distance[5], etc. Of these candidate set methods, few are introduced to ACO's searching process except nearest-neighbors and fixed distance radius[2] as known to our limited knowledge.

In order to define a better limited range for ants' path selection, and investigate the further usage of candidate set method in ACO, we propose a simple candidate set method and apply it into MAX-MIN Ant System (MMAS) in this paper. First we introduce a new distance measure we call it Priority Value (PV), which represents a measure of linking probability of two cities, it means that the ant can only move onto the edges that have the best PV values. Then we apply the PV into MMAS and get a new ant system called PVMMAS. In this paper, we verify the validity of the new candidate method, the PV method, by experiments with the PVMMAS and compare it with other Ant Systems. Note that we only discuss within the scope of symmetric TSP.

The paper is organized as follows: In section 2, we have a review on Ant System (AS) and MAX-MIN Ant System (MMAS). In section 3, we propose the new distance measure PV, describe the main principals of PVMMAS. In section 4, Experimental results and analysis are given. Finally, concluding remarks are placed in Section 5.

2 Ant Colony Optimization

2.1 Ant System

Ant System is a meta-heuristic algorithm[1], it takes the advantage of real ants pheromone mechanism. An ant always selects the way according to the pheromone laid on it by predecessors and the distance of it. The transition probability P_{ij}^k from city i to city j for the k^{th} ant is defined as:

$$P_{ij}^k = \begin{cases} \dfrac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{k \in allowed_k} \tau_{ik}^{\alpha}(t)\eta_{ik}^{\beta}(t)}, if \ \ j \in allowed_k \\ 0, \qquad\qquad\qquad Otherwise \end{cases} \tag{1}$$

where

$$\eta_{ij} = 1/d_{ij} \tag{2}$$

is the visibility information generally taken as the inverse of the distance between city i and city j. $allowed_k$ is the cities that the k^{th} ant can move to. τ_{ij} represents the amount of pheromone on the edge between city i and city j. Two real positive values α and β are adjustable parameters that control the relative weights of pheromone intensity and inter-city distance.

After every ant completes its tour, the pheromone trails of each path are updated according to the formulas:

$$\tau_{ij}(t+n) = \rho \cdot \tau_{ij}(t) + \Delta\tau_{ij} \tag{3}$$

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{4}$$

$$\Delta\tau_{ij}^{k} = Q / d_{ij} \tag{5}$$

where $\rho(0<\rho<1)$ represents an evaporation rate of pheromone, $\Delta\tau_{ij}^{k}$ is the pheromone which is laid on edge *(i,j)* by ant *k*, *m* is the total number of ants, and *Q* is a constant. Detail steps of AS algorithm can be found in [1].

2.2 MAX-MIN Ant System (MMAS)

MAX-MIN ant system[5] is the most widely used ACO algorithm with the limit of pheromone trail, which introduces four main modifications in AS. They can be briefly summarized as follows:

1. When updating pheromone, deposition is only allowed by the iteration best ant or by the best-so-far ant;
2. To avoid the early convergence to a suboptimal solution, the algorithm limits the pheromone trail values in the range $[\tau_{min},\tau_{max}]$. The value of τ_{min} and τ_{max} is conducted in [5];
3. The pheromone trails are initialized to upper pheromone trail limits at the start of the search, and a small evaporation rate is used;
4. When no better tour is found in a predefined number of steps, all pheromone trails are set to τ_{max}.

3 Proposed Method: PV and PVMMAS

In this section, we firstly propose a new distance measure, then generate candidate set and make the limitation for ants' selecting range according to the new distance measure. Furthermore, a new path probability calculation method is introduced in this section. These two improvements compose the new algorithm PVMMAS.

3.1 A New Simple Candidate Set Criterion

During solving symmetric TSP with ACO, an ant usually considers that the shorter distance between the candidate city and current city is, the more likely that city is to be selected as the target. But actually, it does not seem so. Because the tour can be reversed, suppose if an ant is at the other end of the chosen edge and looking for a city to move, it is not certain that the result will be the same edge. For instance, consider the case Fig. 1, in which the edge (A,B) is a long path of city A comparing with its neighbors, according to ACO, ant at A will hardly select this edge to go, especially when d_{AB} is much larger than d_{AC}. But actually, it's the shortest path of city B, so these two cities are very likely to connect at this aspect. Hence, it's necessary to consider a new criterion on selecting target city.

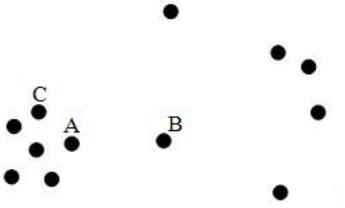


Fig. 1. Case of path selection between city A and city B

Thinking forward, consider two cities: A and B. We found that if A is the shortest city near B, and B is also the shortest city near A, then they are very likely to link together. Extending this relationship, we get a new criterion called Priority Value (PV). First, let T_i be the sorted sequence of distances between city i and other cities by sorting distances ascend, and then PV value between city i and city j is defined as:

$$PV_{ij} = \min\left(I_{ij}, I_{ji}\right) - \frac{1}{\max\left(I_{ij}, I_{ji}\right)} \tag{6}$$

where I_{ij} is the city j's index in T_i.

It can be concluded that actually PV represents a sense of mutual selective priority measure between two cities, the smaller it is, the more probably the two cities are linked together. The smallest value of PV is 0, which means the two cities are most likely to be connected.

A new candidate set can be organized by choosing the smallest PV values in unvisited cities for every ant. The maximum selection range here we define it an integer U, which is a small integer and its value is analyzed later. For ant k, let u_i^k be an integer within range [*1 U*], it represents the selection of ant k at i^{th} step in the U unvisited cities that have smallest PV values. Obviously, in an optimal tour, u_i^k is more usually a lower number than larger ones. Examples are shown in Table 1, here we take the best-known solution of instance *eil51* and *kroA100* for analysis.

Table 1. Examples Of u_i^k's Value in Best-known Solution

Instance	**Occurrence time of** u_i^k							
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*
eil51	38	10	2	0	1	0	0	0
kroA100	84	8	2	2	0	1	0	1

3.2 The Maximum Selection Range *U and* Maximum PV Limitation

The value of U can be set as belowing some constant, some kind like the number of neighbors in NN, but the difference is that the former can obtain a dynamic size of candidate set, since some candidates may have the same PV value.

An optional parameter P_{max} is defined here, to limit the upper bound of the PV value. That is, if the edge has larger PV than the predefined threshold, it will not be

added to the candidate set. The value of P_{max} is related to the cities' distribution, not the number of cities, practically we can set P_{max} a common constant.

The commonly used value of U and P_{max} can be inferred by some best-known solutions. Note that, if the total number of cities is n, then an route can be represented by $2n$ different ways, according to different origin cites and different directions. By analyzing each of the $2n$ sequences infered from the best-known solution, examples of the range of U and the value of P_{max} in some instances are listed in Table 2. The value of U can be set to its lower bound, but it will have few chance to obtain best solution, so we set it a larger value, generally, equals to or above 8 is reasonable. In this paper, we set it equals 10 as default, which has enough probability to obtain best solution, as well as fast speed to converge. Although some instances' best-known solution has a larger P_{max} like *d198*, by our experience, we can set P_{max} as 15 by default, observing that it can lead a fast convergence speed and better performance rather than setting it larger.

Table 2. Examples of Value Range Of U and *Value Of* P_{max} in Best-known Solutions

Instance	Range Of U	P_{max}
eil51	[3,6]	5.83
kroA100	[4,18]	13.96
d198	[6,36]	30.99
lin318	[5,19]	14.95

3.3 A New Visibility Information

In Ant Systems for symmetric TSP, the probability for path selection is related to the path's distance, but according to our analysis above, this is not very reasonable. The probability should consider the both cities' side on the edge, so we have a modification for the visibility information, let the eq.(2) be:

$$\eta_{ij} = \frac{1}{I_{ij} \cdot I_{ji}} \tag{7}$$

In the new visibility information equation, the distance are replaced with the sorted index on both side of the edge, so it can make the differences between densely distributed cities larger.

3.4 PVMMAS

We propose a new method called PVMMAS, which applies the PV modal into MAX-MIN Ant System. Based on MMAS in [5], the main modifications and their reasons are listed as below:

1. After initialized, all PV values are calculated according to eq. (6), and form a sorted list for every city;
2. On every step, when an ant selects its target city to move, only no more than U cities in its unvisited list that have smallest PV values as well as having less PV values than P_{max} can be added into the candidate set. If no city is added, then select the edge that has largest total information to move. After the candidate set is build, use the probability equation modified below instead of eq. (1):

$$P_{ij}^{k}=\begin{cases}\dfrac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{s\in U\_PV^{k}}\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}, if \;\; j\in U\_PV^{k}\\ 0, \qquad\qquad Otherwise\end{cases} \tag{8}$$

where the U_PV^k is the candidate set of the k^{th} ant.

And the total information is:

$$T=\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t) \tag{9}$$

3. Use the modified visibility information eq. (7) for η_{ij} in eq. (8).

4 Experimental Results and Analysis

The proposed method was applied to several instances opened by TSPLIB[6], they are *eil51*, *kroA100*, *d198* and *lin318*, as the same as [4-5], and *st70*, *tsp225*, as the same as [2], in order to compare with these algorithms respectively. Meanwhile, Initial cities are randomized to get statistical results. Maximum number of tour constructions is 2500*n*, here *n* is the number of cities, average over 20 runs. All the programs are modified on the base of Thomas Stützle's ASOTSP software package version 1.01[12].

4.1 Determination of Parameters

In PVMMAS, all the common parameters and settings are set as the same as MMAS in [5] by default, that is $\alpha=1$, $\beta=2$, $Q=1$, $m=n$, $\tau_0=\tau_{max}$, where m is the number of ants. Updating the pheromone by the iteration best, and the global best every 25 iterations. When tour is not improved more than 250 iterations, it starts the reset of pheromones to τ_{max}. For the two new parameters, we set U=10, P_{max} =15 as default.

4.2 Comparison with Other Ant algorithms

In this section, we compare the performance of the PVMMAS with some other algorithms for some symmetric TSP instances.

The computational results in Table 4 show that PVMMAS outperforms MMAS and NMMAS, with lower average results and smaller percentage deviations from the optimal tour length. All results for MMAS and NMMAS are taken from [4] and [5].

Table 3. Experimental Results With NMMAS and MMAS

Instance	Average Results		
	PVMMAS	NMMAS	MMAS
eil51	426.25(0.07%)	427.2(0.28%)	427.6(0.38%)
kroA100	21292.4(0.03%)	21298.2(0.08%)	21320.3(0.18%)
d198	15885.7(0.60%)	15924.4(0.92%)	15972.5(1.22%)
lin318	42320.0(0.69%)	----	*43082.0(0.75%)*

Then we compare PVMMAS with GAS and ASelite, the computational results is shown in Table 5. It can be seen that PVMMAS also outperforms the other two on

both average results and percentage deviations. All results for GAS and ASelite are taken from [2].

Table 4. Experimental Results With GAS And ASelite

Instance	Average Results		
	PVMMAS	GAS	ASelite
eil51	426.3(0.07%)	428.7(0.63%)	434.1(1.9%)
st70	675.35(0.05%)	696.10(3.07%)	701.55(3.93%)
tsp225	3932.55(0.27%)	4042.35(3.23%)	4061.45(3.71%)

4.3 Comparison with α-Measure Candidate Set Method

The α-value of an edge is the increase of the cost of a minimum 1-tree as described in [7], it's adopted by LKH algorithm to generate an initial optimal solution. We apply this candidate set method into MMAS just like PVMMAS, in this case, the method that uses the new visibility information is noted $MMAS_{\alpha}$, and with the old visibility information is noted $MMAS_{\beta}$. The comparison results are listed in Table 6. It shows that a-measure based MMAS can't access better result than PVMMAS. This is due to the quality of the candidate set method in PVMMAS is more suitable.

Table 5. Computational Results With Other Candidate Set

Instance	Average Results		
	PVMMAS	$\mathbf{MMAS_{\alpha}}$	$\mathbf{MMAS_{\beta}}$
kroA100	21292.4	21316.15	21498.4
d198	15885.7	16062.3	16330.5

4.4 Average Convergence Curve

The average length of PVMMAS tours was compared with MMAS. The result is show in Fig. 2, experimented on instance *d198*. The solid line in this figure shows the average length of PVMMAS, the dotted line shows the average length of MMAS. Since PVMMAS tries to limit search space within a reasonable candidate set, it usually constructs a better initial tour. On the other hand, it has faster convergence speed than

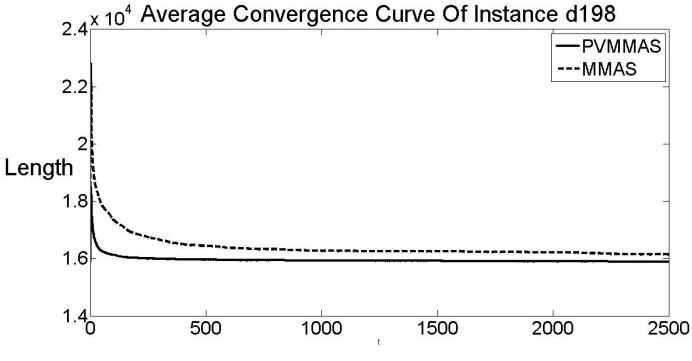


Fig. 2. Average Convergence Curve Comparison on *d198*

MMAS, which means it costs fewer time to get its optimal solution. MMAS converges at about t=500, while PVMMAS converges at about t=150.

4.5 Analysis

The running time cost compared with MMAS can be composed of two parts:

1. Extra computation. Since a mapping table can be established for eq. (6), the initial PV table calculation will only take extra computaional time $O(n^2 log_2 n)*T(comp)$, and sorting will take extra computaional time $nO(nlog_2 n)*T(comp)$, where n is the number of cities, $T(comp)$ is time cost of one comparison operation.
2. Saved computation. Since a mapping table can be established for $\eta_{ij}^{\beta}(t)$, so totally $n^2NM*(T(div)+T(exp))$ time cost is saved, where $T(div)$ is time cost of one division operation, $T(exp)$ is time cost of one exponential operation, N is the number of neighbors in NN method, M is the number of circles.

Put these two parts together, we can see that the PVMMAS can be more efficient than nearest-neighbors MMAS due to the significant saved time in part 2.

From the results of experiments and analysis, it can be seen the PVMMAS gets better performance than other Ant Systems in both speed and quality of solution.

About the parameters P_{max} or U, if they are not large enough, the PVMMAS may by little chance be stagnant in local optimal. However, increase the P_{max} and U doesn't guarantee to get global best solution neither, because it will get slower to converge. Hence, for most instances, the common settings can be adopted.

5 Conclusions

A novel candidate set method is proposed in this paper called Priority Value (PV), which takes the mutual selective priority for two cities' distance measure, then applied into the MMAS algorithm, called PVMMAS. The advantages can be listed as follows:

1. The quality of solution is more notable than other Ant Systems.
2. Easy to implement.
3. The PVMMAS converges earlier than NN based MMAS.
4. Running speed is faster than NN based MMAS.

Two parameters are added in PVMMAS: the selecting range U and the maximum PV value P_{max}. By analyzing the already known best resolutions, the common settings can be set as U=10 and P_{max} =15 for all instances.

Moreover, PV measure can be applied into ohter kinds of Ant System, to get better performance. How to improve the performance and apply it with other algorithms is the next work to do.

Acknowledgements. This work was supported by the National Natural Science Foundation of China (No.61172165 and No. 71101096), the Natural Science Foundation of Guangdong Province (No. S201101000849 and No. 9151001002000014), Shenzhen Scientific Research Project (No. JC201006020807A), Research Project of SZIIT (No. CXTD2-005 and BC2009014). Thanks especially give to the anonymous reviewers for their valuable comments.

References

1. Marco, D., Vittorio, M., Alberto, C.: Ant System: Optimization by a Colony of Cooperating Agents. IEEE Transactions on Systems, Man, and Cybernetics–Part B 26(1), 1–13 (1996)
2. Akira, H., Syuhei, M., Takumi, I., Tetsuyuki, T.: Ant Colony Optimization Using Exploratory Ants for Constructing Partial Solutions. In: IEEE Congress on Evolutionary Computation (CEC), pp. 1–7 (2010)
3. Ayan, A., Deepyaman, M., Aritra, B., Janarthanan, R., Amit, K.: Extension of Max-Min Ant System with Exponential Pheromone Deposition Rule. In: 16th IEEE International Conference on Advanced Computing and Communication, pp. 1–8 (2008)
4. Zhaojun, Z., Zuren, F.: A Novel Max-Min Ant System Algorithm for Traveling. In: IEEE International Conference on Intelligent Computing and Intelligent Systems, ICIS, pp. 508–511 (2009)
5. Thomas, S., Holger, H.H.: MAX–MIN Ant System. Future Generation Computer Systems 16, 889–914 (2000)
6. TSPLIB, http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html
7. Keld, H.: General k-opt submoves for the Lin–Kernighan TSP Heuristic. Mathematical Programming Computation 1, 119–163 (2009)
8. Randall, M., Montgomery, J.: Candidate Set Strategies for Ant Colony Optimisation. In: Dorigo, M., Di Caro, G.A., Sampels, M. (eds.) ANTS 2002. LNCS, vol. 2463, pp. 243–249. Springer, Heidelberg (2002)
9. Othman, Z.A., Rais, H.M., Hamdan, A.R.: Strategies DACS3 Increasing its Performances. European Journal of Scientific Research 27, 488–499 (2009)
10. Rais, H.M., Othman, Z.A., Hamdan, A.R.: Reducing Iteration Using Candidate List. In: International Symposium on Information Technology (ITSim 2008), vol. 3, pp. 1–8. IEEEXplore (2008)
11. Chaoxue, W., Duwu, C., Yikun, Z., Zhurong, W.: A Novel Ant Colony System Based on Delauney Triangulation and Self-adaptive Mutation for TSP. International Journal of Information Technology 12(3), 89–99 (2006)
12. ASOTSP software package, http://iridia.ulb.ac.be/dorigo/ACO/aco-code/public-software.html

Parallel Max-Min Ant System Using MapReduce

Qing Tan[1,2], Qing He[1], and Zhongzhi Shi[1]

[1] The Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences,
100190 Beijing, China
[2] Graduate University of Chinese Academy of Sciences,
100049 Beijing, China
{tanq,heq,shizz}@ics.ict.ac.cn

Abstract. Ant colony optimization algorithms have been successfully applied to solve many problems. However, in some large scale optimization problems involving large amounts of data, the optimization process may take hours or even days to get an excellent solution. Developing parallel optimization algorithms is a common way to tackle with this issue. In this paper, we present a MapReduce Max-Min Ant System (MRMMAS), a MMAS implementation based on the MapReduce parallel programming model. We describe MapReduce and show how MMAS can be naturally adapted and expressed in this model, without explicitly addressing any of the details of parallelization. We present benchmark travelling salesman problems for evaluating MRMMAS. The experimental results demonstrate that the proposed algorithm can scale well and outperform the traditional MMAS with similar running times.

Keywords: Ant colony optimization**,** MMAS, Parallel MMAS, Travelling salesman problem, MapReduce, Hadoop.

1 Introduction

Max-Min Ant System [1] is an optimization algorithm that was inspired by the behavior of real ants. This evolutionary algorithm has become popular and has been found to be effective for solving NP-hard combinatorial problems like travelling salesman problem (TSP). However, as the city number grows, the algorithm often takes a very long time to obtain the optimal solution. Efficient parallel Ant Colony Optimization (ACO) [2] algorithms and implementation techniques are the key to meet the scalability and performance requirements entailed in such cases. So far, there are several parallel implementations of ACO algorithm [3,4]. In the PACS [3] method, the artificial ants are firstly generated and separated into several groups, and ACS is then applied to each group and the communication between groups is applied according to some fixed cycles. [4] proposed two parallel strategies for the ant system: the synchronous parallel algorithm and the partially asynchronous parallel algorithm. And all of the above methods need the programmers to design and implement the detailed parallelization on different processors.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 182–189, 2012.

MapReduce [5] is a programming model and an associated implementation for parallel processing large dataset. Users only specify the computation in terms of a map and a reduce function, and the underlying runtime system automatically parallelizes the computation across the cluster of machines.

In this paper, we adapt MMAS algorithm in MapReduce framework and present a MRMMAS to make the method applicable to dealing with large scale problems. MRMMAS is simple, flexible, and scalable because it is designed in the MapReduce model. Considering TSP is the most typical application of MMAS, we present our MRMMAS method for solving TSP and conduct comprehensive experiments to evaluate its performance on some TSP benchmark problems.

The rest of the paper is organized as follows. In Section 2, we present preliminary knowledge including MapReduce overview and introduction of standard MMAS. Section 3 describes how MMAS can be cast in the MapReduce model and shows the map function and reduce function of MRMMAS in detail. Experimental results in Section 4 demonstrate that the proposed algorithm can scale well through the computer cluster. Finally, we offer our conclusions in Section 5.

2 Preliminary Knowledge

2.1 MapReduce Overview

MapReduce, as the framework showed in figure 1, is a simplified programming model which is well suited to parallel computation [6]. Under this model, programs are automatically distributed to a cluster of machines. In MapReduce, all data are organized in the form of keys with associated values. For example, in a program that counts the frequency of occurrences for different words, the key could be set as a word and its value would be the frequency of that word.

As its name shows, map and reduce are two basic stages in the model. In the first stage, the map function is called once for each input records. At each call, it may produce intermediate output records with the form of key-value pair. In the second stage, these intermediate outputs are grouped by key, and the reduce function is called once for each key. Finally, the reduce function will output some reduced results.

More specifically, the map function is defined as a function that takes a single key-value pair and outputs a list of new key-value pairs. For each call, it may produce any number of intermediate key-value pairs. It could be formalized as:

$$\text{Map: } (\text{Key}_1, \text{Value}_1) \rightarrow \text{list}((\text{Key}_2, \text{Value}_2))$$

In the second stage, these intermediate pairs are sorted and grouped by key, and the reduce function is called once for each key. The reduce function reads a key and a corresponding list of values and outputs a new list of values for the key. Mathematically, this would be written:

$$\text{Reduce: } (\text{Key}_2, \text{list}(\text{Value}_2)) \rightarrow \text{Value}_3$$

The MapReduce model provides sufficient high-level parallelization. Since the map function only takes a single record, all map operations are independent of each other and fully parallelizable. And also the reduce function can be executed in parallel on each set of intermediate pairs with the same key.

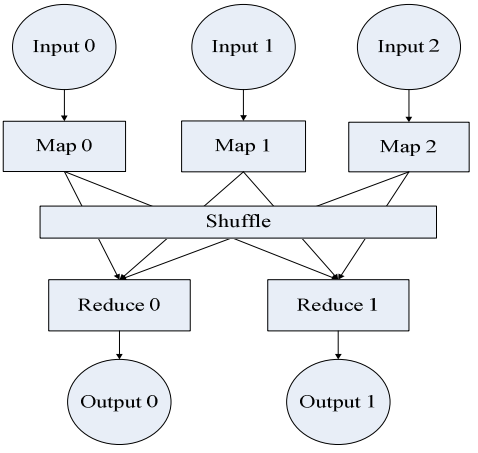


Fig. 1. Overview of the MapReduce execution framework

2.2 Max-Min Ant System (MMAS)

Ant Colony Optimization (ACO) metaheuristic is a population based approach inspired by the behavior of ant colony in real world. In ACO, solutions of the problem are constructed within a stochastic iterative process, by adding solution components to partial solutions. This process, together with the pheromone updating rule in ACO, makes the algorithm efficient in solving combinatorial optimization problems.

Initially, each ant was randomly positioned on a starting node. Then each ant applies a state transition rule to incrementally build a solution. Finally, all of the solutions are evaluated and the pheromone updating rule was applied until all the ants have built a complete solution. The framework of ACO algorithm could be represented as follows:

Procedure: ACO algorithm for static combinatorial problems

---

1. Initialize parameters and pheromone trails;
   Loop /* at this level each loop is called an iteration */
2. Put each ant in a random starting node;
   Loop
3. /*Construct solutions*/
   Each ant applies a *state transition rule* to choose a next city to visit;
   Until all ants have built a complete solution
4. *Pheromone updating rule* is applied;
   Until end condition is satisfied, usually reach a given iteration number

---

Max-Min Ant System [1] is one of the best implementation of ACO algorithm. It combines an improved exploitation of the best solutions with an effective mechanism for avoiding early search stagnation. It differs from Ant System (AS) mainly in the following three aspects: (1) Only one single ant adds pheromone after each iteration; (2) The range of possible pheromone trails on each solution component is limited to an interval $[\tau_{\min}, \tau_{\max}]$; (3) The initial pheromone trails are set to $\tau_{\max}$.

3 MapReduce Based Parallel Max-Min Ant System

In this section, we present the main design for MapReduce Max-Min Ant System (MRMMAS). Firstly, we point out how MMAS be naturally adapted to MapReduce programming model and present the general idea of MRMMAS. Then we explain how the computations can be formalized as map and reduce operations in detail.

3.1 The Analysis of MMAS from Serial to Parallel

The whole procedure of MMAS is an iteration process. In every round of the iteration, the ant colonies construct feasible solutions through two rules: state transition rule and pheromone updating rule. And in MMAS, the pheromone updating rule is applied only when all ants have built a complete solution. In another word, the pheromone level keeps constant during the process of solution construction.

In MMAS, the most intensive calculation to occur is the calculation of solution construction. In each iteration, every ant would require a lot of computations to decide which city to visit from its current city. Fortunately, the pheromone updating rule in MMAS does not require the communications among the ants in the same iteration but only deliver the information to the ants in the following iterations through the updating of the pheromone. It is obviously that the computation of constructing a solution for one ant is irrelevant with the construction of another ant in the same iteration. Therefore, the solution construction process could be parallel executed. After this phase, all the constructed solutions are summed up and pheromone updating rule is carried out. The updated pheromone level will be send to each ant and play a role in the following iteration.

3.2 MMAS Based on MapReduce

In an iteration of MMAS, each ant in the swarm locates at a starting node, chooses a next city to visit step by step, and evaluates its solution. All of these actions are completed independently of the rest of the swarm. As the analysis above, MRMMAS algorithm needs one kind of MapReduce job. The map function performs the procedure of constructing a solution for one ant and thus the map stage realizes the solution construction for all the ants in a parallel way. Then, the reduce function performs the procedure of updating the pheromone. For each round of the iteration, such a job is carried out to implement the whole process of MMAS. The procedure of MRMMAS is shown in the following.

Procedure: MapReduce MMAS for static combinatorial problems

---

1. Initialize parameters and pheromone trails;
   Loop /* for each iteration, a MapReduce job is carried out */
2. /* **Map stage** */
3. /* **a map function** realizes the behavior of an ant */
4. The ant is randomly put in a starting node;

5. The ant applies a *state transition rule* to choose a next city to visit until a complete solution has been built. /* solution construction */
6. Calculate the fitness of the solution. /* solution evaluation */
7. /* **Reduce stage** */
Pheromone updating rule is realized by **a reduce function**;
Until end condition is satisfied, usually reach a given iteration number

Map Function: Firstly, the pheromone values, heuristic information, and all of the parameters used in the state transition rule are transmitted into the map function from the main function of the MapReduce job. The MRMMAS map function, shown as function 1, is called once for each ant in the population. The input dataset is stored on HDFS as a sequence file of <key, value> pairs, each of which represents a record in the dataset. The number of the record is set as the number of the ant population. So the map function would be carried out m times, where m is the population of the ant swarm. The dataset is split and globally broadcasted to all mappers. Consequently, the process of solution construction for the ants is parallel executed. For each map task, one ant constructs one solution according to the state transition rule. Then, the solution is evaluated and expressed as an output <key, value> pair.

Function 1: MRMMAS Map

```
def mapper(key, value):
    /* get η[n][n], τ[n][n], α, β from MapReduce job */

    /* initialize tabuList */
    for i=1 to n do
        tabuList[i] = false;

    /* randomly put the ant in a starting node */
    currentPosition = randomInit(n);
    solution[1] = currentPosition;
    tabuList[currentPosition] = true;

    /* construct the solution through state transition rule */
    for i=2 to n do
        /* calculate the visited probability of each city */
        for (int j=0; j<city; j++) {
            if (list[j] == false)  { product[j] = τ^α_{currentPosition,j} * η^β_{currentPosition,j} ;}
            else { product[j] = 0;}
        }
        /* randomly select a city to visit according to the probabilities */
        currentPosition = randomSelect(product);
        solution[i] = currentPosition;
        tabuList[currentPosition] = true;

    /* solution evaluation */
    fitness = Fit(solution);
```

```
/* output the solution in a <key', value'> pair */
key'= "Solution";   /* a string "Solution" */
Take fitness+solution as vaule';
output <key',  value'> pair;
```

In the above procedure, the constructed solution and its fitness are outputted by a <*key'*, *value'*> pair. All of the mappers have the same *key*', so all of the solutions will be summed up together in the reduce step. And the information of different solution is expressed in different *vaule*'. Suppose a TSP solution is [1-2-3-4-5-6-7-1] and its path length is 123.45, then *vaule*' is a string "123.45+1,2,3,4,5,6,7".

Reduce Function: The input of the reduce function is the intermediate <key, value> pairs obtained from the map function of each host. As described in the map function, each pair includes a solution and its fitness. In the reduce function, we can sum up all the solutions constructed in the map step and obtain the best solution in the iteration and the best solution from the beginning. Then we can update the pheromone according to the pheromone updating rule in MMAS. These results are outputted by a <key, value> pair and will be transmitted to all the mappers in the following iteration. The pseudo code for MRMMAS reduce function is shown in function 2.

Function 2: MRMMAS Reduce

```
def  reducer(key, value_list):
    /* get  τ[n][n],  ρ  and global best solution gBest from MapReduce job */

    /* Of all of the solutions, find the best record in the current iteration */
    for value in value_list:
        fitness = getFitness(value);   solution = getSolution(value);
        if (iBest is null) or (fitness > iBest)
            iBest = fitness;   iBestSolution = solution;

    /* update the global best solution */
    if (iBest > gBest)   { gBest = iBest;}

    /* pheromone updating */
    τ = (1 − ρ) * τ ;
    for all edges(i,j) in iBestSolution
        τ_ij = τ_ij + Δτ_ij^best ;
    /* range pheromone into  [τ_min, τ_max]  */
    for all edges(i,j)
        if ( τ_ij > τ_max )   { τ_ij = τ_max ;}
        if ( τ_ij < τ_min )   { τ_ij = τ_min ;}

    /* output the results in a <key', value'> pair */
    Take gBest+gBestSolution as key';
    Take  τ[n][n]  as vaule';
    output <key',  value'> pair;
```

4 Experiments

In this section, we evaluate the performance of MRMMAS. Experiments were run on a cluster of computers, each of which has two 2.8 GHz cores and 4GB of memory. Some generally available and typical TSP data sets were used as the test material.

Considering that MRMMAS performs the same calculations as a serial implementation of MMAS, MRMMAS and serial MMAS will achieve the same level of accuracy with the same parameter setting. We have compared the quality of solutions and thus verified the correctness of MRMMAS. Thereby, the following experiments mainly focus on the efficiency of MRMMAS. We will check the average execution time per iteration because it shows whether the parallel implementation is an improvement. The first iteration of each run was excluded from averages because it often ran slightly faster or slower than the rest of the runs due to the initialization.

Figure 2 shows the average execution time of MRMMAS on the data set kroA100. The number we reported is averaged after five runs of MRMMAS, and each run has 50 iterations of MMAS. From the results, we can see that the running time could be effectively reduced as the number of processors grows.

We use speedup as a measure of scalability. Speedup is defined as the ratio of the serial runtime of the sequential time for solving a problem to the time taken by the parallel algorithm to solve the same problem on p processing elements [7]. Thus, the speedup with p processors is: $S_p = t_1 / t_p$. To measure the speedup, we increase the number of computers in the system. The perfect parallel algorithm will demonstrate linear speedup: a system with p times the number of computers yields a speedup of p. However, linear speedup is difficult to achieve because the communication cost among the cluster of computers.

Figure 3 shows the speedup performance of MRMMAS on different test set. From the results, we can see that MRMMAS scales well through 32 processors. However, the improvement becomes gradually undramatic as the number of processors grows. That is because the implementation and communication overhead hindered further improvement. Moreover, the speedup performance on large-scale TSP is better than those of smaller TSP due to the higher computation proportion.

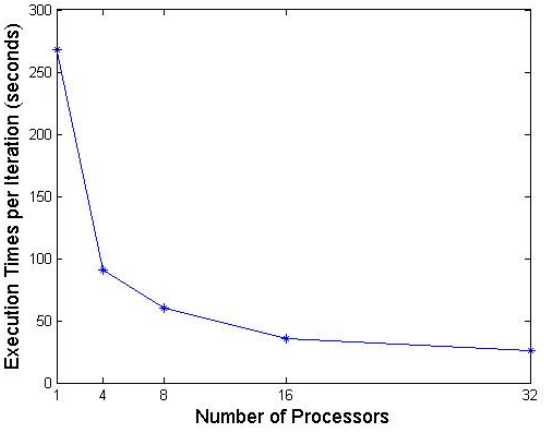


Fig. 2. Execution times per iteration for MRMMAS on kroA100

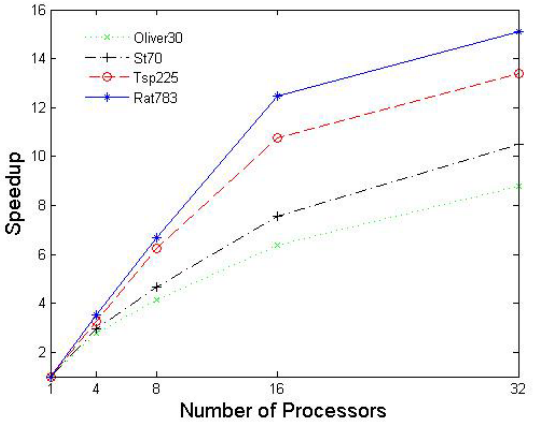


Fig. 3. Speedup for MRMMAS on different test data set

5 Conclusions

Although ACO algorithm has successfully been applied to solve many problems, its long running time is always an issue when dealing with large scale problems. This paper presents a parallel MMAS algorithm based on MapReduce, which will be widely embraced by both academia and industry. In our implementation, the process of solution construction will be carried on in different processors. The MapReduce system can balance the load dynamically and automatically. We have presented that MMAS can be naturally adapted to the MapReduce programming model and the experimental results show that it scales well through the computer cluster.

Acknowledgments. Supported by the National Natural Science Foundation of China (No. 60933004, 60975039, 61175052, 61035003, 61072085), National High-tech R&D Program of China (863 Program) (No.2012AA011003).

References

1. Stützle, T., Hoos, H.: MAX-MIN ant system. Future Generation Computer System 16(8), 889–914 (2000)
2. Dorigo, M., Stützle, T.: Ant Colony Optimization. The MIT Press, America (2004)
3. Chu, S.-C., Roddick, J., Pan, J.-S., Su, C.-J.: Parallel Ant Colony Systems. In: Zhong, N., Raś, Z.W., Tsumoto, S., Suzuki, E. (eds.) ISMIS 2003. LNCS (LNAI), vol. 2871, pp. 279–284. Springer, Heidelberg (2003)
4. Bernd, B., Gabriel, E.K., Christine, S.: Parallel Strategies for the Ant System. University of Vienna, Vienna (1997)
5. Jeffrey, D., Sanjay, G.: MapReduce: Simplified Data Processing on Large Clusters. Communications of the ACM 51, 107–113 (2008)
6. Ralf, L.: Google's MapReduce Programming Model – Revisited. Science of Computer Programming 70, 1–30 (2008)
7. Grama, A., Gupta, A., Karypis, G., Kumar, V.: Introduction to Parallel Computing, 2nd edn. Addison-Wesley, Harlow (2003)

Parallel Implementation of Ant-Based Clustering Algorithm Based on Hadoop*

Yan Yang[1,2], Xianhua Ni[1,2], Hongjun Wang[1,2], and Yiteng Zhao[1]

[1] School of Information Science and Technology, Southwest Jiaotong University, Chengdu, 610031, P.R. China
[2] Key Lab of Cloud Computing and Intelligent Technology, Sichuan Province, Chengdu, 610031, P.R. China
{yyang,wanghongjun}@swjtu.edu.cn,
{sampsonni.cn,yiyiageha4}@gmail.com

Abstract. Hadoop is a distributed system infrastructure of cloud computing. Based on the characteristics of ant-based clustering algorithm, the paper implements the parallelization of this algorithm using MapReduce on Hadoop. The Map function calculates the average similarity of the object with its neighborhood objects. The Reduce function processes the objects with the Map outputs and updates related information of both ants and the objects to get ready for the next job. Results on the Hadoop clusters show that our method can significantly improve the computational efficiency with the premise of maintaining clustering accuracy.

Keywords: Ant-based Clustering, Parallelization, Hadoop, MapReduce model.

1 Introduction

With the rapid development of information technology, vast amounts of data are created throughout the various fields and make the traditional data analysis techniques a challenging problem. Clustering as an important topic in data mining can find the internal relationships in data without supervision and mine the valuable information for further utilizing. However, the sharp growth of data brings the time and space of dual challenges to clustering and on the other hand, so the traditional clustering algorithms are serially calculated by uniprocessors, the conventional approaches can't cluster data successfully under the limited space and time. Parallel computing technology can eliminate the bottleneck of serial clustering task by dividing the data sets into several subsets and arranging each subset to each calculation node respectively; after the clustering processing by each local node, some merge strategies are used to get the final clustering results by collecting clustering results from local nodes. MPI (Message Passing Interface) and MapReduce are the most representative

* This work is partially supported by the National Science Foundation of China (Nos. 61170111, 61003142 and 61152001) and the Fundamental Research Funds for the Central Universities (No. SWJTU11ZT08).

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 190–197, 2012.

calculation model of parallel computing techniques. MPI is the early parallel computing model and the realization is very complicated. MapReduce proposed by Google is a concise and easy control distributed computing model which has been successfully applied to many computing tasks. Hadoop is one of the open source project of Apache and it provides a computing framework of MapReduce and a distributed file system HDFS which is similar to Google's GFS; and Hadoop has been successfully applied to many fields such as image mining, genome sequence, machine translation, grid data analysis, text mining, image analysis and astronomy etc.

In this paper the parallel ant-based clustering algorithm oriented to MapReduce in Hadoop platform is designed, then some datasets are chosen for experiments and the results show that the proposed approach improves the performance.

The rest of this paper is organized as follows. Section 2 states background. Section 3 introduces the parallel strategy based on MapReduce. Section 4 demonstrates the experimental results. Conclusion is drawn in section 5.

2 Background

2.1 MapReduce

MapReduce is a programming model and software framework for developing applications that rapidly process massive datasets in parallel on large clusters of compute nodes. A MapReduce job is composed of Map function and Reduce function, Map automatically partitions the input dataset into a set of splits or shards and takes an input pair to the Map function and finally produces a set of intermediate key-value pairs. Reduce partitions the intermediate key space into pieces using a partitioning function and finally output the summary results. Fig. 1 shows the overall flow of the MapReduce operation.

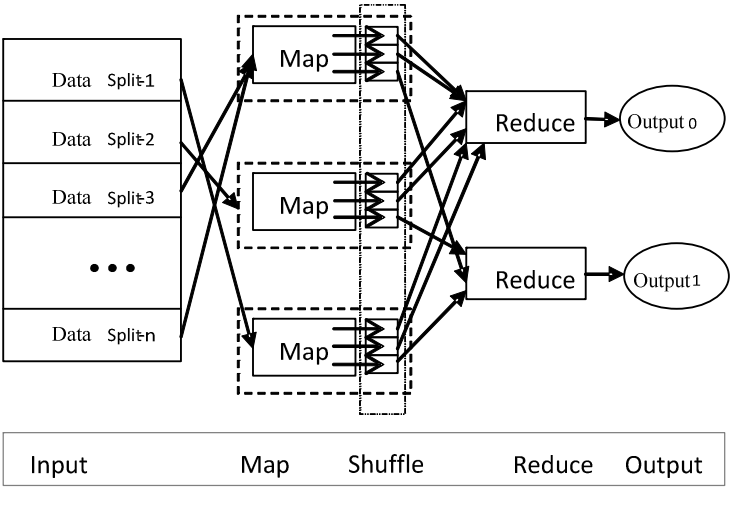


Fig. 1. Flow of MapReduce Operation

A Map task firstly reads the contents of the corresponding data split and parses key-value pairs out of the input data and finally passes each pair to the user-defined Map function. The intermediate key-value pairs produced by the Map function are the input of Reduce function which iterates over the sorted intermediate data and for each unique intermediate key encountered, it passes the key and the corresponding set of intermediate values to the Reduce function. The output of the Reduce function is appended to a final output file for this reduce partition.

2.2 Ant-Based Clustering Algorithm

The self-organization, distribution, communication with pheromone and cooperation between ants are some important features of swarm intelligence. Ant colony algorithm can solve many complex problems with simulating this intelligent behavior of ants. It is mainly used in communication networks, combinatorial optimization and robotics, etc. The Ant Colony Optimization algorithm and Ant Routing algorithm are two successful algorithms in swarm intelligence area. Researchers design the ant-based clustering algorithms by simulating the behavior of ants carrying body and foraging. In this paper we use the algorithm proposed by Yang and Kamel in [7] which can get better results by improving the basic ant-based clustering algorithm through the following method:

1) Using both Euclidean distance and Cosine distance to measure the distance because they can compensate each other. The average similarity of object o_i with its neighborhood is given by

$$f(o_i) = \max\{0, \frac{1}{s^2} \sum_{o_j \in Neigh_{s\times s}(r)} [1 - \frac{d_{ij}}{\alpha(1+(v-1)/v_{\max})}]\} \quad (1)$$

where d_{ij} denotes the new measurement of distance between o_i and o_j, and the distance is defined by

$$d_{ij} = \varepsilon d_{euc}(o_i, o_j) + (1-\varepsilon) d_{sim}(o_i, o_j), \quad 0 < \varepsilon < 1 \quad (2)$$

where $d_{euc}(o_i, o_j)$ denotes the Euclidean distance, and $d_{sim}(o_i, o_j)$ denotes the Cosine distance.

2) Used the sigmoid function as the probability conversion function since it is nonlinear, it can help to solve linearly inseparable problems. The dropping probability for a randomly moving loaded ant to deposit an object is given by

$$P_d = Sigmoid(f(o_i)) \quad (3)$$

and the picking-up probability for a randomly moving ant that is currently not carrying an object to pick up an object is defined by

$$P_p = 1 - Sigmoid(f(o_i)) \quad (4)$$

where

$$Sigmoid(x) = \frac{1-e^{-cx}}{1+e^{-cx}}. \tag{5}$$

3) Increasing the value of $\boldsymbol{c}$ in Eq. (5) to solve the problem that the outliers are hard to be dropped by the ants. By this way the convergence of algorithm is sped up.

Finally the author [7] used the hypergraph method based on the improved ant-based algorithm with different speed of ants to ensemble clustering results.

3 Parallel Strategy Based on MapReduce

Given a dataset $\boldsymbol{O} = \{o_1, \ldots, o_n\}$, $\boldsymbol{O}$ is randomly projected onto a plane without replacement, and each object in $\boldsymbol{O}$ is marked by a two-dimensional coordinate $\boldsymbol{Crd(o_i)}$. For each ant in ant colony $\boldsymbol{A}$, we assign the location $\boldsymbol{Crd(a_i)}$ to be (0,0), and the loading state $\boldsymbol{Sta(a_i)}$ to be false. Initialize the number of ants: $\boldsymbol{n\_ant}$, maximum number of iteration: $\boldsymbol{L}$, side length of local region: $\boldsymbol{s}$, and other parameter: ε, $\boldsymbol{c}$, etc. Fig. 2 shows the parallel implementation procedure of Ant-based Clustering algorithm.

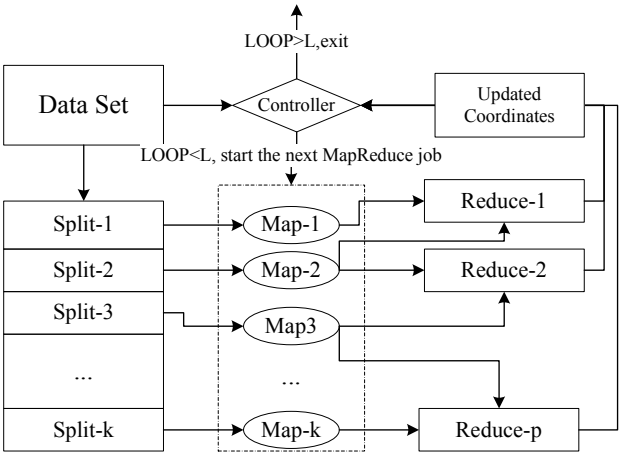


Fig. 2. Parallel implementation procedure of Ant-based Clustering algorithm based on MapReduce

Controller is a global phase to initialize parameters and manage the projection task. Before each loop of MapReduce job, **Controller** updates the location $\boldsymbol{Crd(a_i)}$ to make $\boldsymbol{a_i}$ move to another position (if loaded: moving to a free site; if unloaded: moving to another site occupied by any object without ant). After $\boldsymbol{L}$ times of MapReduce job the **Controller** can access the updated coordinates of ants. If an object is isolated, or the number of its neighbor is less than a given constant, then label it as an outlier, else give this object a cluster sequent number and recursively

label the same sequent number to those objects who is the neighbors of this object within local region.

At the Map phase processor gets each line data from the split text as the input of Map function. With the identifier and coordinate of each ant and object, processor computes the similarity of an object within a local region s by formula (**1**). The $< ID, f_l(o_i) >$ key-value pair is as the output of map procedure, ID is the identify number of o_i and $f_l(o_i)$ is the similarity within its local region.

At the Reduce phase processor computes the global average similarity from outputs of map phase. It is computed as:

$$f^{'}(o_i) = \frac{1}{k}\sum_{l=1}^{k} f_l(o_i). \tag{6}$$

Then ants pick up or drop the object with the probability calculated from Eq. (**3**) and (**4**), the processor updates the location $Crd(o_i)$ and the state $Sta(a_i)$ of ant o_i. After completion of Reduce tasks, global controller turns into the next round of MapReduce job.

4 Experiment Results and Analysis

Experimental environment consists of one server(DELL PowerEdge R710, CPU: Xeon X5560 2.8GHz, RAM:8GB, NetCard:1000Mbps) and seven PCs(DELL V260D-566, CPU: Intel Pentium DualCore 2.7GHz, RAM:2GB, NetCard: 1000Mbps), the server is the master to be designated as NameNode and JobTracker, and the PCs is the slaves and datanodes. The ant-based clustering algorithm is tested with both traditional method and MapReduce programming model. With the datasets of different domains the effectiveness of the algorithm is evaluated according to the experimental results. In our experiments we firstly used three datasets Iris (150 instances, 3 classes, 4 attributes), Ecoli (336 instances, 8 classes, 8 attributes) and Letter (20,000 instances, 26 classes, 17 attributes) from UCI to test the strategy, and then we apply the Reuters-21578 collection to our experiments which is a standard corpus with 21,578 news. We only sampled 5 different document collections each consisting of 1000 documents. After the preprocessing and normalization of all datasets the **Controller** initializes some parameters that ε=0.2, c=0.95, L=1000. And for the important parameter s, we give a simple test on each data set with the step 2 in the area [10,50] and apply the value obtaining the best clustering result to s. According to [7] the parameter α from Eq. (**1**) is to adjust the dissimilarity between objects, it can determine the number of clustering and speed of convergence. And the bigger α makes less clusterings and faster algorithm converges, the smaller α, otherwise. The same as [7] our experiment uses a coefficient β=0.05 to replace for parameters α, v and s.

4.1 Evaluation

To compare the quality of clustering an external evaluation criterion called *F-measure* is used here. The F-measure of cluster *j* and class *i* is defined by

$$F(i,j) = \frac{2 \times Precision(i,j) \times Recall(i,j)}{Precision(i,j) + Recall(i,j)} \tag{7}$$

where

$$Precision(i, j) = \frac{N_{ij}}{N_i} \tag{8}$$

and

$$Recall(i, j) = \frac{N_{ij}}{N_j} \tag{9}$$

where N_{ij} is the number of class i in cluster j, N_j is the number of members of cluster j and N_i is the number of members of class i.

4.2 Experiment Results

We compare the average F-measure of traditional method with MapReduce model on the dataset, and Fig. 3 shows the details of F-measure in our experiment with different number of ants.

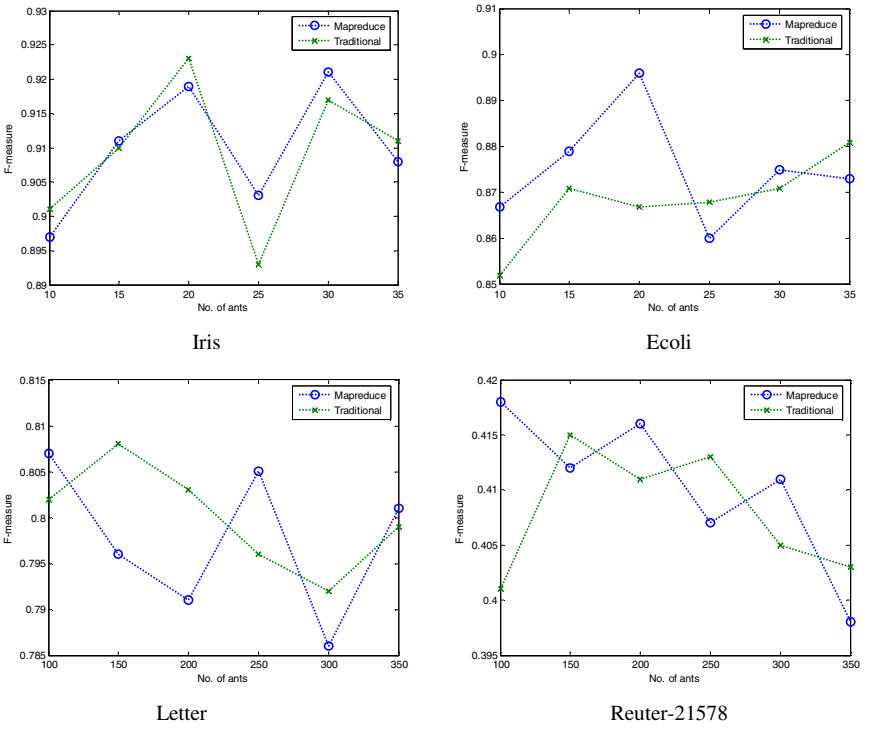


Fig. 3. The value of F-measure from traditional method and MapReduce model

As shown in Fig. 3 there is no obvious advantage between the traditional method and MapReduce model on the datasets, it is a crossing in the shape of wave form. Taken as a whole, clustering precision cannot be significantly improved.

Hadoop can improve the global algorithm efficiency through multiple slaves. In our experiment we also run the parallel Ant-based Clustering algorithm with different number of slaves, and we change the number of slaves from 1 to 8 with step 1. The experimental result is shown in Figure 4, it clearly shows that algorithm gets a better performence when 4 slaves are used, and the execution time is decreasing when more slaves from 1 to 4 are used.

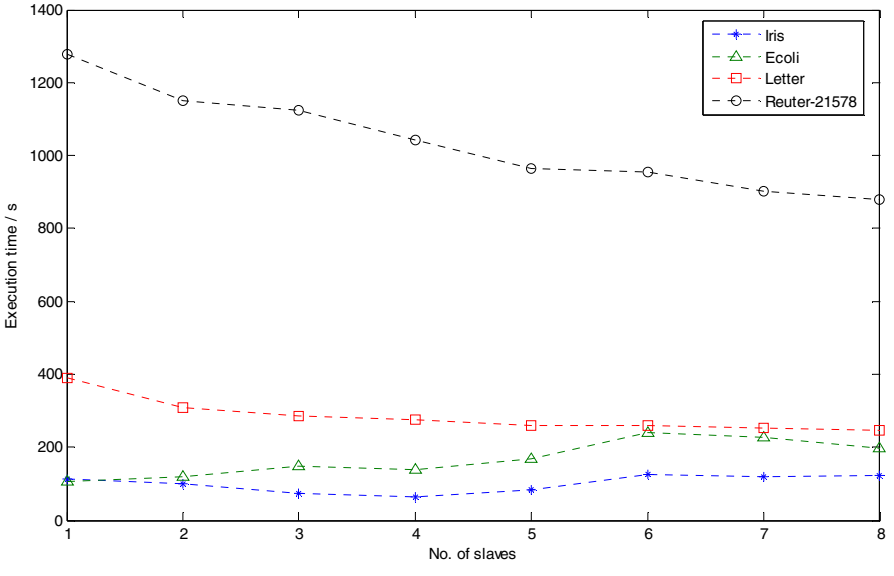


Fig. 4. Speedup of Ant-based Clustering algorithm with different slaves on Hadoop

Moreover, with the comparison of experimental results from Fig. 4, it shows that the Hadoop cannot reduce the execution time on the small datasets Iris and Ecoli when the number of slaves is more than 4. On the other hand Hadoop can get less execution time with the larger dataset Letter and Reuter-21578 when the slaves increase. It also notes that the number of slaves is not proportional to the computational efficiency, and there will be different optimal number of slaves for various datasets. The communication procedure will take the most of the execution time on the small datasets, however the larger datasets can work well because the network communication time can be neglected.

5 Conclusion

In this paper we briefly introduce the principal of MapReduce and Ant-based Clustering algorithm, the parallel method for Ant-based Clustering algorithm is

proposed on the Hadoop platform. The experimental results show that the algorithm can save the global execution time of algorithm with no obvious fluctuations of accuracy when using the proposed method. In addition it is shown that the larger dataset is more suitable for running on Hadoop because the network communication time will take a small proportion of the global execution time.

References

1. Dean, J., Ghemawat, S.: MapReduce: Simplified data processing on large clusters. In: Operating Systems Design and Implementation, pp. 137–149 (2004)
2. Apache Hadoop. Hadoop, http://hadoop.apache.org
3. Ghemawat, S., Gobioff, H., Leung, S.: The Google file system. In: Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles, pp. 29–43. ACM, Bolton Landing (2003)
4. Borthakur, D.: The Hadoop Distributed File System: Architecture and Design. The Apache Software Foundation, http://hadoop.apache.org
5. Wei, J., Ravi, V.T., Agrawal, G.: Comparing map-reduce and FREERIDE for data-intensive applications. In: IEEE International Conference on Cluster Computing and Workshops. CLUSTER 2009, pp. 1–10 (2009)
6. Smith, A.E.: Swarm intelligence: from natural to artificial systems. IEEE Transactions on Evolutionary Computation 4, 192–193 (2000)
7. Yang, Y., Kamel, M.: Clustering Ensemble Using Swarm Intelligence. In: IEEE Swarm Intelligence Symposium, pp. 65–71 (2003)

Ant Colony Algorithm for Surgery Scheduling Problem

Jiao Yin and Wei Xiang

Faculty of Mechanical Engineering and Mechanics, Ningbo University, Ningbo, P.R. China
501225920@qq.com, xiangwei@nbu.edu.cn

Abstract. Considering the complete process of surgery including the preoperative and postoperative stages, multiple resource constraints involved and the integration of surgical upstream and downstream resources, surgery scheduling was described as an extended multi-resource constrained flexible job-shop scheduling problem and an optimization approach was proposed based on an improved ant colony algorithm. A resource selection rule and strategy of overtime judging and adjusting was designed, and the scheduling process with the ant colony algorithm was realized. The case study shows that the improved ant colony algorithm proposed in this paper achieved good results in shortening total time and allocating resources for surgery scheduling.

Keywords: surgery scheduling, ant colony algorithm, multi-resource constrained job-shop, flexible job-shop.

1 Introduction

As the cost and income center of a hospital, the operating room has a significant impact on the profitability of the whole hospital [1]. Hence, an efficient surgery scheduling system is in demand so as to maximize the operating room efficiency, increase the number of operations performed daily, and achieve appropriate resource allocation. From the review of current literature, some researchers evaluated the surgery scheduling by employing a simulation [2]. Most studies describe surgery scheduling as a combinatorial optimization problem, and the commonly used technique is mathematic programming such as goal programming [3], Branch-and–price [4], and column generation [5]. Since surgery scheduling is regarded as an NP-hard problem, it can be modeled by the classical shop scheduling models, such as the Job-shop [6] and flow-shop scheduling problems [7, 8]. However, the limitation of that research lies in the fact that either the multi-resource constraints are not considered or are limited among a few fixed modes. Recently, meta-heuristic algorithms have been recognized as useful methods for solving scheduling problems. Among the various meta-heuristic algorithms used in surgery scheduling, Genetic algorithm (GA) is used most, with the simulated annealing algorithm (SA) [9] and Tabu search algorithm (TA) [10] also frequently employed. They perform well in solving the deterministic NP-hard problems. Due to the kinds of uncertainties endemic in operating room management such as the arrival of an emergency and temporary cancellation, or postponement etc., surgery scheduling should be regarded as a dynamic scheduling problem. The ant

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 198–205, 2012.

colony algorithm, successfully applied to the traveling salesman problem (TSP) by Dorigo et al, is distinguished by positive feedback and self-organizing, making it more capable to apply in complex scheduling problems with uncertainties. However, so far no researcher has applied the Ant colony algorithm to attempt to solve the surgery scheduling problem.

In this paper, surgery scheduling is represented as an extended multi-resource, constrained flexible job-shop scheduling problem and is solved by an improved Ant colony algorithm. The remainder of the paper is organized as following: Section 2 presents the surgery scheduling problem description; Section 3 proposes the improved Ant colony algorithm for surgery scheduling; Section 4 is the case study which compares our algorithm with other surgery scheduling research, and the final section is the Conclusion.

2 Surgery Scheduling Problem

Surgery scheduling is described as an extended multi-resource constrained flexible job-shop scheduling problem. We assume that there are M resources (including operating rooms, nurses, etc.), S surgeons, and N surgeries to be performed in an operating system. Every surgery has three stages (the pre-operative stage, the peri-operative stage, and the post-operative stage) and their sequence has been determined in advance Different types of surgeries have different resource constraints. Whether each stage can start successfully is restricted not only by the variety of resources, but also by the performance of each previous stage. Each surgery can be performed by several surgeons, and its operating time varies by surgeon. The scheduling goal is to select the best resources and surgeons, determine the various surgeries' operating sequences, and get the shortest makespan, considering the variation of types and quantity of resources and the mutual constraint of their available time. Several assumptions are listed as following:

1. One resource (including surgeons) can only be allocated in one surgery at a time.
2. The operating sequence of three stages must be followed completely and in subsequent order.
3. All the surgeries are ready to be performed at a moment's notice.
4. A stage cannot be interrupted or stopped, once it has started.
5. The operating time of a surgery performed by a specific surgeon has been determined before the scheduling.
6. All surgeries have the same priority.
7. There is no order constraint between stages of different surgeries.
8. The transferring time between stages in the same surgery can be neglected.

3 Ant Colony Algorithm for Surgery Scheduling

An improved ant colony algorithm is proposed to solve the extended multi-resource constrained flexible job-shop scheduling problem.

3.1 Algorithm Description

Surgery scheduling can be signified by a disjunctive graph including nodes and arcs to make the problem more operable with ant colony algorithm. Each node represents a stage in a surgery and a surgery always has three nodes. Directed arcs are used to indicate an operating order that stages in one surgery must be in accordance with. Ants move and search in the graph, through traversing all nodes to generate the full problem solution. The surgery scheduling problem thus can be transformed into the problem of searching best path in the graph.

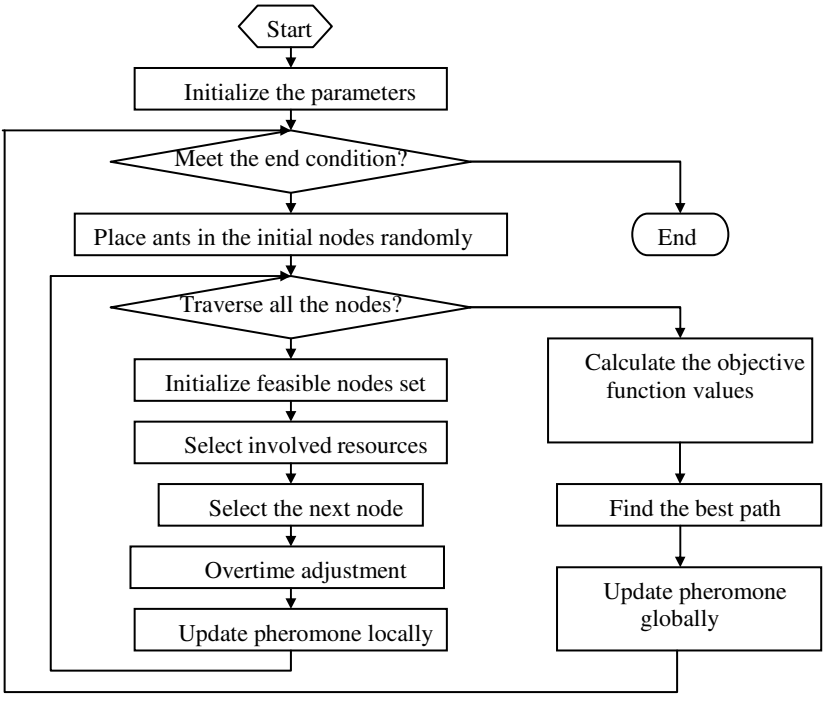


Fig. 1. Algorithm flow chart

The flowchart of the proposed ant colony algorithm for surgery scheduling is shown in Fig.1. Compared to the general ant colony algorithm, several additional processes like resource selection and overtime adjustment are further proposed to realize the multi-resource constrained flexible job-shop scheduling problem.

3.2 Resource Selection Rule

Different types of surgeries may have different resource requirements, and the operating time of surgeries vary with each surgeons' experience and skill. Each stage of a surgery involves multiple resources, so the starting time of a stage is not only affected

by the performance of the former stage but also influenced by the required resource constraints at this stage. Thus, an optimal resource allocation can improve the quality of the whole scheduling and minimize the makespan.

Heuristic rules have the advantage of low computation complexity and are selected to allocate resources. Our resource selection rule is called "ASAP" (as soon as possible), i.e., choosing those resources that allow the stage to start as soon as possible. The starting time of this stage is determined after finishing its resource allocation. EX_{ij} is the finishing time of the predecessor stage of stage j of surgery i, thus if this stage is the first stage of the surgery, EX_{ij}=0; R_{ijmk} denotes the earliest releasing time of resource k of kind m needed by stage j of surgery i at the virtual scheduling moment t. If at the moment the resource is not to be utilized, R_{ijmk}=0. ST_{ij} is the starting time of stage j of surgery i.

$$ST_{ij} = \max\{EX_{ij}, \max\{\min\{R_{ijmk}\}, m \in L_{(Q_{ij})}\}\}. \tag{1}$$

3.3 The State Transition Rule

Ants build solutions using a probabilistic transition rule. In the searching process, ants calculate the state transition probability according to the amount of pheromone and heuristic desirability. The probability that the ant will choose node j from node i is denoted as $P_{ij}(t)$:

$$P_{ij}(t) = \begin{cases} \dfrac{\tau_{ij}^{\alpha}(t) * \eta_{ij}^{\beta}(t)}{\sum_{s \subset S_0} \tau_{is}^{\alpha}(t) * \eta_{is}^{\beta}(t)}, & if\ j \in S_0 \\ 0, & otherwise \end{cases}. \tag{2}$$

where S_0 is the feasible nodes set, including nodes to be scheduled whose pioneers have been completed; $\tau_{ij}(t)$ is the pheromone intensity between node i and j; $\eta_{ij}(t)$ is a problem-specific function, representing the visibility from node i to node j; α and β are two parameters which determine the relative influence of the pheromone trail and the heuristic information.

To minimize the makespan, we construct the visibility function (i.e. $\eta_{ij}(t)$) based on the earliest starting time of the nodes. ST_j is the earliest starting time of node j:

$$\eta_{ij}(t) = 1 \Big/ ST_j + 1, \quad j \in S_0. \tag{3}$$

The next node to be chosen is based on the probability $P_{ij}(t)$ with a roulette wheel mechanism used to make the selection.

3.4 Overtime Adjustment

Generally speaking, hospitals institute an eight-hour working day system. To make the scheduling results closer to actual situation, we should judge and adjust the starting

time of operation to guarantee that surgeries can be finished within the working time as much as possible. The working time interval of a particular day k is defined as $[T_{Sk},T_{Ek}]$, then the next working time interval is $[T_{S(k+1)},T_{E(k+1)}]$.

If the earliest starting time of a surgery is out of the working time interval, to avoid working overtime, we will adjust the starting time to the next neighboring working time interval, i.e.:

$$\text{If } ST_{i1} + \max\{T_{i2(L_{ij})} + \cdots + T_{ij(L_{ij})}\} \notin [T_{Sk}, T_{Ek}], \text{ then } ST_{i1} = T_{S(k+1)} \quad . \tag{4}$$

Notably, we restrict the peri-operative stage in every surgery performed within the working time interval to reduce overtime costs and to help manage the workload of medical workers, i.e.:

$$\text{If } ST_{i2} + T_{i2(L_{ij})} \notin [T_{Sk}, T_{Ek}], \text{ then } ST_{i2} = T_{S(k+1)} \quad . \tag{5}$$

3.5 The Pheromone Update Rule

A pheromone local update rule and a pheromone global update rule are used in this improved Ant colony algorithm to deal with the residue pheromone.

In the searching process of each ant, the ant updates the pheromone after visiting each node:

$$\tau_{ij}(t+n) = (1-\rho) * \tau_{ij}(t) + \rho * \tau_0 \quad . \tag{6}$$

Where ρ denotes the pheromone evaporation rate; τ_0 is the initial pheromone value. By local pheromone updating, the possibility of ants crawling through the same path decreases, thus it can effectively avoid the algorithm falling into stagnation.

After all the ants finish traversing the nodes, the ant with the best schedule in the iteration updates the trails as follows,

$$\tau_{ij}(t+n) = (1-\rho) * \tau_{ij}(t) + \Delta\tau_{ij}(t) \quad . \tag{7}$$

$$\Delta\tau_{ij}(t) = \begin{cases} \dfrac{Q}{L_k}, & \textit{if ant k goes through(i,j) in this iteration} \\ 0, & \textit{otherwise} \end{cases} \quad . \tag{8}$$

Where $\Delta\tau_{ij}(t)$ is the incremental pheromone on the edge (i,j); Q is a adjustable parameter; L_k is the makespan of a solution.

4 Case Study

The ant colony algorithm in this paper is coded in MATLAB in the running environment of MATLAB 7.11.0 and Windows 7.The program is run on a PC with an Intel Core i3 CPU.

The instance from Pham of [6] is used to compare and prove the performance of the algorithm. The test instance contains 10 surgeries. One nurse is needed at the preoperative stage of each surgery, and resources utilized during the perioperative stage include operating room (OR), surgeon, nurse and anesthetist. A PACU bed or ICU bed is needed at the postoperative stage. In all, there are two operating rooms, three nurses, three surgeons, two anesthetists, one PACU and one ICU.

In the article of Pham et al, Every OR is staffed with one nurse and one anesthetist, and a preoperative nurse works only for the preoperative preparation stage. They allocate resources in a fixed resource mode, and formulate the problem as a mixed integer linear programming model, the scheduling result as shown in Fig.2. The Gantt chart actually includes the scheduling arrangement of 9 surgeries. Data in the literature [6] show that 10 surgeries were not arranged wholly within the working time of two days. Surgery 4 was delayed to the third working day and finished 3 hours after the third working day started.

Ensuring resource demand of stages and the number of all the resources in correspondence with literature [6], we release the fixed resource modes and make every resource available to combine with any other resources at the right time. Meanwhile in view of integration of resources, nurses can be allocated free. With the parameters setting shown in Table 1, the improved ant colony algorithm proposed above is used to solve this instance and the corresponding Gantt is shown in Fig.3.

As shown in the Gantt chart of our scheduling result, 10 surgeries were arranged reasonably within two working days and there was no delay. When all the 10 surgeries were finished, there was 2.5h left until the second working day end.

To further verify our algorithm's effectiveness, we experiment with an actual instance from the affiliated hospital of Ningbo University. The operating theater comprises ten operating rooms and twenty surgeons and there are severally five beds in ICU and PACU. Thirty-eight surgeries are scheduled in one day. The experimental results are showed in Table 2, compared with actual scheduling. Standard deviation (STDEV) is used to measure the balance of resources' utilization.

Fig. 2. Gantt chart of Pham's scheduling result

Table 1. Parameters setting of ant colony algorithm

Max Iterations	Quantity of ants	Evaporation rate	Q	α	β
100	30	0.5	100	1	1

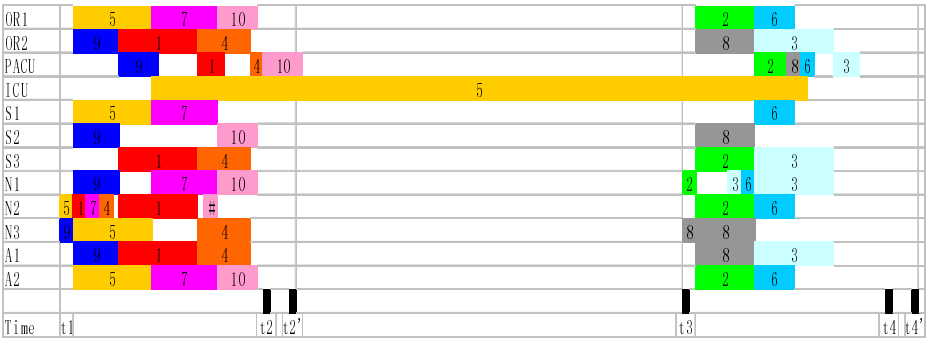


Fig. 3. Gantt chart of our scheduling result

The majorities of departments in hospital have more than one surgeon available in a day, so surgeries should be equally allocated to surgeons in a department as far as possible. The value of STDEV of surgeons' working time in a department can reflect the balance of surgeons' utilization in a scheduling. Fig.4 compares the condition of each surgeon's utilization in actual scheduling and our scheduling.

From the Table 2 and Fig.4, we see that both the makespan index and the balancing utilization index of our scheduling are better than those of actual scheduling. The experiment results show that the improved ant colony algorithm proposed in this paper has good performance on shortening total time and allocating resources for surgery scheduling.

Table 2. Computational results compared with actual scheduling

	Makespan (min)	Overtime (min)	STEDV of OR's working time	STEDV of nurse's working time
Actual scheduling	561	81	73.4	87.3
Our scheduling	473	0	57.3	69.4

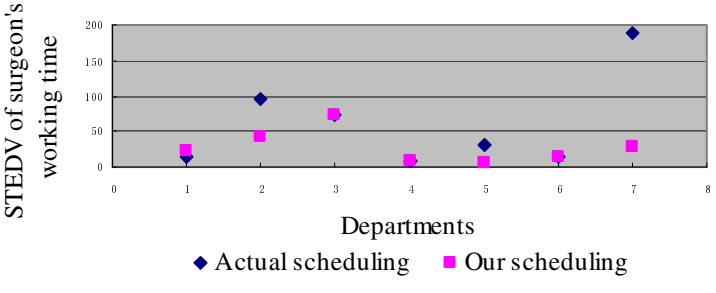


Fig. 4. STDEV of surgeons' working time

5 Conclusion

In this paper, we present an improved ant colony algorithm to solve surgery scheduling problem. In consideration of a complete process of surgery including preoperative and postoperative stages, multiple resource constraints involved and the integration of surgical upstream and downstream resources, surgery scheduling is treated as an extended multi-resource constrained flexible job-shop scheduling problem. To get an optimal resource allocation and minimize the makespan, we apply a resource selection rule called "ASAP" (as soon as possible), and to make the scheduling results closer to reality, a strategy of overtime judging and adjusting is added to our algorithm. Experimental results on both the test instance from literature [6] and the actual instance from a hospital show that the algorithm can solve the surgery scheduling problem effectively. It performs well in shortening total time and allocating resources for surgery scheduling.

References

1. Zskienock, G.B., Zambricki, C.S.: The Health Care Crisis: Impact on Surgery in the Community Hospital Setting. Arch. Surg. 136(5), 585–591 (2001)
2. Sciomachen, A., Tanfani, E., Testi, A.: Simulation models for optimal schedules of operating theatres. International Journal of Simulation 6(12-13), 26–34 (2005)
3. Ogulata, S.N., Erol, R.: A hierarchical multiple criteria mathematical programming approach for scheduling general surgery operations in large hospitals. Journal of Medical Systems 27(3), 259–270 (2003)
4. Fei, H., Chu, C., Meskens, N., Artiba, A.: Solving surgical cases assignment problem by a branch-and-price approach. International Journal of Production Economics 112, 96–108 (2008)
5. van Oostrum, J.M., Van Houdenhoven, M., Hurink, J.L., Hans, E.W., Wullink, G., Kazemier, G.: A master surgery scheduling approach for cyclic scheduling in operating room departments. OR Spectrum 30(2), 355–374 (2008)
6. Pham, D.-N., Klinkert, A.: Surgical case scheduling as a generalized job shop scheduling problem. European Journal of Operational Research 185, 1011–1025 (2008)
7. Fei, H., Meskens, N., Chu, C.: A planning and scheduling problem for an operating theatre using an open scheduling strategy. Computers & Industrial Engineering 58, 221–230 (2010)
8. Augusto, V., Xie, X.L., Perdomo, V.: Operating theatre scheduling with patient recovery in both operating rooms and recovery beds. Computers & Industrial Engineering 58, 231–238 (2010)
9. Beliën, J., Demeulemeester, E., Cardoen, B.: A decision support system for cyclic master surgery scheduling with multiple objectives. Journal of Scheduling 12(2), 147–161 (2009)
10. Hsu, V.N., de Matta, R., Lee, C.-Y.: Scheduling patients in an ambulatory surgical center. Naval Research Logistics 50, 218–238 (2003)

A Method for Avoiding the Feedback Searching Bias in Ant Colony Optimization

Bolun Chen[1] and Ling Chen[1,2,*]

[1] Department of Computer Science, Yangzhou University, Yangzhou, 225127, China
[2] State Key Lab of Novel Software Tech, Nanjing University, Nanjing, 210093, China
chenbolun1986@163.com, yzulchen@gmail.com

Abstract. One of the obstacles in applying ant colony optimization (ACO) to the combinatorial optimization is that the search process is sometimes biased by algorithm features such as the pheromone model and the solution construction process. Due to such searching bias, ant colony optimization cannot converge to the optimal solution for some problems. In this paper, we define a new type of searching bias in ACO named feedback bias taking the *k*-cardinality tree problem as the test instance. We also present a method for avoiding the feedback searching bias. Convergence analysis of our method is also given. Experimental results confirm the correctness of our analysis and show that our method can effectively avoid the searching bias and can ensure the convergence for the problem.

Keywords: ant colony optimization, deceptive problems, K-cardinality tree problem, solution convergence.

1 Introduction

Ant colony optimization (ACO) [1-4] is a popular method for hard discrete optimization problems. Due to its strong ability of optimization, ACO has been applied to solve problems in many areas. Numerous publications in a large variety of fields, particularly the combinational optimization problems, demonstrate the broad applicability and the excellent performance of ACO.

One of the obstacles in applying ACO to the combinatorial optimization is that the search process is sometimes biased by algorithm features such as the pheromone model and the solution construction process. The performance of ACO algorithms may decrease over time, depending on the pheromone model and the problem instance tackled. This behavior caused by the bias is clearly undesirable, because in general it worsens the probability of finding better solutions over time. Blum and Sampels [5,6] studied the application of ACO algorithms to shop scheduling problems. They discovered the bias in the search process. In a similar line of work, Merkle and Middendorf [7,8] studied the bias of a simple ACO by analyzing the dynamics of its model when applied to permutation problems. They discovered that in

* Corresponding author.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 206–216, 2012.

ACO applied to the permutation problem, the latter decisions of the construction process are entirely biased by the earlier one. Montgomery et al.[9,10] studied the searching bias of ACO on the assignment problem, and attributed search bias to different algorithmic components. They defined two types of searching bias in ACO, namely representational bias and construction bias. Due to such searching bias, the search process of ACO is sometimes misled. Ant colony optimization cannot converge to the optimal solution for some problems which are called deceptive problems. To achieve high optimization performance of ACO on such problems, it is important to find an effective method to avoid such searching bias in the process of optimization.

In this paper, we define a new type of searching bias in ACO named feedback bias. We prove the existence and influence of feedback bias in ACO taking the *k*-cardinality tree problem as the test instance. We also present a method for avoiding the feedback searching bias. Convergence analysis of our method is also given. Our experimental results confirm the correctness of our analysis and show that our method can effectively avoid the searching bias and can ensure the convergence the problem.

2 Feedback Bias in ACO and the *k*-Cardinality Tree Problem

ACO is designed for solving constrained optimization problems. In the search space S of an optimization problem, let X_i, $i = 1,\ldots,n$ be the n decision variables, where X_i can take values from the set $D_i = \{C_i^1, C_i^1, ..., C_i^{|D_i|}\}$. A variable assignment is written as $X_i = C_i^j \quad (i = 1,2,...,n)$. A complete assignment to all X_i gives a solution instantiation. The set of all such complete assignments is denoted as S. We also denote the set of all solution components as $R = \{C_i^j \,|\, i = 1,...n,\ j = 1,...,|D_i|\}$. A feasible solution s^* is a global optimum if its fitness $f(s^*) \geq f(s)$ for all $s \in S$.

Definition 1. For a solution component C_i^j, we use G_i^j to denote the set of solutions whose j-th component is C_i^j, namely, $G_i^j = \{(s_1, s_2, ..., s_n) \in S \mid s_i = c_i^j\}$. We define the expected fitness of solution component C_i^j as the summation of the fitness of all the solutions in G_i^j : $F(c_i^j) = \sum_{s \in G_i^j} f(s)$.

Definition 2. Given a constrained optimization problem P, let its optimal solution be $x^* = (x_1^*, x_2^*, ..., x_n^*)$. Suppose solution component $C_i^{k_i}$ has the highest fitness in $c_i^1, c_i^2, ..., c_i^n$. The solution $c_{\max} = (c_1^{k_1}, c_2^{k_2}, ..., c_n^{k_n})$ is defined as the expected solution of problem.

Definition 3. Given a constrained optimization problem, let its optimal solution be $x^* = (x_1^*, x_2^*, ..., x_n^*)$. Let $c_{\max}$ be the expected solution of the problem. ACO algorithm applied to a given constrained optimization problem is said to have a feedback bias if $x^* \neq c_{\max}$.

From the definition, we can see that $c_{\max}$ is the solution consists of the components with the highest expected fitness. Therefore, if feedback bias occurs in ACO, the search process is likely misled to $c_{\max}$, instead of the real optimal solution x^*. Due to such feedback bias, ant colony optimization cannot converge to the optimal solution for some problems. For instance, when ACO is applied to the *k*-cardinality tree problem, feedback bias occurs and makes it a deceptive system [2].

For understanding how the feedback bias affects the performance of ACO, we study this issue on the *k*-cardinality tree (KCT) problem, which is a generalization of the well-known minimum spanning tree problem . It is defined as follows:

Given is an undirected graph $G=(V,E)$, where $|V|=n$, $|E|=m$, with edge-weights $w(e) \in N^+, \forall e \in E$. The set of all trees in G with exactly k edges is henceforth denoted by Γ_k. The goal is to find a solution $S_k \in \Gamma_k$ that minimizes $W(S_k) = \sum_{e \in S_k} w(e)$. We assign a binary decision variable s_e to each edge $e \in E$. If $s_e = 1$ then e is part of the *k*-cardinality tree that is built. We consider the problem of solving 2-cardinality tree as shown in Figure 1.

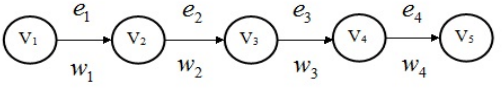


Fig. 1. Instance of 2-cardinality tree problem

The weight settings for this instance are $w_1=w_4=1$, $w_1=w_4=2$. We denote a solution using a vector $S=(s_1,s_2,s_3,s_4)$. It is obvious that the possible solutions are $S_1=(1,1,0,0)$, $S_2=(0,1,1,0)$, and $S_3=(0,0,1,1,)$. We define the fitness of the edge e with weight $w(e)$ as $f(e)=20-w(e)$, and the fitness of a solution S_k as $f(s_k) = \sum_{e \in S_k} f(e)$. Fitness of the solutions are listed in Table 1.

Table 1. Fitness of the solutions

solution	s_1	s_2	s_3	s_4	$W(S_k)$	$f(S_k)$
S_1	1	1	0	0	3	0.3
S_2	0	1	1	0	4	0.25
S_3	0	0	1	1	3	0.3

From the table, we can see that $f(S_1)$=0.3, $f(S_2)$=0.25, $f(S_3)$=0.3. Obviously, the global optimum is S_1=(1,1,0,0) and S_1=(0,0,1,1).

To solve the 2-cardinality tree problem, we use m ants in the algorithm. Each ant is represented by a bit vector $S=(s_1,s_2,s_3,s_4)$, here s_j=0,1, j=1,2,3,4. In each iteration, an ant sequentially fixes the j-th bit of S in order of j = 1, 2,3,4. For the j-th bit, the ant has two choices c_j^0 and c_j^1, corresponding to setting the j-th bit to 0 and 1 respectively. The pheromone of c_j^0 and c_j^1 are τ_j^0 and τ_j^1 respectively.

Define the set $G_j^k = \{(s_1, s_2, s_3, s_4) \in S \mid s_j = k, k = 0,1\}$ where s_j is the value of the j-th bit. Hence, G_j^k is the set of binary numbers whose j-th bit is k. We define the function $F(G_j^k) = \sum_{s \in G_j^k} f(s)$ as the summation of the fitness of all the numbers in G_j^k.

For the instance in Table 1, we have $F(G_1^0) = 0.55$, $F(G_1^1) = 0.3$, $F(G_2^0) = 0.3$, $F(G_2^1) = 0.55$, $F(G_3^0) = 0.3$, $F(G_3^1) = 0.55$, $F(G_4^0) = 0.55$, $F(G_4^1) = 0.3$. Since $F(G_1^0) > F(G_1^1)$, $F(G_2^1) > F(G_2^0)$, $F(G_3^1) > F(G_3^0)$, $F(G_4^0) > F(G_4^1)$, the expected solution of the problem is $c_{\max} = (0,1,1,0)$, which is not the real optimal solution of the problem. Feedback bias may occur in classical ACO solving 2-cardinality tree problem.

3 The Algorithm for Avoiding the Feedback Bias in ACO

To avoid the feedback searching bias in the ACO, we present an algorithm named BA-ACO (Bias-Avoiding ACO). We illustrate the algorithm taking the KCT problem as an instance.

Denote the bits of an KCT as $G=\{bit_1,bit_2,\ldots,bit_n\}$. In algorithm BA-ACO, the artificial ants travel on a digraph DG where the j-th vertex is labeled by bit_j as shown in Figure 2. There are two arcs named C_j^0 and C_j^1 linking two adjacent vertexes bit_j and bit_{j+1}. If an artificial ant at bit_j selects arc C_j^0 (or C_j^1), a bit '0' (or '1') is assigned to bit_j. When an ant completes the search from bit_1 to bit_{n+1}, the arcs on its trace form a solution. In algorithm BA-ACO, ants at node bit_j has two paths C_j^0 and C_j^1 to reach the next node bit_{j+1}. We use $P_j^k(t)$ defined in (10) to denote the probability for an ant on node bit_j to choose the path C_j^k (k = 0,1):

$$P_j^k(t) = \frac{\tau_j^k(t)\eta_j^k}{\sum_{h=0}^{1}\tau_j^h(t)\eta_j^h} \tag{1}$$

Here, $\tau_j^k(t)$ is the pheromone on the arc C_j^k between nodes bit*j* and bit_{j+1} at time *t*, and it reflects the potential tend for bit_j being assigned *k* (*k*=0,1). This forms a positive feedback conducting the ants' searching towards the optimal solution. We set the initial value of τ_j^k as $\tau_j^k(0) = F(G_j^k)$, and set the value of η_j^k as $F(G_j^k)$, ($k = 0,1$).

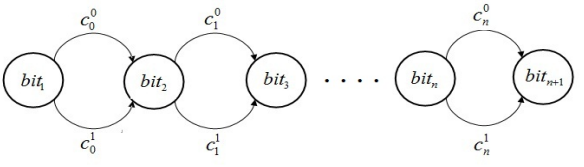


Fig. 2. The digraph

We use a fitness function to measure the quality of a scheme of bisecting. We assume that the subset of the *k*-th ant search is: $G=\{X_1,X_2,X_3,X_4,...X_n\}$. We can get the fitness of this solution from Table1.

In every iteration, algorithm BA-ACO updates the pheromone value on each arc according to formulas (2) and (3). Obviously, the more ants choose C_j^k on bit_j, the more increment of pheromone should be assigned on the arc C_j^k so that the ants could select the arc C_j^k with higher probability in the next iteration. Therefore, in each iteration, the pheromone on arc C_j^k is updated as follows.

$$\tau_j^k(t+1) = \rho\tau_j^k(t) + \Delta\tau_j^k(t) \tag{2}$$

Here,

$$\Delta\tau_j^k(t) = \frac{1}{|S_j^k|}\sum_{s\in S_j^k} f(s) \tag{3}$$

In (3), S_j^k is the set of solutions generated at the *t*-th iteration with C_j^k as the *j*-th bit.

In each iteration, algorithm BA-ACO updates the pheromone value by the best ant so far, the formula as follows:

$$\tau_j^k(t+1) = \tau_j^k(t+1) + Q_j^k(t+1) \tag{4}$$

Here,

$$Q_j^k(t) = \begin{cases} F(G_j^{1-b(j)}) & c_j^k \in s_{best} \\ 0 & otherwise \end{cases} \tag{5}$$

In (5), s_{best} is the best solution found so far, and Q is a positive constant.

We set up a threshold *Nc* to be the maximum number of iterations. The iterations should be ended when the number of iterations goes beyond *Nc*.

4 Convergence Analyses of the Algorithm BA-ACO

In this section we prove that the BA-ACO algorithm can converge to the optimal solution and can effectively avoid the searching bias for the KCT problem. First we give the following Lemmas.

Lemma 1. The pheromone in the *t*-th iteration of BA-ACO satisfies:

$$\tau_j^k(t) \le \frac{1}{1-\rho}\left[F(G_j^k) + Q\right] \quad (j = 1,2,...,n, k = 0,1) \tag{6}$$

Lemma 2. The pheromone in the *t*-th iteration of BA-ACO satisfies

$$\tau_j^0(t) \ge \frac{1-\rho^{t+!}}{1-\rho}\varphi F(G_j^0) \quad (0 < \varphi < 1, j = 1,2,...,n) \tag{7}$$

Lemma 3. For any integer n and real number $\rho \in (0,1)$, there exists a positive integer t_0 such that $\frac{1-\rho^{t+1}}{1-\rho} > t^{\frac{-1}{n}}$ for any $t > t_0$.

Proofs of Lemmas 1 to 3 are omitted due to the limited space.

From Lemmas 2 and 3, we see that when *t* is large enough, $\tau_j^0(t) \ge t^{-\frac{1}{n}}\varphi F(G_j^0)$.

We set $\tau_{\min}^0(j,t) = t^{-\frac{1}{n}}\varphi F(G_j^0)$. Since the value of $F(G_j^0)$ is independent of *j*, we can denote $F(G_j^0)$ as $F(G^0)$, and $\tau_{\min}^0(j,t)$ as $\tau_{\min}^0(t)$,

namely: $\tau_{\min}^{0}(t) = t^{-\frac{1}{n}}\varphi F(G^{0})$. We define $\tau_{\max}^{k}(j) = \frac{1}{1-\rho}\left[F(G_j^k)+Q\right]$ $(j=1,2,...,n,k=0,1)$. Since the value of $F(G_j^k)$ is independent of *j*, we define $\tau_{\max}^{k} = \frac{1}{1-\rho}\left[F(G^k)+Q\right]$, and $\tau_{\max} = \max\{\tau_{\max}^{0}, \tau_{\max}^{1}\}$.

Theorem 1. At least one ant in the BA-ACO algorithm can reach the optimum solution s^*.

Proof. Although there are multiple ants, we need only to prove that one ant can reach the optimum solution s^* in order to establish the result. Let *P*(*t*) be the probability that the ant can find the optimal solution s^* in the *t*-th iteration, and 1-*P*(*t*) be the probability that it cannot find the optima in this iteration. From Lemma 3 and Lemma 4, we know that for $t>t_0$:

$$p(c_j^k, t) = \frac{\tau_j^k(t)}{\tau_j^0(t)+\tau_j^1(t)} \geq \frac{\tau_{\min}^k(j,t)}{2\tau_{\max}}$$

Here, t_0 is as defined in Lemma 4. Therefore we have

$$p(t) \geq \left[\frac{\tau_{\min}^k(j,t)}{2\tau_{\max}}\right]^n \tag{8}$$

Let $P_{succ}(T)$ and $P_{fail}(T)$, respectively, be the probabilities that the ant does and does not find *S** in the first *T* iterations, then we have $1-P_{succ(T)} = P_{fail}(T) = \prod_{t=1}^{T}(1-P(t))$. From (17) and Lemma 4 we know:

$$p_{fail}(T) \leq \prod_{t=1}^{T}\left[1-\left[\frac{\tau_{\min}^0(j,t)}{2\tau_{\max}}\right]^n\right] \leq \prod_{t=1}^{T}\left[1-\left[\frac{\varphi F(G_j^k)}{2\tau_{\max}}\right]^n\left[t^{-\frac{1}{n}}\right]^n\right]$$

Denote $\left[\frac{\varphi F(G_j^k)}{2\tau_{\max}}\right]^n = \gamma$, then $\lim_{T\to\infty} P_{fail}(T) = \lim_{T\to\infty}\prod_{t=1}^{T}\left[1-\frac{\gamma}{t}\right]$. From [11]

We know $\lim_{T->\infty} \prod_{t=|\gamma+1|}^{T} (1-\frac{\gamma}{t}) = 0$, therefore $\lim_{T\to\infty} P_{fail}(T) = 0$, and thus

$$\lim_{T\to\infty} P_{succ}(T) = 1 \ .$$

Q.E.D.

From Theorem 1, we can see that BA-ACO algorithm can converge to the optimal solution, namely, at least one ant can reach the global optimal solution in the process.

5 Experimental Results and Analysis

To test the effectiveness of our method, we implement the BA-ACO algorithm on the KCT problem on Pentium IV, Windows XP, P1.7G, using V C++ 6. 0.

In our experiments, we set evaporation rate ρ = 0.9, the number of ants m=20. We set different fitnesses of S_0 and S_2 form 0.26 to 0.33 to test the convergence of the algorithm on KCT problem. We make 1000 trials, and record the percentage of the trials which reach the optimal solution. The test results are shown in Table 3.

Table 2. The test results of the algorithm BA-ACO and the ACO

f (Sx)	ACO			BA_ACO		
(x=0,2)	S_0	S_1	S_2	S_0	S_1	S_2
0.26	25	946	29	433	103	464
0.27	32	930	38	446	110	454
0.28	54	893	53	465	93	452
0.29	57	880	63	452	95	453
0.3	72	866	62	441	107	452
0.31	69	850	81	370	111	529
0.32	80	831	89	412	106	482
0.33	90	810	100	510	90	400

From Table 3, we can see that algorithm BA-ACO has high probability to reach the optimal solution on the KCT problem. We also test classical ACO to compare the percentage of trials reaching the optimal solution with BA-ACO. Figure 3 shows the comparison of the results.

We define $w(t) = \frac{\tau_j^{1-b(j)}(t)}{\tau_j^{b(j)}(t)}$. It is obvious that if the value of $w(t)$ approximates 0 in the iterations of an ACO algorithm, the pheromone will concentrate on the paths of the optimal solution, and the ACO can converge to the optimal solution. Figure 4 shows the value of $w(t)$ in each iteration of BA-ACO.

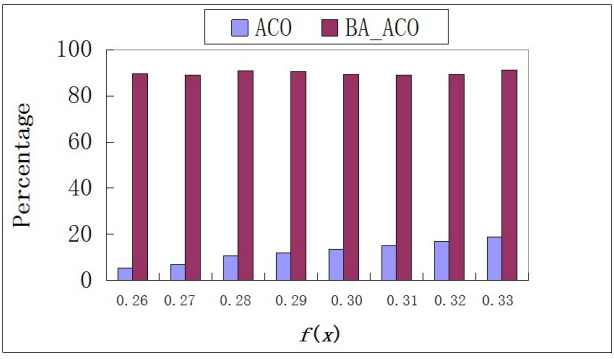


Fig. 3. Comparison of the percentage of trials reaching the convergence of solution by BA-ACO and classical ACO

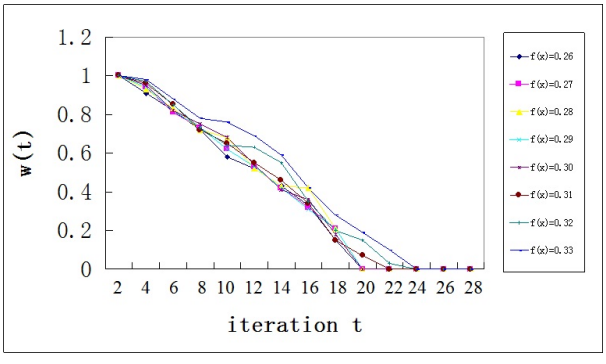


Fig. 4. The value of $w(t)$ in each iteration of BA-ACO

From Figure 4 we can see that the value of $w(t)$ decreases after iterations and convergences to 0. This confirms that $\lim_{t\to\infty} E\left[\frac{\tau_j^{1-b(j)}(t)}{\tau_j^{b(j)}(t)}\right] = 0$ in BA-ACO, and show that BA-ACO can converge to the optimal solution in solving the KCT problem.

6 Conclusions

Due to the searching bias, ant colony optimization cannot converge to the optimal solution for some problems. We define a new type of searching bias of ACO named feedback bias in addition to the other existing types of biases of ACO. We prove the existence and influence of feedback bias in ACO taking the *k*-cardinality tree problem as the test instance. We also present a method for avoiding the feedback searching

bias. Convergence analysis of our method is also given. Our experimental results confirm the correctness of our analysis and show that our method can effectively avoid the searching bias and can converge to the optimal solution.

Acknowledgments. This research was supported in part by the Chinese National Natural Science Foundation under grant Nos. 61070047、61070133 and 61003180, Natural Science Foundation of Jiangsu Province under contracts BK2010318 and BK21010134, the Scientific Research Foundation for Graduated Students in Jiangsu Province, and Natural Science Foundation of Education Department of Jiangsu Province under contract 09KJB20013.

References

1. Dorigo, M., Stützle, T.: Ant Colony Optimization. MIT Press (2004)
2. Dorigo, M., Blum, C.: Ant colony optimization theory: A survey. Theoretical Computer Science 344, 243–278 (2005)
3. Blum, C.: Ant colony optimization: Introduction and recent trends. Physics of Life Reviews 2, 353–373 (2005)
4. Shtovba, S.: Ant Algorithms: Theory and Applications. Programming and Computer Software 31, 167–178 (2005)
5. Blum, C., Sampels, M.: Ant colony optimization for FOP shop scheduling: A case study on different pheromone representation. In: Proceedings of Congress on Evolutionary Computation 2002, vol. 2, pp. 1558–1563 (2002)
6. Blum, C., Sampels, M.: When Model Bias Is Stronger than Selection Pressure. In: Guervós, J.J.M., Adamidis, P.A., Beyer, H.-G., Fernández-Villacañas, J.-L., Schwefel, H.-P. (eds.) PPSN VII 2002. LNCS, vol. 2439, pp. 893–902. Springer, Heidelberg (2002)
7. Merkle, D., Middendorf, M.: Modeling the dynamics of ant colony optimization algorithms. Evolutionary Computation 10(3), 235–262 (2002)
8. Merkle, D., Middendorf, M.: Modelling ACO: Composed Permutation Problems. In: Dorigo, M., Di Caro, G.A., Sampels, M. (eds.) ANTS 2002. LNCS, vol. 2463, pp. 149–162. Springer, Heidelberg (2002)
9. Montgomery, J., Randall, M., Hendtlass, T.: Solution bias in ant colony optimization: Lessons for selecting pheromone models. Computers and Operations Research 35(9), 2728–2749 (2008)
10. Montgomery, J., Randall, M., Hendtlass, T.: Structural Advantages for Ant Colony Optimisation Inherent in Permutation Scheduling Problems. In: Ali, M., Esposito, F. (eds.) IEA/AIE 2005. LNCS (LNAI), vol. 3533, pp. 218–228. Springer, Heidelberg (2005)
11. Chen, L., Sun, H., Wang, S.: First Order Deceptive Problem of ACO and Its Performance Analysis. Journal of Networks 4(10), 993–1000 (2009)
12. Blum, C., Dorigo, M.: Deception in Ant Colony Optimization. In: Dorigo, M., Birattari, M., Blum, C., Gambardella, L.M., Mondada, F., Stützle, T. (eds.) ANTS 2004. LNCS, vol. 3172, pp. 118–129. Springer, Heidelberg (2004)
13. Blum, C., Dorigo, M.: Search bias in ant colony optimization: on the role of competition-balanced systems. IEEE Transactions on Evolutionary Computation 9, 159–174 (2005)
14. Gutjahr, W.J.: A Graph-based Ant System and its convergence. Future Generation Computer Systems 16, 873–888 (2000)

15. Gutjahr, W.J.: ACO algorithms with guaranteed convergence to the optimal solution. Info. Processing Lett. 82, 145–153 (2002)
16. Stützle, T., Dorigo, M.: A Short Convergence Proof for a Class of Ant Colony Optimization Algorithms. IEEE Transactions on Evolutionary Computation 6, 358–365 (2002)
17. Blum, C., Dorigo, M.: The hypercube framework for ant colony optimization. IEEE Transaction on Systems, Man, and Cybernetics—Part B 43, 1161–1172 (2004)
18. Blum, C.: Theoretical and practical aspects of ant colony optimization. Dissertations in Artificial Intelligence, vol. 2463. Akademische Verlagsgesellschaft Aka CmbH, Berlin (2004)

Novel Binary Biogeography-Based Optimization Algorithm for the Knapsack Problem

Bingyan Zhao[1], Changshou Deng[2,*], Yanling Yang[2], and Hu Peng[2]

[1] School of Business, Jiujiang University, Jiujiang 332005, China
[2] School of Information Science and Technology, Jiujiang University, Jiujiang 332005, China
petramm@163.com,{csdeng,yangyanlin,xx_penghu}@jju.edu.cn

Abstract. Mathematical models of biogeography inspired the development of the biogeography-based optimization algorithm. In this article we propose a binary version of biogeography-based optimization (BBO) for the Knapsack Problem. Two new mutation operators are proposed to extend the biogeography-based optimization algorithm to binary optimization problems. We also demonstrate the performance of the resulting new binary Biogeography-based optimization algorithm in solving four Knapsack problems and compare it with that of the standard Genetic Algorithm. The simulation results show that our new method is effective and efficient for the Knapsack problem.

Keywords: Knapsack Problem, Biogeography-based optimization, Migration operator, mutation operator.

1 Introduction

Knapsack problems have been widely studied not only because of their immediate application in industry and financial management, but also more pronounced for theoretical reasons. Many industrial problems such as resource allocation, investment decision-making, and hold loading can be modeled as Knapsack problems. The Knapsack problem can be described as follows. Given a set of items, each with a value v_i and a weight w_i, determine how often each item should be included in a collection so that the total weight is less than or equal to a given limit (W) and the total value is as large as possible. The most common formulation of the problem is the 0-1 Knapsack problem, where each item can either be included one time or not at all, which can be mathematically formulated as:

$$\begin{aligned} &Maximize \quad f = \sum_{i=1}^{n} v_i x_i \\ &s.t. \quad \sum_{i=1}^{n} w_i x_i \le W, \qquad x_i \in \{0,1\} \end{aligned} \tag{1}$$

* Corresponding author. csdeng@jju.edu.cn

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 217–224, 2012.

Recently, many exact algorithms and nature-inspired algorithms have been proposed to cope with the Knapsack problem [1-8]. However, several methods of them often can only provide locally optimal solutions.

The biogeography-based optimization algorithm is a new nature-inspired computation technique proposed by Simon in 2008 [9]. It is originally designed for unconstrained integer programming problems. In this paper, we propose a novel binary BBO for Knapsack problems. To the best of our knowledge, this is the first application of BBO to the Knapsack problem.

The rest of the paper is organized as follows. Section 2 gives an overview of BBO. A novel Binary BBO with two mutation operators for the Knapsack problems is presented in section 3. Four Knapsack Problems with different sizes are used to compare the performance of the Binary BBO with that of the Genetic Algorithm in section 4. Section 5 concludes this paper.

2 Biogeography-Based Optimization Algorithm

2.1 Mathematical Model of BBO

Biogeography studies the migration, speciation, and extinction of species from one island to another. With the mathematical models in the Biogeography, Simon (2008) proposed the BBO algorithm [9], which is an example of how a natural process can be modeled to solve general optimization problems. Islands that are well suited as habitats for biological species are said to have a high habitat suitability index (HSI). Islands with a low HSI have a small number of species, whereas islands with a high HSI have many species that emigrate to nearby islands because of the accumulation of random effects on its large populations. Emigration occurs as animals ride flotsam, fly, or swim to neighboring islands [10, 11].

Suppose that we have some problem, and that we also have several candidate solutions. A good solution is analogous to an island with a high HSI, and a poor solution is like an island with a low HSI. High HSI solutions are more likely to share their features with other solutions, and low HSI solutions are more likely to accept shared features from other solutions. This approach to problem solving is called biogeography-based optimization.

2.2 Migration Operator

In biogeography, species may migrate between islands. In BBO, the solution features may affect each other between islands. For the sake of simplicity, we set $E = I$ and two solutions are given in Figure 1. Figure 1 illustrates the migration curves along with two solutions. S_1 represents a poor solution and S_2 represents a good solution. The immigration rate for S_1 will therefore be larger than the immigration rate for S_2, and the emigration rate for S_1 will be smaller than the emigration rate for S_2.

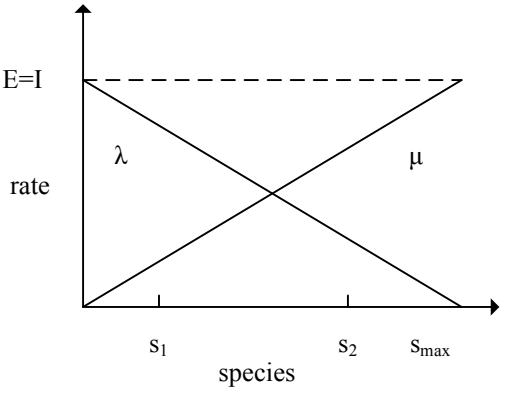


Fig. 1. Illustration of two solutions

The BBO uses the migration rates of each solution to probabilistically share features between solutions. The migration operator in the BBO is as follows. The immigration curve is used to probabilistically decide each feature whether or not to immigrate in each solution. If a given solution feature is selected to be immigrated, then the emigrating island is selected probabilistically. The pseudo code of the migration operator is presented in Figure 2.

```
// NP is the size of population, D is the dimension size
1：For i=1 to NP
2：Select the H_i with probability λ_i
3：If H_i is selected, then
     For j=1 to D
4：Select the H_j with probability μ_i
5: If H_j is selected, then
6:   to generate a random integer k ∈ [1, D]
7: let H_ik = H_jk
8：endfor
9: endfor
```

Fig. 2. Migration Operator

2.3 Mutation Operator

Cataclysmic events can drastically change the HSI of a natural habitat. An island habitat's HSI can therefore change suddenly due to apparently random events. In the BBO, a mutation operator can be derived from this. The species count

probabilities are used to determine mutation rates. The mutation rate is given in formula (2) [10].

$$m(S) = m_{\max}(\frac{1-P_s}{P_{\max}}) \quad . \tag{2}$$

where $m_{\max}$ is a user-defined parameter.

The mutation operator is given in figure 3.

```
// NP is the size of population, D is the dimension size
1：for i=1 to NP
2：Using the formula (2)to calculate mutation probability P_i
3：select SIV in H_i(j) with probability P_i(j)
4：if H_i(j) is selected, then
5：Replace the H_i(j) with a random generated SIV
6: endfor
```

Fig. 3. Mutation Operator

2.4 Elitism

In order to achieve global convergence, similar to the common practice in GAs, elitism is incorporated into BBO. The best solution in the current population is preserved from one generation to the next.

3 Binary Biogeography-Based Optimization Algorithm for Knapsack Problem

The traditional BBO uses an integer coding scheme. The Knapsack Problem is a constrained zero-one programming problem. The migration operator in the conventional BBO is inherited in the binary BBO and two new mutation operators were used as well.

3.1 Migration Operator

The migration operator in the BBO is used in the binary BBO as is.

3.2 Binary Mutation Operator

The mutation operator in binary BBO, defined in Figure 4, uses formula (2) to decide the probability of each bit number in the population to flip.

```
// NP is the size of population, D is the dimension size
1：for i=1 to NP
2：Using the formula (2) to calculate mutation
   probability P_i
3：select SIV in H_i(j) with probability P_i(j)
4：if H_i(j) is selected, then
5：Replace the H_i(j) with 1-H_i(j)
6: endfor
```

Fig. 4. Binary Mutation Operator

3.3 Greedy Mutation Operator for the Infeasible Solutions

During the optimization process in BBO, infeasible solutions may occur. An additional greedy mutation operator was defined in binary BBO to repair these infeasible solutions. Firstly, it sorts the items according to v_i / w_i in ascending order. Then, when an infeasible solution is found, it flips the features if they are ones in order of v_i / w_i until the solution becomes feasible.

4 Numerical Experiments

In this section, we investigate the performance of binary BBO solving Knapsack Problems of different scales. Four benchmark problems from [5-8] (thereafter they are denoted by Kp1, Kp2, Kp3 and Kp4 respectively) were used to compare the binary BBO to the standard Genetic Algorithm (GA).

4.1 Results

The parameter setting of the binary BBO for each problem is presented in table 1.

Table 1. Parameters of the binary BBO

Problems	E	I	Population size	$m_{\max}$	Elitism size	Maximum Generation
Kp1	1	1	100	0.005	2	100
Kp2	1	1	250	0.005	2	100
Kp3	1	1	250	0.005	2	100
Kp4	1	1	250	0.005	2	100

For comparison, the standard GA with single point crossover with a crossover probability 0.8, and a mutation probability of 0.05 is used. In the GA, we deal with

the infeasible solutions using the same greedy mutation operator from the binary BBO. The population size and the maximum generation number of the GA are the same as that of the binary BBO for each problem.

For each of the four Knapsack Problems, 100 runs have been conducted and the best results (Best), average results (Avg), worst results (Worst), standard deviations (Dev) and the success rate (SR) of the binary BBO are shown in Table 2.

Table 2. Compared results between binary BBO and GA

Examples	Algorithms	Best	Avg	Worst	Dev	SR
Kp1	Binary BBO	1042	1041.1	1037	1.9714	81%
	GA	1042	1040.2	1037	2.4121	64%
Kp2	Binary BBO	3119	3118.5	3113	1.4530	82%
	GA	3116	3090.4	3060	13.6569	0%
Kp3	Binary BBO	26559	26554	26534	9.4814	80%
	GA	25435	2432.2	23076	534.8347	0%
Kp4	Binary BBO	5375	5361.9	5343	13.1674	25%
	GA	5326	5298.5	5269	11.3649	0%

Figures 5 and 6 present the compared average convergence of the 100 independent trials for the four Knapsack Problems. They clearly show that the average convergence speed of binary BBO is superior to that of the GA for each benchmark problem.

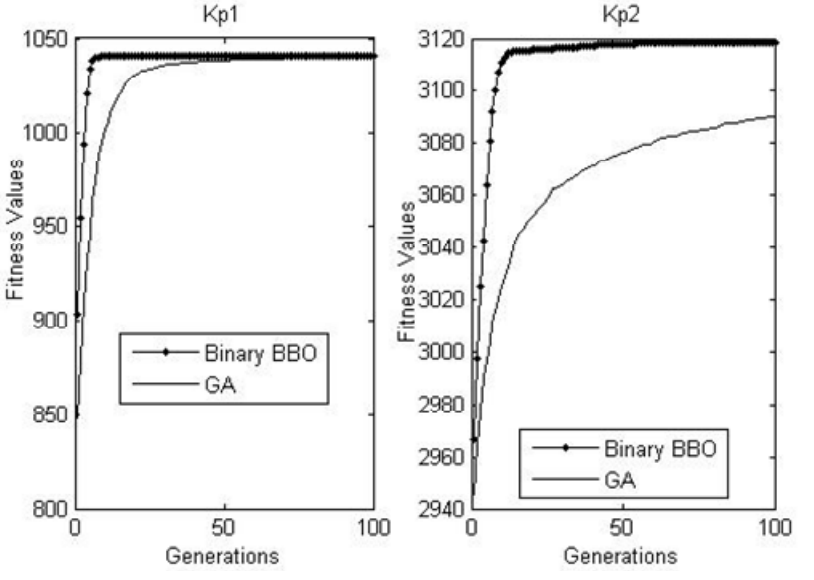


Fig. 5. Convergence speed of binary BBO versus GA for Kp1 and Kp2

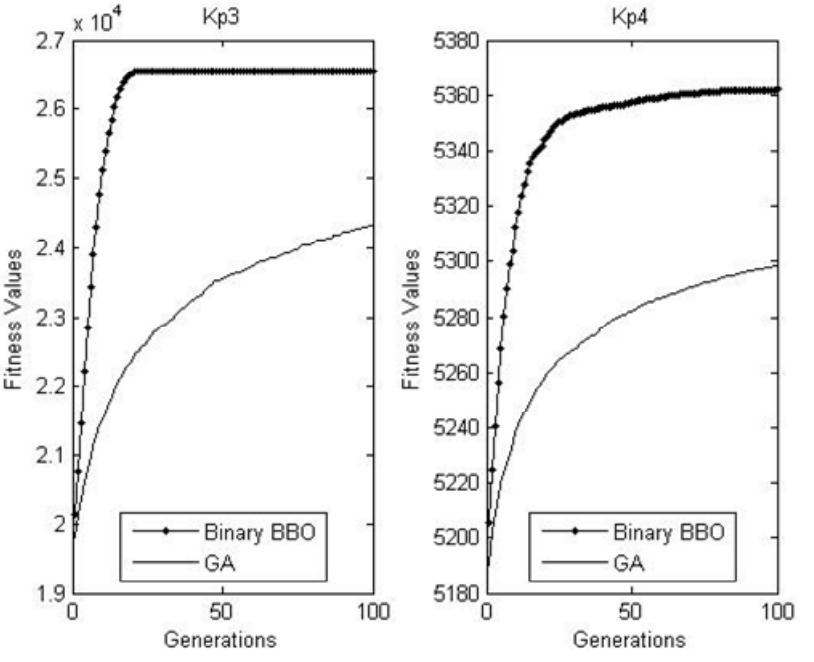


Fig. 6. Convergence speed of binary BBO versus GA for Kp3 and Kp4

5 Conclusion

In this article, a binary version of BBO has been proposed for the solution of Knapsack Problems in this paper. A binary mutation operator was used to improve the exploration ability of the algorithm and a greedy mutation operator was used to handle infeasible solutions of the Knapsack Problem. Simulation results show that the performance of the binary BBO is superior to that of the standard GA. Application of this binary BBO to other combinatorial optimization problems will be our future work.

Acknowledgment. This work is partially supported by the Science and Technology Foundation of Jiangxi Province under Grant No. GJJ11616.

References

1. Masako, F., Takeo, Y.: An exact algorithm for the Knapsack sharing problem with common items. European Journal of Operational Research 171(2), 693–707 (2006)
2. Didier, E.B., Moussa, E.: Load balancing methods and parallel dynamic programming algorithm using dominance technique applied to the 0-1 Knapsack Problem. Journal of Parallel and Distributed Computing 65(1), 74–84 (2005)

3. Stephen, C.H., Leung, Z.D.F., Zhou, C.L., Wu, T.: A hybrid simulated annealing metaheuristic algorithm for the two-dimensional Knapsack packing problem. Computers & Operations Research 39(1), 64–73 (2010)
4. Kong, M., Peng, T., Kao, Y.C.: A new ant colony optimization algorithm for the multidimensional Knapsack Problem. Computers & Operations Research 35(8), 2672–2683 (2008)
5. Maria, J.A., Marla, A., Motg, A.: A multiobjective Tchebycheff based genetic algorithm for the multidimensional Knapsack Problem. Computers & Operations Research 34(11), 3458–3470 (2007)
6. Zhang, L., Zhang, B.: Good point set based genetic algorithm. Chinese Journal of Computers 24(9), 917–922 (2001)
7. Shen, X.J., Wang, W.W., Zheng, P.J.: Modified particle swarm optimization for 0/1 Knapsack problem. Computer Engineering 32(18), 23–24, 38 (2006)
8. Gao, T., Wang, M.G.: The research for the reductive dimension and replacive variable algorithm of special restrict ILP. Systems Engineering Theory Methodology Applications 11(2), 125–130 (2002)
9. Simon, D.: Biogeography-based optimization. IEEE Trans. Evol. Comput. 12(6), 702–713 (2008)
10. Simon, D., Mehmet, E., Dawei, D.: Population Distributions in Biogeography-based Optimization algorithms with Elitism. In: SMC, pp. 1–6 (2009)
11. Simon, D., Mehmet, E., Dawei, D., Rick, R.: Markov Models for Biogeography-Based Optimization. IEEE Transactions on Systems, Man, and Cybernetics—PART B: Cybernetics 41(1), 299–306 (2011)

Path Planning Based on Voronoi Diagram and Biogeography-Based Optimization

Ning Huang, Gang Liu, and Bing He

Xi'an Hongqing Research Institute of High-tech, Xi'an, 710025, China
hn520425@163.com

Abstract. In this paper, an approach of cruise missile hierarchical path planning based on Voronoi diagram and Biogeography-Based Optimization (BBO) is proposed. First, based on Voronoi diagram, we establish the threat model to describe the planning environment and generate the initial paths and navigation nodes. Then the Biogeography-Based Optimization (BBO) is utilized to search the optimal path. In order to improve the performance of BBO, we adopt an integer priority-based encoding, analyze and discuss the migration rate model and design the migration, mutation and elite operator. Finally, the simulation results show that this approach is effective in cruise missile path planning.

Keywords: Cruise missile, path planning, BBO, Voronoi diagram.

1 Introduction

In recent years, cruise missile has become the main arm in modern warfare. The purpose of cruise missile path planning is, under certain given conditions, to find out the optimal path between start point and target point with the information of terrain and battlefield. Typically, because of complex constraint conditions and extensive planning area, cruise missile path planning is a nonlinear global optimization problem. It is difficult to be solved effectively by general optimization algorithm.

By studying the laws of nature, the establishment of bionic algorithm to solve engineering problems has been an important research embranchment of intelligent optimization algorithm. There have been a series of intelligent algorithms such as ACO [1], ES [2], GAs [3], PSO [4], etc. Biogeography-Based Optimization (BBO) [5] is a new global optimization algorithm, it applies the biogeography theory in solving global optimization problems. BBO uses biological habitat to simulate the solution to problem and depends on the migration of species between different habitats to exchange information. BBO improves the habitat suitability index (HSI) and gets the solution by adjusting the migration and mutation strategies.

We propose a Voronoi diagram and BBO-based approach of path planning based on hierarchical planning. Firstly, the threat model is established under the base of Voronoi to describe the planning environment and the initial paths and navigation nodes are generated. Then, BBO is utilized to search the optimal path.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 225–232, 2012.

2 Cruise Missile Path Planning Model

2.1 Planning Environment Description Based on Voronoi Diagram

Voronoi diagram was put forward by Russian mathematician G. Voronoi first in 1908. In two-dimensional plane, Voronoi diagram is composed of many polygons that are composed of perpendicular bisector between every two adjacent points.

We regard enemy radars as threat points to build Voronoi diagram, establish the threat model and generate the initial paths and navigation nodes. Fig. 1 illustrates a Voronoi diagram which has 30 threat points. In this graphic, the node is navigation node $p_i(i=1,2,\ldots,N)$ and the edge is the path segment whose threat is the least when cruise missile get through two adjacent threat points. We connect start and target point to four nodes which are closest to them, evaluate the cost weight for each edge based on cost function. In this case, the Voronoi diagram equals a weighted graph.

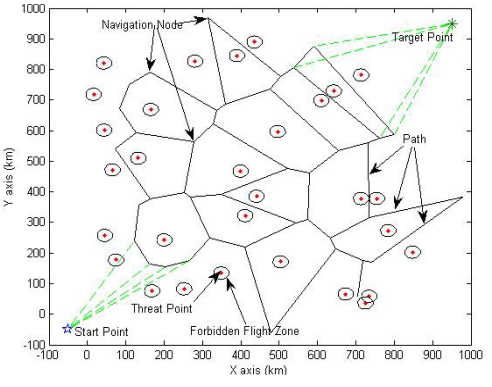


Fig. 1. Planning Environment Description Based on Voronoi Diagram

2.2 Path Description

The purpose of path planning is to search out a path start from start point, along with the edges of Voronoi diagram and pass through some navigation nodes sequentially, eventually reach target point. It can be described as $P=\{start,p_1,p_2,\ldots,p_N,target\}$.

2.3 Constraint Conditions

In path planning, we must consider some constraint conditions such as the way through the target area, speed, mobility, required arrival time and so on [6]:

The Shortest Direct Flight Distance l_{min}. The distance of path segment l_i must not be less than the shortest direct flight distance l_{min} : $l_i \geq l_{min}$.

The Longest Missile Range L_{max}. The total distance L of the path must be less than the missile range L_{max}: $L \leq L_{max}$.

The Maximum Number of Navigation Nodes N_{max}. The number of navigation nodes of the path which is set pre-launch must be less than the maximum: $N \le N_{max}$.

The Maximum Turning Angle Φ_{max}. During the cruise missile attitude adjustment in navigation node p_i, the turning angle Φ_i must be less than the maximum Φ_{max}. If the coordinates of navigation node p_i are (x_i,y_i), the vector of the i^{th} path segment is $\vec{r}_i = (x_i - x_{i-1}, y_i - y_{i-1})$. Thus, the turning angle Φ_i should satisfy with

$$\cos\Phi_i = \frac{(\vec{r}_i^{\,T} \cdot \vec{r}_{i+1})}{\left|\vec{r}_i^{\,T}\right| \cdot \left|\vec{r}_{i+1}\right|} \ge \cos\Phi_{\max} \tag{1}$$

Environment Constraints of the Battlefield. Cruise missile flight also should subject to some other constraint conditions such as avoid zone or no-fly zone, geographical environment, the direction of enter the target, etc.

2.4 Cost Function

Evaluating the path $P=\{start,p_1,p_2,\ldots,p_N,target\}$ to survival of the fittest is the key to guide the algorithm. There are two aspects to be considered:

The first one is fuel cost. The fuel of the cruise missile must be enough to ensure missile reach the target point along with the pre-planned path and particular speed. Assuming a constant speed cruise flight, fuel cost is equivalent to distance cost.

The second one is threat cost. The threat cost is the total threat one missile suffered from each threat point $t_j(j=1,2,\ldots,M)$. As radar's detection capability and the fourth power of the distance $R_j(x)$ between missile position x and radar point t_j are inversely proportional. So that, we have

$$f_{TAj}(x) = K_j / R_j^{\,4}(x) \tag{2}$$

where K_j is the strength of the threat point t_j. We assume that all of the threat points are same and their strength is K. In order to make the calculation more accurate, we calculate 1/10, 3/10, 1/2, 7/10 and 9/10 five points of each path segment:

$$f_i = \frac{1}{5} l_i K \sum_{j=1}^{M} \left(\frac{1}{d^4_{\frac{1}{10},i,j}} + \frac{1}{d^4_{\frac{3}{10},i,j}} + \frac{1}{d^4_{\frac{5}{10},i,j}} + \frac{1}{d^4_{\frac{7}{10},i,j}} + \frac{1}{d^4_{\frac{9}{10},i,j}}\right) \tag{3}$$

where $d_{1/10,i,j}$ is the distance between t_j and the 1/10 position of the path segment l_i.

In a word, we have the cost function as follows:

$$f = \sum_{i=1}^{N+1} [\omega l_i + (1-\omega) f_i], 0 \le \omega \le 1 \tag{4}$$

where ω is the weight parameter, it expresses the relationship between fuel cost and threat cost. It is determined by tactical requirement or commander's decision.

2.5 The Mathematical Model of Cruise Missile Path Planning

In summary, path planning can be expressed as following optimization model:

$$\begin{cases} f = \min \sum_{i=1}^{N+1} [\omega l_i + (1-\omega) f_i] \\ s.t. \\ \quad l_i \geq l_{\min} \\ \quad L = \sum_{i=1}^{N+1} l_i \leq L_{\max} \\ \quad N \leq N_{\max} \\ \quad \cos \Phi_i = \dfrac{(\vec{r}_i^{\,T} \cdot \vec{r}_{i+1})}{\left|\vec{r}_i^{\,T}\right| \cdot \left|\vec{r}_{i+1}\right|} \geq \cos \Phi_{\max} \end{cases} \tag{5}$$

3 Biogeography-Based Optimization

Biogeography was proposed by Alfred Wallace [7] and Charles Darwin [8] in the 19th century, until the 1960s, Robert MacArthur and Edward Wilson [9] perfected it and formed an independent discipline. BBO was formally proposed by Dan Simonin 2008. As can be seen from the performance of BBO applied for 14 benchmarks and compared with several other algorithms, it has good convergence and stability [5].

3.1 Encoding and Decoding of Suitability Index Vector (SIV)

In BBO, each habitat corresponds to a solution to the problem. Whether the habitat is suitable for living or not is described by habitat suitability index (HSI). There are several natural factors are related to HIS such as temperature, humidity, rainfall, etc. We call them suitability index variables (SIVs). The composition of these SIVs is named as suitability index vector (SIV), this SIV is the solution to the problem.

In this paper, we adopt integer priority-based encoding to encode the SIV. The coding sequence $\{x_{i1},x_{i2},\ldots,x_{iN}\}$ is the SIV of habitat i, each encoded bit j of it corresponds to a navigation node p_j, the value of x_{ij} is an integer and $x_{ij} \in [1,N]$, $x_{ij} \neq x_{ik}$. Start point and target point are not involved in encoding process but in decoding. In the decoding process, start from the start point, find out the highest priority node from its adjacent nodes and add it to the path sequence, loop until the target point is added into. During the decoding process, in order to avoid duplicate choosing, the priority of the selected nodes should be adjusted to the lowest and the priority of the target point is the highest always. An example of decoding is shown in Fig. 2.

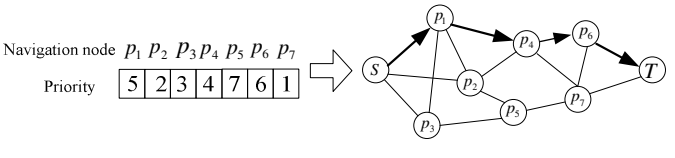


Fig. 2. Decoding of suitability index vector

3.2 Biogeography Migration Rate Model

MacArthur and Wilson [9] adopted a linear migration rate model which is shown in Fig. 3. The abscissa S is the species count of habitat and the ordinate is the rate of migration, λ and μ are the immigration rate and emigration rate.

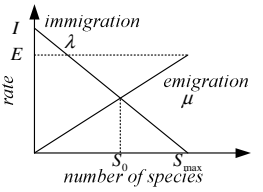


Fig. 3. A simple linear biogeography migration rate model [9]

We define P_S as the probability of the species count. It is calculated as follows [5]: P_S changes from time t to time $(t+\triangle t)$ as follows:

$$P_S(t+\Delta t) = P_S(t)(1-\lambda_S\Delta t-\mu_S\Delta t)+P_{S-1}\lambda_{S-1}\Delta t+P_{S+1}\mu_{S+1}\Delta t \tag{6}$$

where λ_S and μ_S are immigration and emigration rates when there are S species. We assume that $\triangle t$ is small enough to ignore the probability of immigration or emigration. We define $P=[P_0,P_1,\ldots,P_n]^T$, $n=S_{max}$. From (6) we have

$$P_S^{'}(t) = \lim_{\Delta t\to 0}\frac{P_S(t+\Delta t)-P_S(t)}{\Delta t} = -(\lambda_S+\mu_S)P_S(t)+P_{S-1}\lambda_{S-1}+P_{S+1}\mu_{S+1} \tag{7}$$

$$P_S^{'} = \begin{cases} -(\lambda_S+\mu_S)P_S+\mu_{S+1}P_{S+1} & S=0 \\ -(\lambda_S+\mu_S)P_S(t)+\lambda_{S-1}P_{S-1}+\mu_{S+1}P_{S+1} & 1\le S\le S_{\max}-1 \\ -(\lambda_S+\mu_S)P_S(t)+\lambda_{S-1}P_{S-1} & S=S_{\max} \end{cases} \tag{8}$$

Now, we can arrange the above equations into a single matrix equation: $P'=AP$.

$$A=\begin{bmatrix} -(\lambda_0+\mu_0) & \mu_1 & 0 & \cdots & 0 \\ \lambda_0 & -(\lambda_1+\mu_1) & \mu_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \lambda_{n-2} & -(\lambda_{n-1}+\mu_{n-1}) & \mu_n \\ 0 & \cdots & 0 & \lambda_{n-1} & -(\lambda_n+\mu_n) \end{bmatrix} \tag{9}$$

For the simple linear migration rate model given by Fig.3, we have $\mu_k=Ek/n$ and $\lambda_k=I(1-k/n)$, and the k is the species count of the habitat. If $E=I$, we have $\lambda_k+\mu_k=E$ and the probability of each species count is given by

$$P_k = \begin{cases} P_0 = \dfrac{1}{1+\sum_{l=1}^{n}\dfrac{\lambda_0\lambda_1\ldots\lambda_{l-1}}{\mu_1\mu_2\ldots\mu_l}}, k=0 \\ P_k = \dfrac{\lambda_0\lambda_1\ldots\lambda_{k-1}}{\mu_1\mu_2\ldots\mu_k\left(1+\sum_{l=1}^{n}\dfrac{\lambda_0\lambda_1\ldots\lambda_{l-1}}{\mu_1\mu_2\ldots\mu_l}\right)}, 1\le k\le n \end{cases} \tag{10}$$

3.3 Operation of BBO

Migration. Whether the habitat i is selected to be modified or not depends on the global migration probability $P_{mod} \in [0,1]$. If it is selected to be modified, we use its immigration probability $\lambda(S_i)$ to determine which suitability index variable $x_{ij}(j=1,2,\ldots,D)$ is selected to be modified. If x_{ij} is selected to be modified, then we use the emigration rates $\mu(S_k)(k \neq I, k \in [1,n])$ of other habitats to determine which habitat is selected to emigrate SIV to x_{ij}.

Fig. 4. Migration operation

Mutation. Whether SIV x_i of habitat i will mutate into a new form or not based on the probability $P_S(i)$, $i \in [1,n]$ of species count of habitat i. How to calculate the mutation probability $m(x_i)$ is the key of mutation operation. Mutation probability $m(x_i)$ and the probability $P_S(i)$ is inversely proportional, the function [5] is given in (11):

$$m(x_i) = m_{\max}\left(\frac{1 - P_S(i)}{P_{\max}}\right) \tag{11}$$

where $m_{\max}$ is the largest mutation probability and $P_{\max}$ is the maximum of $P_S(i)$.

In order to ensure the encoding priority of the SIV to maintain the uniqueness of each other, we randomly select two encoding bits j and k from x_i and exchange them according to probability $m(x_i)$. Fig.5 illustrates the process of mutation operation.

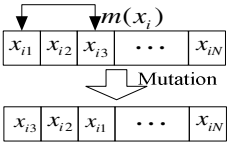


Fig. 5. Mutation operation

Elite Operator. We sort all the suitability index vectors by their HIS and retain z suitability index vectors of maximum HIS. These vectors do not participate in immigration and mutation operation. In this case, the number of elite z is defined during the initialization phase of the algorithm.

3.4 Algorithm Description

Generally, the approach of cruise missile hierarchical path planning based on Voronoi diagram and BBO can be described as follows:

Step 1: Initialize the planning environment.
Step 2: Initialize these parameters of BBO.

Step 3: Encode the SIV and generate the initial populations.
Step 4: Calculate each HIS $f(x_i)$,i=1,2,…,n, species count S_i, immigration rates $\lambda(S_i)$ and emigration rates $\mu(S_i)$. Then sort of HIS and retain the elite individuals.
Step 5: Perform the migration operation and recalculate $f(x_i)$.
Step 6: Update the probability of the species count and perform the mutation operation, then recalculate each $f(x_i)$;
Step 7: Whether reach the termination condition? Yes to Step 8, no to Step 4.
Step 8: Output of the algorithm results.

4 Simulation

We simulate the algorithm and compare it with Dijkstra. The planning zone is 1000 km × 1000 km. We randomly generate some threat points in the zone, the radius of no-fly zone around the threat point is 20km. The coordinates of start point is (0,0) and the target point is (900,900). The population of BBO is 50 and run for 100 generations. ω=0.6, z=2, P_{mod}=1, $m_{\max}$=0.01, I=E=1. The simulation results as follows:

Table 1. Results of simulation

	The number of threat points	The value of objective function f			
		$f_{Dijkstra}$	f_{BBO}	Δf	$\Delta f / f_{Dijkstra}(\%)$
1	30	533.5884	533.5884	0	0
2	30	560.9514	560.9514	0	0
3	30	560.8053	560.8053	0	0
4	50	559.6701	570.5437	-10.8736	-1.94286
5	50	557.8746	557.8819	-0.0073	-0.00131
6	50	556.3643	556.3643	0	0
7	70	544.2603	564.5483	-20.288	-3.72763
8	70	565.5556	579.829	-14.2734	-2.52378
9	100	571.3041	628.8392	-57.5351	-10.0708
10	100	577.0269	614.1059	-37.079	-6.42587

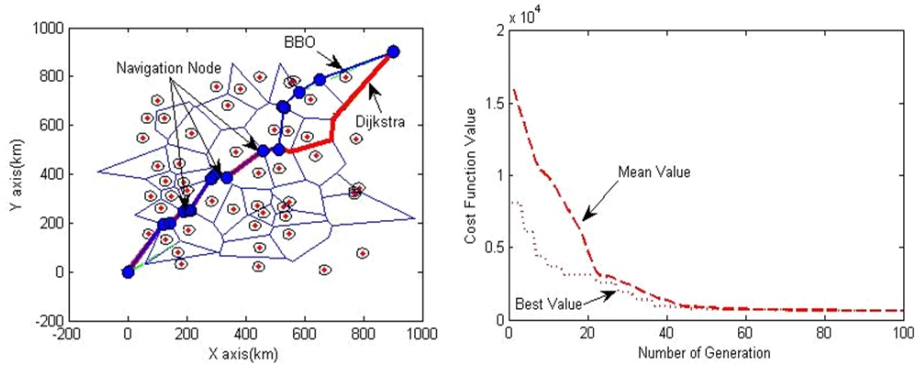


Fig. 6. The result of path planning and the trend of cost function value of BBO changes with the number of generations. In this test, the number of threat points is 50.

Simulation results show that the approach could work out a good path effectively. When the number of the threat points is small, BBO could work out the theoretical optimal path. However, the error of BBO increases with the increase of the number of treat points. We conclude that the reason of it includes two aspects:

1. The length of the SIV will increase with the increase of problem scale.
2. In this paper, the biogeography migration rate model in BBO is a simple linear model. It couldn't describe the natural situation accurately.

5 Conclusion

This paper studied the application of BBO on cruise missile path planning. First, we established the threat model to describe the planning environment and generated the initial paths and navigation nodes. Then we encoded the SIV based on priority and established the migration rate model, designed the migration, mutation and elite operator. Finally, we demonstrated the performance of BBO. Overall, judging from the simulation results, BBO has a good performance and the approach of cruise missile hierarchical path planning is effective. However, as the development time is short, many aspects of BBO are still not perfect such as SIV code, migration rate model, etc. It is still looking forward to more research in-depth.

References

1. Dorigo, M., Gambardella, L.M.: Guest editorial: special section on ant colony optimization. IEEE Transactions on Evolutionary Computation 6, 317–319 (2002)
2. Mezura-Montes, E., Coello, C.A.C.: A simple multimembered evolution strategy to solve constrained optimization problems. IEEE Transactions on Evolutionary Computation 9, 1–17 (2005)
3. Man, K.F., Tang, K.S.: Genetic Algorithms, 1st edn. Springer, London (1999)
4. Eberhart, R.C., Shi, Y.: Guest editorial: special issue on particle swarm optimization. IEEE Transactions on Evolutionary Computation 8, 201–203 (2004)
5. Simon, D.: Biogeography-based Optimization. IEEE Transactions on Evolutionary Computation 12, 702–713 (2008)
6. Rathbun, D., Capozzi, B.: Evolutionary Approaches to Path Planning Through Uncertain Environments. In: AIAA's 1st Technical Conference and Workshop on Unmanned Aerospace Vehicles, Systems, Technologies, and Operations, Portsmouth, Virginia (2002)
7. Wallace, A.R.: The Geographical Distribution of Animals. Adamant Media Corporation, Boston (2005)
8. Darwin, C.: The Origin of Species, Gramercy, New York (1995)
9. MacArthur, R.H., Wilson, E.O.: The Theory of Island Biogeography. Princeton University, New Jersey (1967)

Unconscious Search – A New Structured Search Algorithm for Solving Continuous Engineering Optimization Problems Based on the Theory of Psychoanalysis

Ehsan Ardjmand and Mohammad Reza Amin-Naseri*

Department of Industrial Engineering, Tarbiat Modares University, Ale Ahmad Highway, P.O. Box 14115-143,Tehran, Iran
{ardjmand,amin_nas}@modares.ac.ir

Abstract. Many metaheuristic methods are based on the ability of systems in Nature to optimize on aspects of their performance. One such system is the human brain with its capacity for optimizing towards a general state of mental balance. The Theory of Psychoanalysis propounded by Sigmund Freud is generally recognized as an account of the mechanisms involved in psychological processes. It is possible to draw an analogy between the practice of psychoanalysis and the treatment of optimization problems. The proposed new Unconscious Search (US) method shares in some features with the procedure attempted in psychoanalysis to elicit the suppressed contents of the subject's mind. One bounded and several unbounded benchmark problems have been solved using the proposed algorithm; the results were satisfactory when compared against earlier results obtained using other known methods.

Keywords: Unconscious Search, Psychoanalysis, Metaheuristic, Optimization.

1 Introduction

"Metaheuristics, in their original definition, are solution methods that orchestrate an interaction between local improvement procedures and higher level strategies to create a process capable of escaping from local optima and performing a robust search of a solution space" [1]. Some of the most well-known metaheuristics include Genetic Algorithm [2], [3], Simulated Annealing [4], Tabu Search [5], [6], Ant Colony [7], [8], and Particle Swarm Optimization [9].

Since the original conceptualization that led to the development of metaheuristics and the inspired deployment of the probabilistic process entailed in the 'survival of the fittest' as proposed by the Darwinian Theory of Evolution, further research has led to analogies with systems in new domains, in all of which the emphasis is on rules by which the state of the domain shifts towards improvement. Among such systems a less explored area is psychology and psychoanalysis [10]. This paper will draw an

* Corresponding author. Tel.: +98 21 8288 3344; Fax: +98 21 8800 5040

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 233–242, 2012.

analogy between concepts in psychoanalytic psychotherapy and optimization problems and will use the analogy as the basis for the introduction of a new search algorithm termed Unconscious Search (US).

The remainder of this paper is organized as follows. Section 2 is devoted to a brief discussion of the Theory of Psychoanalysis. Section 3 contains an analogy between psychoanalysis and optimization. In Section 4 Unconscious Search is proposed based on the analogy in the previous section. In Section 5 one bounded and six unbounded benchmark continuous optimization problems are solved and the results are compared with existing best solutions in literature and a parameter analysis is carried out to observe the behavior of the proposed algorithm. Section 6 states the conclusion.

2 Psychoanalysis

"Psycho-analysis is the name of 1) a procedure for investigating mental processes which are almost inaccessible in any other way, 2) a method (based upon that investigation) for the treatment of neurotic disorders and 3) a collection of psychological information obtained along those lines, which is gradually being accumulated into a new scientific discipline" [11].

In psychoanalysis there are two basic concepts: 'the conscious' and 'the unconscious'. "Consciousness is the subject's immediate apprehension of mental activity" [12]. The unconscious contains basic instincts and repressed impulses [13]. According to Sigmund Freud, the founder of psychoanalysis, conscious thinking is directed at protecting the self, while the unconscious drives the self towards attainment of objects of desire, often despite resistance by the conscious mind. An extreme instance of such resistance can cause a mental disorder.

In psychoanalytic psychotherapy the therapist breaks down any resistance in the conscious mind of the patient against emergence of unconscious impulses. Breakdown of resistance is achieved through Free Association, patient's free verbal responses to words, images and topics evoked by the psychoanalyst. Two forms of resistance have been noted: Displacement and Condensation [14]. Displacement is substitution of a thought expressed verbally with a more 'acceptable' one. Condensation occurs when two or more thoughts are 'condensed' into one symbol in the course of free association.

The steps involved in treatment of patient consist of psychoanalyst's encouraging free association by patient and detecting instances of displacement and condensation resistances, making informed guesses about the contents of the unconscious, feeding these guesses back to the patient and encouraging a fresh course of free association. This cycle continues, every time bringing the therapist closer to uncovering the root cause of the mental disorder which is lodged in the unconscious, till the unconscious is revealed to both therapist and patient. At this point the patient has lost resistance in his consciousness against unconscious impulses and is considered cured.

3 Analogy between Optimization Problems and Psychoanalysis

Let us start with a question. Why do we search for an optimum solution? At first glance the answer seems simple enough: because the optimum solution is the best

possible solution. However, from the point of view, not of optimum solution, but of the search, the answer would be different. An optimum solution seems never readily available. The search must eliminate the occurring non-optimum solutions in order to lead to the optimum solution. As such, it is part and parcel of optimization. Likewise, the unconscious is never readily accessible; there is always a need to surmount the obstacles – the resistance – set in the path of the analyst trying to reach the unconscious. If the complex resistance patterns were non-existent, the psychoanalyst would not need to make a distinction between the conscious and the unconscious.

In psychoanalysis, resistance in the form of displacement or condensation needs to be surmounted by the analyst. Such patterns can be formulated in optimization. For example, in a continuous 'knapsack' problem with n continuous items of various degrees of desirability to be packed into a knapsack of a fixed capacity, the objective in loading up the knapsack with the right amount of every item is to achieve maximum desirability. Optimally, items with a greater desirability to volume ratio tend to take a greater proportion of the total capacity. Just as packing an item with a low desirability to volume ratio can cause a reduction in the amount of a high-ratio item, making the final composition of the knapsack less than optimum, so condensing two thoughts expressed as lingual symbols into an encoded one creates a resistance in the path of the analyst/therapist who is attempting to access the unconscious.

Also in an optimum situation in a continuous knapsack problem, a right proportion exists for each item relative to the total capacity of the knapsack. However, if the solution is not optimal, then for every amount wrongly chosen a corresponding resistance is generated that can be construed as a displacement type of resistance.

We may conceive of two types of adaptive memory, namely 'condensational' and 'displacement' memories, in optimization. These memories check and calculate every instance of resistance during solution-finding. All instances are gradually removed from newly generated solutions by continual comparison of generated solutions against a 'memorized' set of sorted best solutions collected in the 'measurement matrix'. Analyst, too, retains a list of associations considered closest to the contents of the unconscious. Free association is a process similar to construction of new feasible neighbour solutions used to update contents of condensational and displacement memories. Analyst removes instances of resistance by revealing their nature to the patient, after which it is possible to construct the next free association exercise.

Based on the analogy established above between the search space in an optimization problem and the human psyche, the former may reasonably be conceived of as having 'unconscious' optimum or near-optimum solution(s) which need to be searched for, beginning with 'conscious' feasible solution(s). Approaching the contents of the unconscious in the course of the treatment of a mental patient towards a complete cure resembles approaching and finally reaching optimum or near-optimum solution(s) and improving the value of the objective function. The method of free association consists of extracting instances of resistance from among the patient's verbal allusions. This is imitated in optimization in the form of searching the neighbour solutions and evaluating the desirability of a solution with the aid of the condensational and displacement memories. Table 1 lists the features of psychoanalysis and cites the equivalent element of optimization for every feature.

Table 1. Common features of optimization problems and psychoanalysis

Psychoanalysis	Optimization
Mental space, the unconscious, the conscious	Search space
The consciousness, verbal exchange	Feasible solution
The unconscious	Near-optimum and optimum solution(s)
Free association	Construction of neighbour solutions
Resistance	Local optimums and search path
Psychological condition improvement	Objective function value improvement

In designing a search algorithm based on psychoanalytic psychotherapy the steps are matched with their corresponding steps in optimization; in this way a framework for the algorithm is constructed. The steps are:

a) Analyst describes the analysis conditions, asking subject to describe the problem and encouraging free association. Patient generates a set of speech components containing information. The speech components resemble the feasible solutions in optimization that make up a set.
b) Analyst provides subject with a starting point based on what is speculated to have been corrupted by the existence of built-in resistances. Free association hovers around this starting point. In optimization the choice of starting point is likewise made through condensational and displacement memories and is based on the behavior of resistance observed within the search space.
c) Analyst's focus is on the metaphorical character of speech produced by subject during free association. Analyst identifies and evaluates instances of resistance. Similarly, local search yields values for displacement and condensational memories and leads to an updated set of feasible solutions. The solutions are used to update the values in the two memories through a determination of the values of the resistances.
d) Analyst again provides subject with a point to begin to freely associate from a point evaluated as closer to the unconscious. In optimization, updating memories leads to construction of a new starting point for a local search. The process described above is repeated till a good solution is reached and the termination criteria is met.

4 Unconscious Search

Considering the analogy between psychoanalysis and optimization in Section 3, we propose Unconscious Search (US). An optimization problem can be represented in the following form:

$$(Q) \quad Minimize\ C(x):\ g(x) \le b, h(x) = d, x \in X\ in\ \mathcal{R}_n$$

The objective function $C(x)$ may be linear or nonlinear. Functions $g(x)$ and $h(x)$ are constraints of vector x where x is the set of decision variables, and condition $x \in X\ in\ \mathcal{R}_n$ restricts components of x to continuous ranges of values. For solving optimization problem (Q) we follow the same steps mentioned in Section 3 above.

Initially, a set of feasible solutions $\boldsymbol{P} = \left(P_1, P_2, \dots, P_{|MM|}\right)$ is generated. $|MM|$ is the size of the measurement matrix (MM) in which the sorted set of the best feasible

solutions, i.e. those nearest to the optimum solution that are visited during search process, are held. MM can be defined as follows:

$$MM = \{P_q \mid C(P_q) < C(P_{q+1}), q = 1,2,\dots,|MM|\} \tag{1}$$

The solutions kept in MM are used to measure the resistance, and are ranked by means of a 'translation function' according to the value of their resistance. The translation function maps the value of the objective function of any solution P_q (i.e. a solution that belongs to MM) into a range $(\alpha, 1-\alpha)$ for $\alpha \in (0,1)$ where $P_q \in MM$.

Any solution that does not belong to MM and for which the objective function is greater than the worst solution within MM, is assigned a scalar penalty value $h \in \mathcal{R}^+$.

The translation function f_{t_I} is defined as follows:

$$f_{t_I}\left(C(P_q)\right) = \frac{1}{1 + e^{a\left(C(P_q)\right)+b}}, \qquad P_q \in MM. \tag{2}$$

In (2) above, f_{t_I} is a sigmoid function and is used to calculate the proximity of solutions in MM to the optimum solution. a and b are the parameters of f_{t_I} and are calculated in every iteration throughout the search.

The best member P_{best} of measurement matrix is assigned the value $1-\alpha$, the worst member P_{worst} the value α, by the translation function. For any solution lying outside MM for which objective function is greater than the worst solution in MM, there is a penalty value h assigned to that solution. Evaluating resistance level in solutions is performed by translation function f_{t_I} and by displacement and condensational memories. f_{t_I} measures quality of solutions while displacement and condensational memories memorize the resistance patterns in the solutions. Displacement memory, Π, memorizes the displacement pattern of resistance in the solutions, i.e., dividing the possible range (considering that $x \in X$) of every solution component into $|X|$ equal parts, it assigns the output of $f_{t_I}(C(P_q))$ to the corresponding part. In other words, Π determines how much resistance will occur if a specified range of X is assigned to solution component x.

Π is defined as follows:

$$\Pi = \{(\Pi_{j_I}, \Pi_{j_E}) \mid j = 1,2,\dots,|X|\}. \tag{3}$$

in which,

$$\Pi_{j_I} = \{\pi_{j_{Ii}} \mid i = 1,2,\dots,n, \qquad j = 1,2,\dots,|X|\}, \tag{4}$$

$$\Pi_{j_E} = \{\pi_{j_{Ei}} \mid i = 1,2,\dots,n, \qquad j = 1,2,\dots,|X|\}. \tag{5}$$

and n is the number of decision variables. $\pi_{j_{Ii}}$ and $\pi_{j_{Ei}}$ are defined as follows:

$$\pi_{j_{Ii}} = \left(\sum_{\text{MS}} f_{t_I}\left(C(P_q)\right), P_q \in MM, P_q(i) \in X_j, j = 1,2,\dots,|X|, q = 1,2,\dots,|MM|, i = 1,2,\dots,n\right) \tag{6}$$

$$\pi_{j_{Ei}} = (\textstyle\sum_{MS} h, \text{ for solutions with an objective function greater than the worst solution in } MM, j = 1,2,\dots,|X|, i = 1,2,\dots,n) \tag{7}$$

in which $MS \in \mathbb{Z}^+ - \{0\}$ is the Memory Size that shows the MS last performed iterations of algorithm that are memorized. X_j represents the jth subinterval of X. P_{worst} is worst solution in MM, $P_q(i)$ is value of ith decision variable in solution P_q.

By means of the displacement memory Π, a new solution can be constructed. This solution is denoted by S_1. The ith solution component $S_1(i)$ will be assigned to one of the possible ranges X_j in solution space with a probability defined as follows:

$$Prob\{S_1(i) \in X_j\} = \frac{\frac{\pi_{j_{Ii}}}{1+\left(\pi_{j_{Ei}}\right)^{\beta}}}{\sum_{j=1}^{|X|}\frac{\pi_{j_{Ii}}}{1+\left(\pi_{j_{Ei}}\right)^{\beta}}} \tag{8}$$

in which $Prob$ is the probability function and β is a predefined constant. When the solution component $S_1(i)$ is assigned to X_j, it will choose a number in X_j at random. The larger the value of $\pi_{j_{Ii}}$, the higher the probability of $S_1(i) \in X_j$; the larger the value of $\pi_{j_{Ei}}$, the less the probability of $S_1(i) \in X_j$.

Once a displacement-free solution (DFS) has been reached, condensational memory Π' is used to eliminate the condensational resistance pattern. Displacement memory Π is used to construct a new DFS, while condensational memory Π' is used to improve the solution constructed with the help of Π, making it a condensation-free solution (CFS).

Condensational memory Π' is defined as follows:

$$\Pi' = \{(\Pi_i^+, \Pi_i^-)|i = 1,2,\dots,n\}, \tag{9}$$

in which,

$$\Pi_i^+ = \{(\pi_{iI}^+, \pi_{iE}^+)^T|i = 1,2,\dots,n\} \tag{10}$$

$$\Pi_i^- = \{(\pi_{iI}^-, \pi_{iE}^-)^T|i = 1,2,\dots,n\}, \tag{11}$$

where,

$$\pi_{iI}^+ = (\textstyle\sum_{MS} f_{t_I}\left(C\left(P_q\right)\right), q = 1,2,\dots,|MM|, i = 1,2,\dots,n\,|P_q(i) \text{ is increased with respect to its previous value in first iteration of local search)} \tag{12}$$

$$\pi_{iE}^+ = (\textstyle\sum_{MS} h, \text{ for solutions with an objective function greater than worst solution in } MM, i = 1,2,\dots,n|i\text{th decision variable is increased with respect to its previous value in the first iteration of local search)} \tag{13}$$

$$\pi_{iI}^- = (\textstyle\sum_{MS} f_{t_I}\left(C\left(P_q\right)\right), q = 1,2,\dots,|MM|, i = 1,2,\dots,n\,|P_q(i) \text{ is decreased with respect to its previous value in first iteration of local search)} \tag{14}$$

$$\pi_{iE}^- = (\textstyle\sum_{MS} h, \text{ for solutions with an objective function greater than the worst solution in } MM, i = 1,2,\dots,n|i\text{th decision variable is decreased with respect to its previous value in the first iteration of local search)} \tag{15}$$

Note that, since in the beginning of the first iteration of US, we do not perform a local search, we have no information with which to update Π'; thus equations 9~15 are applied from the second iteration onwards.

Once S_1 is constructed, Π' determines whether $S_1(i)$ is to be decreased or increased by calculating two values $v_i^- = \frac{\pi_{iI}^-}{1+\pi_{iE}^-}$ and $v_i^+ = \frac{\pi_{iI}^+}{1+\pi_{iE}^+}$ and generating a

random number ψ in the range $(0, v_i^+ + v_i^-)$. If $\psi \leq v_i^+$, the value of $S_1(i)$ will be increased by as much as a predefined number $\delta \in \mathcal{R}^+ \cup \{0\}$; otherwise, the value of $S_1(i)$ will be decreased by the same amount δ. Decreasing or increasing the value of $S_1(i)$ will be repeated until limits of $S_1(i)$ are reached with the solution still remaining feasible.

Having constructed S_1, the first solution in an iteration known as the 'mother solution', by using memory Π', solutions S_2, S_3, … are generated from S_1. Solution S_1 is DFS, while solutions S_2, S_3, … are CFSs derived from mother solution S_1. The best solution among S_1,S_2, S_3, …, denoted by S^*, is the starting point in the local search.

Memories Π and Π' help to appoint the region where the mother solution should be located and the direction along which the mother solution is to be moved by increments of δ in order for the solutions S_2, S_3, … to be generated.

After obtaining the solution S^*, a local search is conducted with S^* as the starting point. If the result of the search is S', $C(S') \leq C(S^*)$. In the process of reaching S', more resistance patterns are revealed and Π and Π' are updated using f_{t_l}. Notice that MM is updated only if objective function value of S' is better than objective function value of P_{worst}, in which case MM is updated so that the following inequality holds:

$$C(P_u) < C(S') < C(P_{u+1}) \tag{16}$$

in which P_u and P_{u+1} are the members of MM before S' is augmented. In order for above inequality to always hold, MM should remain sorted through every update. If MM is changed, f_{t_l} must be corrected to match the new MM, i.e. the coefficients a and b must be adjusted. Denoting the new coefficients by $a'$and b', we have:

$$a' = \frac{2\left(ln\left(\frac{1-\alpha}{\alpha}\right)\right)}{C(P_{worst})-C(P_{best})} \text{ and } b' = \left(\frac{C(P_{best})+C(P_{worst})}{C(P_{worst})-C(P_{best})}\right)\left(ln\left(\frac{\alpha}{1-\alpha}\right)\right) \tag{17-18}$$

US is a multi-start metaheuristic which contains three main phases: construction, construction review, and local search. The first phase consists in constructing a displacement-free, or 'mother', solution. The second phase is that of constructing condensation-free solutions derived from the mother solution in Phase 1. The third phase corresponds to recognition of resistance patterns by exploring the search space.

5 Performance Evaluation and Parameter Analysis

To demonstrate the performance of the Unconscious Search algorithm, a benchmark bounded and six benchmark unbounded continuous engineering optimization problems have been solved and the results tabulated against solutions reached using established heuristic and metaheuristic methods. The computer on which the tests were run was a 1.5 GHz Intel Centrino.

5.1 Pressure Vessel Design

In this problem there are four decision variables: x_1 (=T_s, shell thickness), x_2 (=T_h, spherical head thickness), x_3 (=R, radius of cylindrical shell) and x_4 (=L, shell

length). The thicknesses are integer multiples of 0.0625 in. and R and L have continuous values within the constraints $40 \leq R \leq 80in.$ and $20 \leq L \leq 60in.$ The mathematical expression is as follows:

$$f(x_1, x_2, x_3, x_4) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1611x_1^2x_4 + 19.84x_1^2x_3$$

$$subject\ to,$$

$$g_1(x_1, x_2, x_3, x_4) = 0.0193x_3 - x_1 \leq 0, \quad g_2(x_1, x_2, x_3, x_4) = 0.00954x_3 - x_2 \leq 0,$$

$$g_3(x_1, x_2, x_3, x_4) = 750.0 \times 1728.0 - \pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 \leq 0, \ g_4(x_1, x_2, x_3, x_4) = x_4 - 240.0 \leq 0, \ g_5(x_1, x_2, x_3, x_4) = 1.1 - x_1 \leq 0, \ g_6(x_1, x_2, x_3, x_4) = 0.6 - x_2 \leq 0.$$

The results in Table 2 below are reproduced from the literature [15] on harmony search and also from test runs made using the US algorithm. The US results, with their run time of approximately 2 seconds, are better than the best found solution.

Table 2. Numerical results for pressure vessel design

Variables Optimal Value	Sandgren	Wu and Chow	HS	US
$T_s(x_1)$	1.125	1.125	1.125	1.125
$T_h(x_2)$	0.625	0.625	0.625	0.625
$R\ (x_3)$	48.97	58.1978	58.2789	58.2900
$L\ (x_4)$	106.72	44.2930	43.7549	43.6934
Cost ($)	7980.894	7207.494	7198.433	7197.737

The number of iterations needed for the US algorithm to lead to the optimum solution in the pressure vessel design problem is 360 in the best case, with the value of 7197.737 for the objective function. To analyze effect of parameter α we consider the average number of necessary iterations in 100 runs in which $MS = 75$ and $|MM| = 30$ for different values of α, given that objective function is 7200.00.

As α increases, the average number of iterations tends to decrease for values of $\alpha \leq 0.3$. The reason is that US behaves more prohibitively towards inferior solutions as regards their inclusion in MM with increase in value of α. For values of $\alpha > 0.3$, solutions entered in MM are so close to each other in terms of their corresponding objective function values as to reduce ability of algorithm to escape local optima; this weakens diversification of algorithm causing average number of iterations to rise.

To appraise the effect of $|MM|$ and MS on convergence in pressure vessel design problem, value of α was kept at 0.3 and the two parameters were given 10 different values (to produce 100 combinations). US was run 100 times for each combination and average number of iterations for each combination calculated and taken as the response variable. Using Minitab software, a two-way variance analysis with alpha level 0.05 was carried out. The analysis showed no meaningful change in response variable (average number of iterations) as objective function tended to 7200.00.

It may be inferred from the results of the analysis that for the pressure vessel design problem values of $|MM|$ and MS do not considerably affect convergence of US algorithm. This low sensitivity to changes in parameters $|MM|$ and MS constitutes an important advantage of US algorithm in the adjustment of its parameters.

5.2 Other Benchmark Problems

To further test effectiveness and robustness of US algorithm, six unbounded functions were solved for global minimum in each instance. The functions, also tested by harmony search [15], are as follows:

- Rosenbrock function: $f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$
- Goldstein and cost function 1: $f(x_1, x_2) = \{1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\} \times \{30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\}$
- Goldstein and cost function 2: $f(x_1, x_2) = e^{\left\{\frac{1}{2}(x_1^2 + x_2^2 - 25)^2\right\}} + sin^4(4x_1 - 3x_2) + \frac{1}{2}(2x_1 + x_2 - 10)^2$
- Eason and Fenton's gear train inertia function: $f(x_1, x_2) = \left(\frac{1}{10}\right)\left\{12 + x_1^2 + \frac{1+x_2^2}{x_1^2} + \frac{x_1^2x_2^2+100}{x_1^4x_2^4}\right\}$
- Wood function: $f(x_1, x_2, x_3, x_4) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 + 90(x_4 - x_3^2)^2 + (1 - x_3)^2 + 10.1[(x_2 - 1)^2 + (x_4 - 1)^2] + 19.8(x_2 - 1)(x_4 - 1)$
- Powell quartic function: $f(x_1, x_2, x_3, x_4) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$

The results of parameter analysis in these problems led us to consider the following values for the algorithm parameters: $|MM| = 10$, $MS = 75$ and $\alpha = 0.3$. A similar variance analysis for appraisal of sensitivity produced similar results to those obtained in the pressure vessel design problem, with the conclusion that US algorithm has the advantage in adjustment of its parameters. Results obtained by US (in Table 3) point to global optimum in each instance, while running time never exceeds one second.

Table 3. Numerical results in unbounded function minimization

Function	Optimum value	HS	US
Rosenbrock	$f(1,1) = 0$	$5.6843418860E-10$	$0.1E-15$
Goldstein I	$f(0,-1) = 3$	$0.3000000000E+01$	$3-(0.955E-13)$
Goldstein II	$f(3,4) = 1$	$0.1000000000E+01$	$1-(0.289E-13)$
Easton and Fenton	$f(1.7435,2.0297) = 1.74$	1.74415	1.744152
Wood	$f(1,1,1,1) = 0$	$4.8515E-09$	$0.14959E-9$
Powell quartic	$f(0,0,0,0) = 0$	$0.1254032468E-11$	$0.156201E-8$

6 Conclusion

In this paper a new search algorithm, Unconscious Search (US), was proposed based on analogy between optimization and human psyche as it is described in Freud's Theory of Psychoanalysis. Concepts of 'free association' and 'resistance analysis' in psychoanalysis were simulated and incorporated in US algorithm as three main phases of Construction, Construction review and Local search.

In construction phase, using the displacement memory, displacement patterns of resistance are analyzed to construct a displacement-free solution (DFS). In construction review phase, considering DFS and condensational memory we analyze condensation patterns of resistance and construct a set of condensation-free solutions (CFS). Finally, in local search, simulating free association, we improve the best solution found in the construction and the construction review phases.

The results of the benchmark problems and parameter analysis demonstrate that Unconscious Search is robust, a validated optimization method for solving continuous hard problems, and easy to use. The proposed Unconscious Search algorithm has been coded by using C++.The continuous benchmark problems attempted were Pressure vessel design which is bounded in character and several unbounded benchmarks. The solution obtained for the pressure vessel design was ever best one.

References

1. Glover, F., Kochenberger, G.A.: Handbook of Metaheuristics. Kluwer Academic Publishers, USA (2003)
2. Holland, J.H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975)
3. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley, MA (1989)
4. Kirkpatrick, S., Gelatt, C., Vecchi, M.: Optimization by simulated annealing. Science 220, 671–680 (1983)
5. Glover, F.: Tabu Search - Part I. ORSA Journal on Computing I(3) (1989)
6. Glover, F.: Tabu Search - Part II. ORSA Journal on Computing II(3) (1989)
7. Dorigo, M.: Optimization, learning and natural algorithms. Dipartimento di Elettronica, Politecnico di Milano, Milan (1992) (in Italian)
8. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B 26(1), 29–41 (1996)
9. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, pp. 1942–1948. IEEE Service Center, NJ (1995)
10. Glover, F.: Tabu Search-Uncharted Domains. Annals of Operations Research 149, 89–98 (2007)
11. Mijolla, A.: International Dictionary of Psychoanalysis. In: Mijolla, A. (ed. in chief), pp. 1362–1366. Thomson Gale, USA (2005)
12. Cahn, R.: International Dictionary of Psychoanalysis. In: Mijolla, A. (ed. in chief), pp. 333–334. Thomson Gale, USA (2005)
13. Assoun, P.L.: Le Vocabulaire de freud. Ellipses, France (2002)
14. Freud, S.: The Interpretation of Dreams. In: Strachey, J. (ed.), 3rd (Revised) English edn., New York (2010)
15. Lee, K.S., Geem, Z.W.: A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. Computers Methods in Applied Mechanics and Engineering 194, 3902–3933 (2005)

Brain Storm Optimization Algorithm with Modified Step-Size and Individual Generation*

Dadian Zhou[1], Yuhui Shi[1], and Shi Cheng[1,2]

[1] Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, Suzhou, China
yuhui.shi@xjtlu.edu.cn
[2] Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool, UK

Abstract. Brain Storm Optimization algorithm is inspired from the humans' brainstorming process. It simulates the problem-solving process of a group of people. In this paper, the original BSO algorithm is modified by amending the original BSO. First the step-size is adapted according to the dynamic range of individuals on each dimension. Second, the new individuals are generated in a batch-mode and then selected into the next generation. Experiments are conducted to demonstrate the performance of the modified BSO by testing on ten benchmark functions. The experimental results show that the modified BSO algorithm performs better than the original BSO.

Keywords: Brain Storm Optimization, Adaptive step-size, Selection.

1 Introduction

Brain Storm Optimization algorithm (BSO) is one type of swarm intelligence algorithms introduced by Shi in 2011 [3]. Similar to many other swarm intelligence algorithms, such as particle swarm optimization (PSO) [5], ant colony optimization (ACO) [2], artificial immune system [1], firefly optimization algorithm [7], etc., BSO is a population-based algorithm.

BSO algorithm simulates the human brainstorming process in which there are, generally speaking, three roles i.e.a facilitator, a group of people, and problem owners. The facilitator should facilitate the brainstorming process to help the brainstorming people to come up as many diverse ideas as possible, rather than generate ideas by himself/herself [6]. The brainstorming group of people should better have as diversified backgrounds and knowledge as possible. Under this circumstance, the ideas coming from these people will be diverse and affluent. After one round of an idea generation process, the problem owners will pick up some better ideas from all generated ideas for solving the problem in this round.

* The authors' work is partially supported by National Natural Science Foundation of China under grant No.60975080, and Suzhou Science and Technology Project under Grant No.SYJG0919.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 243–252, 2012.

For a picked-up idea, it will then have a higher probability to be treated as a clue to further generate more ideas in the next round of the idea generation process. This process is repeated for many times until a sufficiently good idea emerges or time runs out [3]. The above repeated brainstorming processes serve as a foundation for the BSO algorithm.

In a BSO algorithm, there are four general steps, which are initialization, clustering, generation, and selection. The last three steps are repeated iteration over iteration until a terminal condition is met.

The rest of the paper is organized as follow. Section 2 introduces the original BSO algorithm and the modified BSO algorithm. Ten benchmark functions are utilized for testing the modified BSO algorithms and comparing it with the original BSO in Section 3. Finally, Section 4 explains the reasons for modifications of the original version of BSO algorithm and analyzes the experimental results, followed by conclusions in Section 5.

2 BSO Algorithms

2.1 Original Version of BSO Algorithm

The general procedure of BSO algorithm is given in [4]. Based on the original BSO algorithm published in the paper, new individuals are generated by adding Gaussian noise. The formulas for generating new individuals are given as follows,

$$X_{new}^{d} = X_{RP}^{d} + \xi \times N(\mu, \sigma) \tag{1}$$

$$\xi = \log\mathrm{sig}\left(\frac{0.5 \times max\_iteration - current\_iteration}{k}\right) \times \texttt{rand}() \tag{2}$$

where X_{RP}^{d} is the dth dimension of the chosen reference point, which is used for creating a new individual; X_{new}^{d} is the dth dimension of a new individual; $N(\mu, \sigma)$ denotes Gaussian random function with mean μ and standard derivation σ; ξ is a coefficient called step-size, affecting the contribution of the Gaussian noise, and is updated according to equation (2); $\log\mathrm{sig}$ is a logarithmic sigmoid transfer function; $max\_iteration$ is the maximum iteration number for one execution of BSO program; $max\_iteration$ is the number of current iteration in a BSO program; k is a coefficient and `rand`() denotes a random variable with a uniform distribution between 0 and 1.

2.2 New Version of BSO Algorithm

Three parts of the modified BSO algorithm are different from the original one. Firstly, the step-size for the new version is adaptive and is updated concurrently according to the current range of all individuals on each dimension. Secondly, the method of new individuals' generation is modified. The program creates more individuals to fully exploit each reference point rather than creates the same number of individuals as the population size. Finally, a selecting strategy

is utilized to form the new population of individuals in the next generation. The procedure of the modified BSO is given in Table 1 below.

In the modified BSO, the generation step and selection step is modified. The step-size, which weights the influence of Gaussian noise,is different in the modified BSO from that in the original BSO. The new step-size is updated as follows.

$$\xi_i^{center} = k_1 \times (x_{n\_\max,i} - x_{n\_\min,i}) \tag{3}$$

$$\xi_i^{individual} = k_2 \times (x_{n\_\max,i} - x_{n\_\min,i}) \tag{4}$$

where ξ_i^{center} and $\xi_i^{individual}$ are the values of step-size in ith dimension, and ξ_i^{center} is used in step 3.1 and 3.2 while $\xi_i^{individual}$ is used in step 3.3; k_1 and

Table 1. Procedure of new version of BSO algorithm

1. Initialization:
   1.1. Randomly generate ***n*** individuals in a specified range.
2. Clustering:
   2.1. Cluster the ***n*** individuals into ***m*** clusters in accordance with k-means clustering method, and find the best individual based on their fitness values in each cluster as cluster center.
   2.2. Randomly generate a number between 0 and 1. If the number is smaller than ***P1***, go to 2.3. Otherwise, go to 3.
   2.3. Randomly choose a cluster center, then randomly generate an individual in the specified range, and replace the chosen cluster center by the new individual.
3. New individual generation: (Different from original version)
   3.1. Randomly choose a cluster center. The probability for every cluster center to be chosen is relevant with its cluster size. Based on the chosen cluster center, generate a new individual through adding Gaussian noise. Repeat 3.1 for n times (where n denotes the population size) and therefore generate n individuals.
   3.2. Randomly choose two cluster centers and merge the two centers with a random weight to be a new center. Use the new center to generate a new individual through adding Gaussian noise. Repeat 3.2 for n times and generate n individuals.
   3.3. Randomly choose an individual as a reference point. Add Gaussian noise to generate a new individual. Repeat 3.3 for n times and generate n individuals.
4. Selection: (Different from original version)
   4.1. Evaluate the total 3n newly generated individuals.
   4.2. Randomly sort totally 4n individuals into n groups with equal number of 4 individuals. In each group, select the individual with the best fitness value among the four and copy it into the next generation.
5. Iteration:
   5.1. Go to 2 until the terminal criteria are met (e.g. iterate for ***Max_iteration*** times in one execution of BSO program).
6. End of BSO and Output the best fitness value.

k_2 are two coefficients; $x_{n\_\max,i}$ and $x_{n\_\min,i}$ are maximum value and minimum value among the entire population in ith dimension, respectively.

Therefore, according to equation (3), (4), the values of step-size are adaptive according to the dynamic range of the population of individuals. If all individuals in the population are dispersively distributed, the value of step-size will be relatively large. If all individuals in the population are aggregated, the value of step-size will be relatively small. The step-size is updated automatically in every generation.

3 Experimental Results

3.1 Benchmark Functions

Ten benchmark functions are used for testing the performance of BSO algorithms. All ten benchmark functions and their dynamic ranges are from [8]. Among the ten functions, function f_1, f_2, f_3, f_4, and f_5 are uni-modal functions, while function f_6, f_7, f_8, f_9, and f_{10} are multi-modal functions. Each benchmark function has its unique minimum value. The performance of two BSO algorithms will be compared by using these benchmark functions.

3.2 Results

A better BSO algorithm should have a capacity of finding optimum values with little number of generations compared with its 'rivals'. The experimental results of two versions of BSO algorithms will be provided and discussed in this section. Parameters used for the original BSO is the same as that in [3], whereas parameters for the modified BSO are listed in Table 2.

Table 2. Parameters of the modified BSO

n	**m**	P_1	k_1	k_2	*max_iteration*	μ	σ
50	5	0.4	0.15	0.13	1000	0	1

Where **n** is the population size; **m** is the number of clusters; μ and σ are two parameters used in Equation (1); P_1 is the constant used in Table 1. k_1 and k_2 are two coefficients in Equations (3), (4). *max_iteration* denotes the maximal number of iterations for one execution of BSO algorithm.

The purpose of the experiment is to test the performance of the modified BSO algorithm, and compare it with the original BSO with regards to the ten benchmark functions, each of which has dimension set to be 10, 20, and 30,respectively. For both BSO algorithms, they will be run 50 times for each benchmark function with each different dimension.Maximum generation number is 1000 for each run. Minimum, median, and maximum function values in the 50 runs for each function are recorded in addition to variance. Tables 3 and 4 show the experimental results.

Table 3. Results of uni-modal functions

Uni-modal Function	Version	Dimension	Best	Median	Worst	Variance
Sphere	Original	10	7.58E-23	2.29E-22	5.37E-22	5.65E-45
		20	7.95E-22	1.48E-21	2.83E-21	1.54E-43
		30	3.53E-21	5.29E-21	1.88E-20	6.25E-42
f_1	New	10	2.8E-139	1.52E-68	8.33E-36	1.52E-72
		20	1.15E-56	2.42E-36	4.01E-19	3.15E-39
		30	1.75E-27	3.25E-20	4.06E-10	3.22E-21
Schwefel's P221	Original	10	5.51E-12	8.67E-12	1.29E-11	2.7E-24
		20	4.21E-11	2.74E-06	0.001695	5.99E-08
		30	0.008452	0.045602	5.677727	1.368525
f_2	New	10	3.03E-36	2.2E-20	1.72E-11	5.82E-24
		20	3.26E-05	0.070197	1.621416	0.11648
		30	0.160679	4.096068	13.11929	9.345655
Step	Original	10	0	0	0	0
		20	0	0	0	0
		30	0	0	0	0
f_3	New	10	0	0	0	0
		20	0	0	0	0
		30	0	0	0	0
Schwefel's P222	Original	10	2.37E-11	3.63E-11	5.43E-11	4.13E-23
		20	1.07E-10	1.41E-10	3.28E-05	2.11E-11
		30	5.81E-06	0.0056	0.086858	0.000359
f_4	New	10	9.42E-47	2.39E-30	3.08E-17	1.85E-35
		20	4.36E-40	9.32E-21	3.84E-14	3.09E-29
		30	4.97E-22	1.11E-13	4.78E-09	4.58E-19
Quartic Noise	Original	10	0.000525	0.022042	0.077237	0.000314
		20	0.003682	0.025093	0.105127	0.000356
		30	0.013142	0.043104	0.134042	0.000856
f_5	New	10	0.009694	0.03714	0.076704	0.000236
		20	0.022794	0.051455	0.141916	0.000747
		30	0.042362	0.084453	0.146334	0.000684

The experimental results for uni-modal functions are given in Table 3. From the results, it can be observed that the modified BSO algorithm performs better for f_4. Smaller values can be obtained under all 10, 20 and 30 dimensions. The results for f_3 and f_5 for both versions of BSO are comparable. However, for f_1 and f_2, new version performs better in low dimensions rather than in high dimensions. The fitness values, obtained by new BSO, for ten and twenty dimensional f_1 and for ten dimensional f_2 are much smaller than the values obtained by the original BSO.

In Table 4, the experimental results are for multi-modal functions. In this case, the superiority of new version of BSO algorithm is apparent. For f_6C f_{10} in all dimensions, the best fitness values, obtained by the modified BSO, are much better than the values obtained by the original BSO. In addition, the results of the new algorithm, including the best value, median, worst value, average, and

Table 4. Results of multi-modal functions

Uni-modal Function	Version	Dimension	Best	Median	Worst	Variance
Ackley	Original	10	1.14E-11	1.8E-11	2.41E-11	9.77E-24
		20	2.33E-11	3.54E-11	4.47E-11	1.91E-23
		30	3.83E-11	5.49E-11	1.41E-10	2.35E-22
f_6	New	10	8.88E-16	4.44E-15	4.44E-15	2.02E-30
		20	4.44E-15	4.44E-15	6.48E-14	7.68E-29
		30	2.14E-13	2.1E-07	2.118958	0.36697
Rastrigin	Original	10	0.994959	4.974795	9.949586	3.617645
		20	9.949586	19.89917	36.81344	22.33305
		30	26.86387	36.81346	69.64694	81.57233
f_7	New	10	0	2.984877	4.974795	1.475016
		20	4.974795	8.954632	16.91429	8.984319
		30	6.964713	19.40169	29.84875	25.5901
Rosenbrock	Original	10	2.420121	6.697281	221.531	907.624
		20	16.59699	18.06682	103.1998	735.3692
		30	26.1446	28.46412	903.5095	16515.81
f_8	New	10	1.184153	4.955994	8.191643	1.381576
		20	3.074479	16.40247	155.6893	1003.395
		30	8.737314	79.83059	477.6855	6378.183
Schwefel's P226	Original	10	1026.504	1608.895	2428.527	114655.8
		20	2210.907	3583.748	4603.574	253207.3
		30	3752.177	5567.439	7051.786	628155.3
f_9	New	10	0.000127	473.7535	952.058	38179.77
		20	473.7536	1176.791	2118.222	145664.8
		30	1304.339	2037.75	3702.03	214829.6
Griewank	Original	10	0.598233	2.048943	4.069798	0.54909
		20	0	0.022141	1.028908	0.025973
		30	9.44E-05	1.266122	4.229146	1.047128
f_{10}	New	10	0	0.012321	0.04433	0.000142
		20	0	0	0.046483	0.000108
		30	0	0.013547	0.210806	0.00189

variance, are smaller for f_7, f_9 and f_{10} in all dimensions. However, only for f_6 and f_8 in high dimensions, the medians and worst values obtained by the original BSO are better than that by the modified one.

According to the experimental results in the above tables, there are three main observations. The first one is that the new algorithm performs much better for multi-modal functions compared with the original version. Testing functions with many local minima are real challenges for optimization algorithms, because algorithms are easy to trap into poor local minima for multi-modal functions. Therefore, a good algorithm should have an ability of fleeing poor local minima and finding near-global minima [8]. Secondly, the best values obtained by the new version in 50 runs are better than that by the original one for almost all testing functions, except for f_2 and f_5 with regards to some dimensions. It can be concluded that the modified BSO has an ability to find much smaller function

values. However, the experimental results for the modified BSO from Table 3 and 4 are not consistent. This phenomenon mainly occurs for uni-modal functions, for example, f_1 and f_2 as well as f_6 with respect to high dimensions. This is the third feature for the modified BSO algorithm.

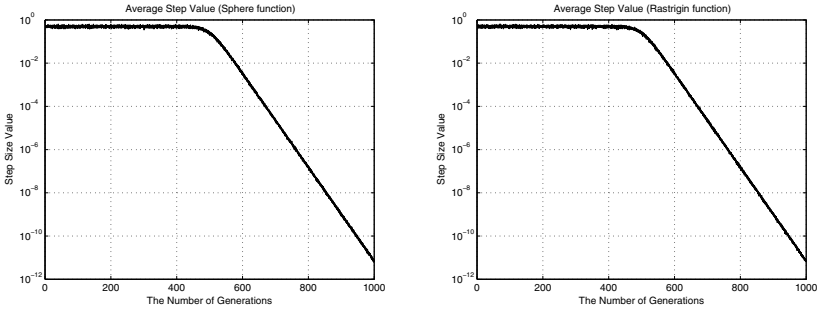
4 Analyses and Discussions

In the modified BSO algorithm, the step-size is updated according to the dynamic range of the individuals, and generation and selection strategy are also different. The modifications give the new version of BSO algorithm a capacity of escaping from poor local minima and fast convergence on one point. The step size in the new version is adaptive to facilitate the convergence of the algorithm, while generation and selection strategy keeps the diversity for the whole population. The functionalities of new step-size and new individuals' generation and selection are analyzed as follows.

4.1 Original Step Size

From Equation (2), the step-size is independent from the dynamic range of population. It only relates to the number of generations. With the increment of iterations, the step-size tends to decrease. Due to the random variable in Equation (2), the step-size values for individuals are different. Therefore, we use an average value to exhibit the trend of step-size. The following figure shows the average values of step-size in a logarithmic scale against the number of generations for two testing functions, which are ten-dimensional sphere function f_1 and ten-dimensional Rastrigin function f_7, respectively.

According to Figure 1, the step-size trends for two functions are almost the same. The step-size is not adaptive with the dynamic range of population. If the number of generations is large, the step-size will be quite small. The new individuals created from previous generation will be adjacent to their 'parents', due to a small step-size. Hence, it is difficult for the algorithm to escape from local minima in this case, especially at the late stage of the algorithm's running.



(a) (b)

Fig. 1. 10 dimensional function: (a) sphere, (b) rastrigin

4.2 Modified Step-Size

Based on the Equation (3) and (4), the step-size adjusts according to the dynamic range of values in each dimension. At the beginning of a program, individuals on each dimension are randomly assigned some values in the specified range, for instance, $[-100, 100]$ for f_1. Therefore, the values of step-size on each dimension are quite large initially. After several generations, the population will aggregate around the optimal value or local minimum values. The range of all individuals on each dimension will be much smaller than the initially specified range. Hence, the step-size will become increasingly smaller with the increasing number of iterations. However, there are some mutations occurring during the process of new individuals' generation and selection. These mutations have significant impact on the step-size. If some mutations occur, the values of step-size, as well as the dynamic range of individuals, may become large again. This gives the algorithm a chance to jump out of local minima.

Two benchmark functions, Sphere function f_1 and Rastrigin function f_7, are used for illustration in this section. The values of step-size in logarithm against the number of generations are displayed in the following figure for the two functions with dimension 10.

It can be observed from the Figure 2 that the magnitude of step-size reduces dramatically. However, it will increase abruptly due to some mutations. Based on Equation (3) and (4), the step-size is used as a coefficient to multiply a Gaussian noise. When the step-size is small, it means the population is converging and each individual in the program has a low 'speed' to move. It is impossible for individuals to escape a local minimum in this case. Conversely, when the step-size becomes large, the speed of moving for each individual is fast, and escaping becomes possible. Hence, the step-size in the modified BSO algorithm is adaptive, and can avoid individuals being trapped into local minima.

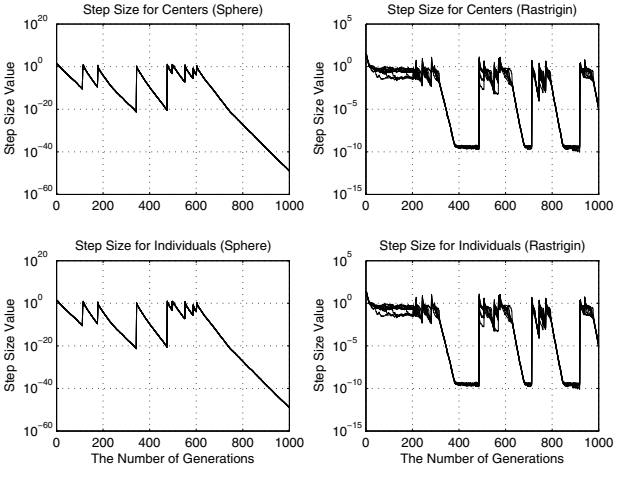


Fig. 2. 10 dimensional sphere function (f_1)

4.3 New Individuals' Generation and Selection

The new step-size can adjust according to the dynamic range of population. However, in some cases, the population converges too fast through using the step-size. This will result in the disappearance of population diversity. Based on the experimental results of f_2 and f_3 on high dimensions, the algorithm performs poorly if only the step-size is modified. For example, the best, the median and the worst values for 30-dimension f_2 are 5.79, 11.07 and 18.24, and for 30-dimension f_3 are 0, 2 and 34. The results are much worse than the results in Table 3. Therefore, a new generating method and a selecting strategy are utilized in the modified BSO algorithm.

In the new version, 'parents', which are individual in the current generation, will generate their offspring, which form the next generation. Some of the individuals are created by adding Gaussian noises to cluster-centers. Others are generated by adding Gaussian noises to arbitrary individuals from current generation. Part of the 'parents' and their 'offspring' will be selected into the next generation in accordance with a selecting strategy. In the selecting process, all individuals ('parents' and 'offspring') are divided into n groups, where n is the population size. The individual with the best fitness value in each group is selected as the one into the next generation. The new version does not directly select n good individuals from all 'parents' and 'offspring'. The purpose of this selecting strategy is to keep the diversity of the population, and to promote influence of mutations.

5 Conclusions

A modified BSO algorithm was introduced in this paper. In the modified BSO, an adaptive step-size is utilized to gain the ability of escaping local minima for the algorithm. The generation and selection strategy in the new version is also different from the original one. The selection strategy will maintain the diversity of the population and promote the influence of mutations in the algorithm. Therefore, the modified BSO algorithm is suitable to deal with benchmark functions with many local minima. For uni-modal functions, new version performs better for functions with low dimension, for example, ten-dimensional function or twenty-dimensional function. These conclusions are illustrated by experimental results in this paper. Other generation methods and selection strategies will be applied to BSO algorithm in future research to obtain better BSO, which is our near-future research work.

References

1. de Castro, L.N., Timmis, J.: Artificial Immune Systems: A New Computational Intelligence Approach. Springer (November 2002)
2. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 26(1), 29–41 (1996)

3. Shi, Y.: Brain Storm Optimization Algorithm. In: Tan, Y., Shi, Y., Chai, Y., Wang, G. (eds.) ICSI 2011, Part I. LNCS, vol. 6728, pp. 303–309. Springer, Heidelberg (2011)
4. Shi, Y.: An optimization algorithm based on brainstorming process. International Journal of Swarm Intelligence Research (IJSIR) 2(4), 35–62 (2011)
5. Shi, Y., Eberhart, R.: A modified particle swarm optimizer. In: Proceedings of the 1998 Congress on Evolutionary Computation (CEC1998), pp. 69–73 (1998)
6. Smith, R.: The 7 Levels of Change: Different Thinking for Different Results, 2nd edn. Tapestry Press (May 2002)
7. Yang, X.S.: Nature-Inspired Metaheuristic Algorithm. Luniver Press (February 2008)
8. Yao, X., Liu, Y., Lin, G.: Evolutionary programming made faster. IEEE Transactions on Evolutionary Computation 3(2), 82–102 (1999)

Group Search Optimizer for Power System Economic Dispatch

Huilian Liao[1], Haoyong Chen[1], Qinghua Wu[2],
Masoud Bazargan[3], and Zhen Ji[4]

[1] South China University of Technology, Guangdong, China
[2] University of Liverpool, Liverpool, L69 3GJ, U.K.
[3] ALSTOM Grid UK Limited
[4] Shenzhen University, Shenzhen, 518060, China

Abstract. This paper presents the application of a group search optimizer (GSO) to solve a power system economic dispatch problem, which is to reduce the fuel cost and transmission line loss in the power system. GSO is inspired by animal searching behavior and group living theory. The framework of GSO is mainly based on the cooperation of producer, scroungers and rangers, which play different roles during the search. GSO has been successfully applied to solve a wider range of benchmark functions [1]. This paper investigates the application of GSO to resolve the power system economic dispatch problem with consideration of minimizing the objectives of fuel cost and transmission line loss. The performance of GSO has been compared with that of genetic algorithm (GA) and particle swarming optimizer (PSO), and the simulation results have demonstrated that GSO outperforms the other two algorithms. The application is also extended to determine the optimal locations and control parameters of flexible AC transmission system (FACTS) devices to achieve the objective. Simulation studies have been carried out on a standard test system and better results have been obtained by GSO.

Keywords: group search optimizer, animal searching behavior, economic dispatch, FACTS devices.

1 Introduction

Evolutionary Algorithms (EAs), which stem from the study of adaptation in natural and artificial systems [2], have been investigated comprehensively in last decades. Particle Swarm Optimization (PSO) [3], Genetic Algorithm (GA) [4] and other population-based optimization techniques [5] have been applied widely for problem solving. Recently, group search optimizer (GSO) was proposed by He *et al*. [6], inspired by animal behavior in nature. GSO employs the theory of group living, which is a widespread phenomenon in the animal kingdom. GSO especially concerns animal searching behavior and utilizes Producer-Scrounger (PS) biological model [7], which assumes group members search either for 'finding' (producer) or for 'joining' (scrounger) opportunities. In both searching patterns, GSO employs animal scanning mechanism. The performance of GSO is

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 253–260, 2012.

not sensitive to other parameters such as maximal pursuit angle, which makes it particularly attractive for real-world applications. An extensive discussion and intensive analysis of GSO can be found in [1], in which comprehensive comparison between GSO and other EAs on a range of single-objective benchmark functions is reported.

Power system economic dispatch has received attention for a long time [8]. It aims at finding a solution to an optimal control of a power system with an objective to reach. The objective is commonly concerned with Summer fuel cost in power plants and Summer power loss in transmission lines. This paper investigates the application of GSO in solving this economic dispatch problem. Although GSO has been applied to solve economic dispatch problem considering valve loading effects [9], the problem concerned in this paper is different. GSO is evaluated for a an optimization problem considering fuel cost and transmission loss, in simulation studies on IEEE 14-bus system, in comparison with GA and PSO respectively, to demonstrate the capability of GSO for finding better solutions. GSO is also applied to incorporate the co-ordinated control of FACTS devices for solving the power system dispatch problem. FACTS control has been regarded as a promising approach to improve power system stability and dispatch economy [10], and lower operational cost without introducing additional cost [11] [12]. Besides, FACTS devices can be controlled flexibly and rapidly. By placing FACTS devices in optimal locations and setting optimal control parameters, the power flow can be controlled so that more stable and efficient power system operation can be achieved. Based on a preliminary study of the optimal placement of FACTS devices [13], GSO is evaluated for coordinated control of FACTS devices, by simulation studies on the IEEE 14-bus system, in comparison with GA and PSO. The results will be presented to show that GSO outperforms GA and PSO in terms of finding accurate solutions.

2 Group Search Optimizer

GSO has a population called a *group* which has m individuals and each individual in the group is called a *member*. Each member has its own position and the fitness values of all objective functions. The process of calculating the fitness values of all objective functions associated with each member is called an *iteration*, in which the group search behavior performs once with change in the position of each member. According to the biological PS model, a group consists of three kinds of members:

- A producer is the leader of the group, who searches the food and shares information with the rest of the group. In each iteration, the producer is renewed by selecting the best member from the group.
- Scroungers follow the producers, and search resource uncovered by other scroungers. Aside from the producers, 80% of the rest members are randomly selected as scroungers and the scroungers are renewed in each iteration as well.

– The remainder members are rangers. They walk randomly in the searching space. This behavior allows the group to discover resources far away.

In an n-dimensional searching space, the i_{th} member at the k_{th} iteration, has a current position $X_i^k \in \mathbb{R}^n$, a head angle $\varphi_i^k = (\varphi_{i1}^k, ..., \varphi_{i(n-1)}^k) \in \mathbb{R}^{n-1}$ and a head direction $D_i^k(\varphi_i^k) = (d_{i1}^k, ..., d_{in}^k) \in \mathbb{R}^n$ which can be calculated from φ_i^k via a Polar to Cartesian coordinates transformation:

$$\begin{aligned} d_{i1}^k &= \prod_{v=1}^{n-1} \cos(\varphi_{iv}^k) \\ d_{ij}^k &= \sin(\varphi_{i(j-1)}^k) \cdot \prod_{v=1}^{n-1} \cos(\varphi_{iv}^k),\ 2 \le j \le n-1 \\ d_{in}^k &= \sin(\varphi_{i(n-1)}^k). \end{aligned} \tag{1}$$

The searching behavior of the producer X_p^k is carried out in such a way that firstly, it selects three positions in the scanning field [14]. Aside from the producer, a large percent of the rest members in the group are selected as scroungers. The scroungers keep searching for opportunities to locate the resources around the producer, which means a scrounger moves towards the producer and searches in a small area around it. Besides producers and scroungers, the rest members are rangers. The range behavior is an initial phase of a searching behavior that starts without cues leading to a specific resource [15]. The searching strategies used by rangers include random walk and systematic searching strategies, which help to locate resources efficiently [16]. The detail introduction to the behavior of producer, scrounger and ranger can refer to [1].

3 The Application of GSO in Economic Dispatch Problems

3.1 Problem Formulation

In the power system economic dispatch problem, we need to optimize control variables of the power system towards the target of minimizing fuel cost and transmission line loss. Control variables are generator real power outputs P_{G} expect at the slack bus P_{G_1}, generator voltages V_{G}, transformer tap setting T, and reactive power generations of VAR sources Q_c. Suppose there are N_{G} generator buses in the system and denote the control variables by a vector $\mathbf{u}$. Aside from control variables, there also exist the so-called dependent variables, $\mathbf{x}$, which involve slack bus power $P_{\text{G}1}$, load bus voltage V_{L}, generator reactive power outputs Q_{G} and apparent power flow S. Suppose there are N_{L} load buses and N_{E} transmission lines in the system.

In simulation studies, $\mathbf{u}$ acts as the 'position' X of the members and is updated by GSO. Obviously, the problem has a dimension of $N_{\text{G}} \times 2 - 1$. The values of $\mathbf{x}$ are calculated from $\mathbf{u}$ by solving the power flow, *i.e.*, the inequality constraint of

(6), using Newton-Raphson method. Both $\mathbf{u}$ and $\mathbf{x}$ are involved in the evaluation of the fitness values.

The objective function is concerned with the power loss which includes the fuel cost of power plants and the transmission line loss, which is expressed as:

$$F(\mathbf{x}, \mathbf{u}) = \sum_{i=1}^{N_{\mathrm{G}}} f_i + \lambda \mathrm{PL}(\mathbf{x}, \mathbf{u}), \tag{2}$$

where f_i is the fuel cost (\$/h) of the i_{th} generator:

$$f_i = a_i + b_i P_{\mathrm{G}i} + c_i P_{\mathrm{G}i}^2, \tag{3}$$

a_i, b_i, and c_i are the fuel cost coefficients, $P_{\mathrm{G}i}$ is the real power output generated by the i_{th} generator, λ is the coefficient between the fuel cost and transmission line loss, which is set to be 5.1×10^5. By adjusting λ, the feasible range of fuel cost is equal as the feasible range of transmission line loss. PL stands for the transmission line loss, which is expressed as:

$$\mathrm{PL}(\mathbf{x}, \mathbf{u}) = \sum_{i=1}^{N_{\mathrm{E}}} P_i \tag{4}$$

where and P_i is the real power loss representing the fuel cost in line i calculated from Q_{G} and S.

Equality Constraints. The equality constraints are that the input and output reactive power at each bus should be equal as well. Therefore, $g(\mathbf{x}, \mathbf{u})$ can be formulated by nonlinear power flow equation as follows:

$$Q_{\mathrm{G}i} = Q_{\mathrm{D}i} + V_i \sum_{j=1}^{N_i} V_j (G_{ij} \sin\theta_{ij} - B_{ij} \cos\theta_{ij}) \qquad i = 1, \cdots, N_{\mathrm{PQ}} \tag{5}$$

where $Q_{\mathrm{D}i}$ is demanded reactive power at bus i; G and B are the real and imaginary part of the admittance matrix of the system, respectively; N_i is the number of buses adjacent to bus i including bus i; N_{PQ} and N_0 are the number of PQ buses and total buses excluding slack bus, respectively. Using the Newton method mentioned before, $\mathbf{x}$ can be solved from these power flow functions.

Inequality Constraints. The inequality constraints $h(\mathbf{x}, \mathbf{u})$ are limits of control parameters. Reactive power Q_{G} and voltage V_{G} are restricted by their limits as follows:

$$\begin{aligned} {Q_{\mathrm{G}i}}^{\min} \le Q_{\mathrm{G}i} \le {Q_{\mathrm{G}i}}^{\max} \; i = 1, \cdots, N_{\mathrm{G}} \\ {V_{\mathrm{G}i}}^{\min} \le V_{\mathrm{G}i} \le {V_{\mathrm{G}i}}^{\max}. \end{aligned} \tag{6}$$

The constraints of voltage at load buses V_{L} and apparent power flow S are represented as:

$$\begin{aligned} {V_{\mathrm{L}i}}^{\min} \le V_{\mathrm{L}i} \le {V_{\mathrm{L}i}}^{\max} \; i = 1, \cdots, N_{\mathrm{L}} \\ |S_i| \le S_i^{\max} \qquad\qquad\quad i = 1, \cdots, N_{\mathrm{E}}. \end{aligned} \tag{7}$$

In the simulation studies, if the values of control variables exceed their limits, they are restricted to the boundary. If the values of dependent variables are beyond the range, the fitness functions will have large values; hence, the corresponding **u** is rejected by GSO.

3.2 Simulation Results

Simulation studies are carried out on IEEE 14-bus system, which consists of 20 branches, 5 generator buses and 11 load buses. The total system loads are 259 MW and 81.3 MVar. The single-line diagram of the system can be found in [13].

Two EAs, GA [17] and PSO [18], are employed to compare with GSO in the simulation studies. In all the studies, for GA and PSO, the population size and the maximal number of iterations are set to 50 and 1000, respectively. As for GSO, the population size and the maximal number of iterations are set to 47 and 1000, respectively The average and best results, calculated from 30 trial runs of these algorithms respectively, are given in Table 1. The fitness value obtained by GSO can be minimized to 8079.4218 $/h. Among the three algorithms, GSO obtains the best result in both terms of average and minimum. The results of the three algorithms is also measured statistically by Kruskal-Wallis test. The obtained P-value is 0.003, which suggests that the results are not significant.

Table 1. The optimization results on the IEEE 14-bus system

F ($/h)	Optimization algorithms		
	GSO	PSO	GA
Summer	**8079.4218**	8079.6003	8079.8315
Average	**8079.8992**	8079.9408	8080.5937

4 Application of GSO in Economic Dispatch with FACTS Devices Involved

4.1 Problem Formulation

FACTS Devices. FACTS represents a recent technological development of electric power systems. The adoption of FACTS devices increases the stability of transmission lines, improves the security of the system and the power flow can be controlled by adjusting the their control variables.

Four FACTS devices in four different types are to be placed to control power flow. The optimal placement of these four devices in four different locations and the determination of their control parameters are required. The first is Thyristor Controlled Series Capacitor (TCSC), which permits the modification of transmission line reactance. The second is Thyristor Controlled Phase Shifting Transformer (TCPST), which controls the phase-angle between the bus voltages at the two ends of a branch. Thyristor Controlled Voltage Regulator (TCVR) is also selected to act principally on the magnitude difference between the bus voltages

at the two ends of a branch. Static Var Compensator (SVC) is used as the fourth type of FACTS devices to absorb or inject reactive power at the bus which is chosen to place an SVC.

The TCSC may have one of the two possible characteristics: capacitive or inductive, corresponding to decreasing or increasing the reactance of the line, X_{L}, respectively. The value of the capacitance or inductance of the TCSC, X_{S}, is limited to:

$$-0.8X_{\mathrm{L}} < X_{\mathrm{S}} < 0.2X_{\mathrm{L}} \tag{8}$$

The TCPST acts by adding a quadrature component to the prevailing bus voltage in order to adjust its angle. The model used for this device is an ideal phase shifter which has series impedance equal to zero. It is inserted in series in a transmission line and may take a value of angle: θ_{P}, which is bounded by:

$$-5^{\circ} < \theta_{\mathrm{P}} < 5^{\circ} \tag{9}$$

The TCVR operates by inserting an in-phase voltage to the main bus voltage so as to change its magnitude. An ideal tap-change transformer without series impedance is used to model this controller. The value of the turns ratio T_{V} is chosen in the following range:

$$0.9 < T_{\mathrm{V}} < 1.1 \tag{10}$$

The SVC has two different characteristics as well: inductive or capacitive. In the first case it absorbs reactive power while in the second the reactive power is injected. The value of reactive power injected or absorbed, Q_{S}, is limited between:

$$-100\ \mathrm{MVar} < \mathrm{Q_S} < 100\ \mathrm{MVar} \tag{11}$$

Control Variables. Placing FACTS devices in the power system means more control variables are to be optimized aside from those listed in the previous section. New control variables include the locations of the FACTS devices L and their control parameters X_{S}, θ_{P}, T_{V} and Q_{S}. Suppose N_{F} FACTS devices are installed in the system, which includes N_1 TCSC, N_2 TCPST, N_3 TCVR and N_4 SVC. The dependent variables and objective function are the same as explained in subsection 3.1.

Inequality Constraints. The setting parameters of multi-type FACTS devices are restricted by their limits as follows:

$$\begin{aligned}
&{X_{\mathrm{S}}}_i^{\min} \le X_{\mathrm{S}i} \le {X_{\mathrm{S}}}_i^{\max} \quad i = 1, \cdots, N_1 \\
&{\theta_{\mathrm{P}}}_i^{\min} \le \theta_{\mathrm{P}i} \le {\theta_{\mathrm{P}}}_i^{\max} \quad i = 1, \cdots, N_2 \\
&{T_{\mathrm{V}}}_i^{\min} \le T_{\mathrm{V}i} \le {T_{\mathrm{V}}}_i^{\max} \quad i = 1, \cdots, N_3 \\
&{Q_{\mathrm{S}}}_i^{\min} \le Q_{\mathrm{S}i} \le {Q_{\mathrm{S}}}_i^{\max} \quad i = 1, \cdots, N_4
\end{aligned} \tag{12}$$

The limits of control parameters for the FACTS devices are given in subsection 4.1.

4.2 Simulation Results

In the following simulation study, four FACTS devices with one device of each type needing to be installed in the IEEE 14-bus system. GSO is compared with PSO and GA respectively. The optimization results of 30 trial runs are listed in Table 2. Comparing between Table 1 and 2, it can be seen that with the placement of FACTS devices, the system can achieve a much smaller fitness value, no matter which optimization algorithm is adopted. The results given in Table 2 again show that GSO obtains the best optimization result among the three algorithms. The best solution for the optimal placement of the FACTS devices found by GSO is listed in Table 3, including the locations and control parameters.

Table 2. The optimization results on the IEEE 14-bus system with placement of FACTS devices

F	Optimization algorithms		
($/h)	GSO	PSO	GA
Summer	**8072.9411**	8073.0517	8073.4958
Average	**8073.1704**	8073.5277	8073.9002

Table 3. The placement of FACTS devices obtained by GSO on IEEE 14-bus system

FACTS devices	Location	Control parameters	Values
TCSC	Branch 7-8	X_S	-0.1974
TCPST	Branch 3-4	θ_P	-1.4663
TCVR	Branch 4-9	T_V	0.9000
SVC	Bus 6	Q_S	56.4143

5 Conclusion

In this paper, a promising optimization algorithm, group search optimizer (GSO), has been applied to solve the economic dispatch problem, which is to reduce the fuel cost and transmission loss in power system. Simulation studies have been carried out on a standard test systems, and GSO have been compared with two popular EAs, PSO and GA. The simulation results have demonstrated that GSO outperforms GA and PSO in terms of finding accurate solutions. GSO has also been applied to solve the dispatch problem when FACTS are involved. Simulation studies have shown that GSO is capable of optimizing the locations and control parameters of FACTS devices accurately. With the optimal placement of FACTS devices, the fuel cost and transmission line loss can be significantly improved.

References

1. He, S., Wu, Q.H., Saunders, J.R.: Group search optimizer - an optimization algorithm inspired by animal searching behavior. IEEE Transactions on Evolutionary Computation (2009)
2. Bounds, D.G.: New optimization methods from physics and biology. Nature 329(6136), 215–219 (1987)
3. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, Australia, pp. 1942–1948 (November 1995)
4. Holland, J.H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975)
5. Kang, F., Li, J.J., Ma, Z.Y.: Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions. Information Sciences 181, 3508–3531 (2011)
6. He, S., Wu, Q.H., Saunders, J.R.: A novel group search optimizer inspired by animal behavioral ecology. In: 2006 IEEE Congress on Evolutionary Computation, pp. 16–21 (July 2006)
7. Barnard, C.J., Sibly, R.M.: Producers and scroungers: a general model and its application to captive flocks of house sparrows. Animal Behaviour 29, 543–550 (1981)
8. Wu, Q.H., Ma, J.: Power system optimal reactive dispatch using evolutionary programming. IEEE Transactions on Power Systems 10(3), 1243–1249 (1995)
9. Zare, K., Haque, M.T., Davoodi, E.: Solving non-convex economic dispatch problem with valve point effects using modified group search optimizer method. Electric Power Systems Research 84(1), 83–89 (2011)
10. Acha, E., Fuerte-Esquivel, C.R., Ambriz-Perez, H., Angeles-Camacho, C.: FACTS: Modelling and Simulation in Power Networks. Wiley, West Sussex (2005)
11. Haque, M.H.: Evaluation of first swing stability of a large power system with various FACTS devices. IEEE Transactions on Power Systems 23(3), 1144–1152 (2008)
12. Shao, W., Vittal, V.: LP-based OPF for corrective FACTS control to relieve overloads and voltage violations. IEEE Transactions on Power Systems 21(4), 1832–1839 (2006)
13. Wu, Q.H., Lu, Z., Li, M.S., Ji, T.Y.: Optimal placement of FACTS devices by a group search optimizer with multiple producer. In: IEEE World Congress on Computational Intelligence, Hong Kong, June 1-6, pp. 1033–1039 (2008)
14. O'Brien, W.J., Evans, B.I., Howick, G.L.: A new view of the predation cycle of a planktivorous fish, white cappie. Canadian Journal of Fisheries and Aquatic Sciences 43, 1894–1899 (1986)
15. Dusenbery, D.B.: Ranging strategies. Journal of Theoretical Biology 136, 309–316 (1989)
16. Higgins, C.L., Strauss, R.E.: Discrimination and classification of foraging path produced by search-tactic models. Behavior Ecology 15, 248–254 (2003)
17. Wu, Q.H., Cao, Y.J., Wen, J.Y.: Optimal reactive power dispatch using an adaptive genetic algorithm. International Jounal of Electrical Power Energy System 20(8), 563–569 (1998)
18. Zhao, B., Guo, C.X., Cao, Y.J.: A multiagent-based particle swarm optimization approach for optimal reactive power dispatch. IEEE Transactions on Power Systems 20(2), 1070–1078 (2005)

An Improved Bean Optimization Algorithm for Solving TSP

Xiaoming Zhang[1], Kang Jiang[2], Hailei Wang[1], Wenbo Li[1], and Bingyu Sun[1]

[1] Institute of Intelligent Machines, Chinese Academy of Sciences
[2] Hefei University of Technology
230031 Hefei, Anhui, P.R. China
xmzhang@iim.ac.cn

Abstract. Inspired by the transmission of beans in nature, a novel swarm intelligence algorithm-Bean Optimization Algorithm (BOA) is proposed. In the area of continuous optimization problems solving, BOA has shown a good performance. In this paper, an improved BOA is presented for solving TSP, a typical discrete optimization problem. Two novel evolution mechanisms named population migration and priori information cross-sharing are proposed to improve the performance of BOA. The improved BOA algorithm maintains the basic idea of BOA and overcomes the shortcoming that BOA with continuous distribution function can not be applied to solve the discrete optimization problems. The experimental results of TSP show that the improved BOA algorithm is suit for solving discrete optimization problems with high efficiency.

Keywords: swarm intelligence, Bean Optimization Algorithm, TSP, population migration, priori information, discrete optimization.

1 Introduction

The traveling salesman problem (TSP) is a kind of NP-hard problems in combinatorial optimization area. Given a list of cities and their pairwise distances, the task is to find a shortest possible tour that each city is exactly visited once. The TSP has several applications even in its purest formulation, such as planning, logistics, and the manufacture of microchips. Slightly modified, it appears as a sub-problem in many areas, such as robot path plan. In many applications, additional constraints such as limited resources or time windows make the problem considerably harder.

Swarm intelligence optimization algorithm is a kind of modern optimization methods which imitate or refer to the acts of nature biological swarm systems. The typical algorithms include ant colony optimization (ACO) [1], particle swarm optimization (PSO)[2]. Inspired by the transmission mode of seeds, a novel swarm intelligence optimization algorithm named Bean Optimization Algorithm (BOA) is proposed. It is the combination of nature evolutionary tactic and limited random search. BOA can be used to solve complex optimization problems by simulating the adaptive phenomenon of plants in the nature. BOA has stable robust behavior on explored tests and stands out as a promising alternative to existing optimization methods for engineering designs or applications [3].

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 261–267, 2012.

At present, based on the basic BOA algorithm, a variety of improved BOA algorithms have been proposed, such as BOA based on normal distribution, BOA based on negative binomial distribution. BOA algorithm has been successfully used to solve many continuous optimization problems [4] [5]. In the discrete domain, the researches and applications of BOA are still limited. Only in the reference [6], combined with ant colony optimization algorithm, BOA solved TSP successfully. In this paper, an improved BOA is proposed to solve discrete optimization problems and achieves good results.

2 Introduction of Traveling Salesman Problem

Traveling Salesman Problem (TSP) was proposed by K. Menger in 1932. Since then, it has been a concern of many researchers. It is one of the most intensively studied problems in optimization and is used as a benchmark for many optimization methods. TSP is to find the shortest way of visiting all of the cities and returning to the starting city. Though the rule of TSP is simple, with the increase of the number of travel cities, the difficulty of solving TSP will grow rapidly and there will be index explosion. Now many mathematicians try to find an efficient algorithm to solve TSP. There are $1000 Prize for solving the TSP problem of mona-lisa100K[7].

According to the definition of traveling salesman problem, its mathematical description is:

$$\begin{aligned} &\min \sum d_{ij} x_{ij} \\ &s.t. \sum_{j=1}^{n} x_{ij} = 1,\ i = 1,2,3,...,n \\ &\sum_{i=1}^{n} x_{ij} = 1,\ j = 1,2,3,...,n \\ &\sum_{i,j \le S} x_{ij} \le |S| - 1, \qquad 2 \le |S| \le n-2,\ S \subset \{1,2,...,n\} \\ &x_{ij} \in \{0,1\} \quad i,j = 1,2,...,n \quad i \ne j \end{aligned} \tag{1}$$

In the above description, d_{ij} is the distance between city i and city j. $x_{ij} = 0$ means that route $i \to j$ has not been chosen. $x_{ij} = 1$ means that route $i \to j$ has been chosen. The first two constraints guarantee that each city will be passed through once. The third constraint guarantees that the answer does not loop in any city subset. Variable S is the subset of cities in city set.

3 An Improved BOA for Solving TSP

3.1 Introduction of Bean Optimization Algorithm

Inspired by the transmission mode of beans, we propose a swarm intelligence algorithm namely Bean Optimization Algorithm (BOA) which can be used to solve complex optimization problems by simulating the adaptive phenomenon of plants in the

nature. Now the algorithm has been used in function optimization, scheduling, etc. In BOA, the position of a individual bean is expressed with real number vector like

$$X = \{ x_1, x_2, x_3, \ldots, x_n \}, \tag{2}$$

where n is determined by the scale of problem to be resolved. The bean group is composed of a large number of beans. And the size of the bean group can be adjusted according to the practical situations. In addition, beans are sown to the region which is defined by the problem. Father beans are those beans whose fitness values are larger than most of others. In BOA, the number and distribution of descendant beans will be set according to their father bean's fitness value. The basic equation of BOA is shown as following:

$$X[i] = \begin{cases} X[i], & \text{if } X[i] \text{ is a father bean} \\ X_{mb} + \text{Distribution}(X_{mb}) \times A, & \text{if } X[i] \text{ is not a father bean} \end{cases} \tag{3}$$

In the above equation, X[i] is the position of bean i. X_{mb} is the position of the father bean. Distribution(X_{mb}) is the a random variable with a certain distribution of father bean in order to get the positions of its descendants. Parameter A can be set according to the range of the problem to be resolved.

In addition, the distribution of some beans does not follow the equation discussed above. They choose random positions in order to reinforce the global optimization performance.

When the descendant beans finished locating, their fitness values are to be evaluated. The beans with most optimal fitness value will be selected as the candidates of father beans in the next generation. The candidates of father beans should also satisfy the condition that the distance between every two father beans should be larger than the distance threshold. This condition assures that the father beans can have a fine distribution to avoid premature convergence and enhance the performance of the BOA for global optimization. If all the conditions can be satisfied, the candidate can be set as the father bean of next generation.

3.2 Core Operation of the Improved BOA

BOA algorithm uses population evolution mechanism to find the optimal solution. Because most of the population evolution models are continuous, they are difficult to solve discrete optimization problems, such as TSP. To this end, an improved BOA for solving discrete optimization problems is proposed by taking advantage of the crossover idea of genetic algorithm. The algorithm design is shown as follows.

1) Code individual beans

According to the definition of TSP, the dimension n of location of individual beans is set to be the number of cities. Each element of a bean's position vector represents a city and they are not repeated. The position vector of a individual bean is set as $X = (x_1, x_2, \ldots, x_i, \ldots, x_j, \ldots, x_n)$. That means there is a route as $x_1 \rightarrow x_2 \rightarrow \ldots \rightarrow x_n$ and $x_i \neq x_j$ $(i \neq j)$.

2) Population migration
In the process of population migration, at least two populations should be initialized. The best individual (father bean) in each population will be involved in cross-species operation through the interaction between populations in order to promote the prosperity of populations.

3) Cross-sharing of a priori information
In order to keep the priori information of the father beans, there are cross operations between the father beans and the individual beans to produce new offspring individuals. The specific operation is shown as follows.

(1) Select a random position in the vectors of a father bean f and an individual bean s separately as a cross-region.
(2) Exchange cross-region between f and s. Then delete the duplicate elements in f and s separately. Two new offspring individuals a and b will be generated.

For example: Let $n = 5$. The vector of father bean $f = (1\ 2 \mid 34 \mid 5)$. The individual bean $s = (5\ 4 \mid 3\ 2 \mid 1)$. The cross-region randomly selected is the sections: 34 and 32. The offspring individuals after cross operation are $a = (1\ 4\ 3\ 2\ 5)$ and $b = (5\ 2\ 3\ 4\ 1)$.

4) Algorithm design
The sketch diagram of population migration for TSP solving is shown as follows.

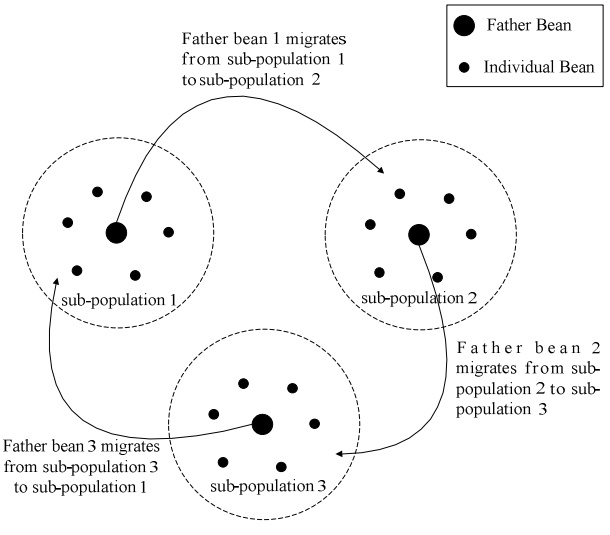


Fig. 1. Sketch diagram of population migration

According to the principle of BOA, the population should be initialized firstly (let the size of population be m). According to the fitness values of individual beans, select the father beans (let the number of father beans be 3): f_1, f_2, f_3. $(m-3)/3$ individuals will be screened as sub-populations 1 according to the Euclidean distance between individual beans and f_1. Use the same method, sub-populations 2 and sub-population 3 will also be generated. Then let f_2 be the cross father bean of sub-population 3 and cross operations will be carried out between f_2 and individual beans in sub-population

3. Select the offspring with the best fitness value to displace the former individual bean in sub-population 3. Let f_3 be the cross father bean of sub-population 1 and cross operations will be carried out between f_3 and individual beans in sub-population 1. Select the offspring with the best fitness value to displace the former individual bean in sub-population 1. Let f_1 be the cross father bean of sub-population 2 and cross operations will be carried out between f_1 and individual beans in sub-population 2. Select the offspring with the best fitness value to displace the former individual bean in sub-population 2. Repeat the above process until the termination condition is met. The position vector of the bean with the minimal fitness value is the optimal solution. The pseudocode of BOA for solving TSP is described as follows:

Table 1. Pseudocode of BOA

```
Set the number of iterations be q.
Randomly generate m initial beans.
Calculate the fitness value of the initial beans and select z
father beans.
Generate z sub-populations by clustering algorithm.
While(the number of iterations<q)
  For  i=1:z
    For  j=1:m
          Cross operations are carried out between X_j and
     f_(i+1);
      The bean with the best fitness value is recorded as X_j1;
         X_i=X_j1;
    End
    Father beans update;
  End
End
Output the optimal solution.
```

4 Experiment and Analysis

4.1 Introduction of TSP Experiment

We select five typical TSP problems and show them in table 2.

Table 2. Introduction of TSP Experiment

Parameters / TSP	Number of Cities	Optimal Route Length	Type of Distance
ULYSSES22	22	75.31	Euclidean Distance
BAYG29	29	9074.15	
OLIVER30	30	423.74	
EIL51	51	426	
BERLIN52	52	7542	

4.2 Parameter Settings of BOA

The parameters of BOA algorithm are set in table 3.

Table 3. Parameters of BOA

Parameters of BOA / TSP	Population Scale	Number of Sub-populations	Number of Father Beans	Number of Iterations
ULYSSES22	50	3	3	200
BAYG29	50	3	3	200
OLIVER30	50	3	3	200
EIL51	100	3	3	1000
BERLIN52	100	3	3	1000

4.3 Experimental Results

The experiments were carried out on a PC with a T8100 2.10-GHz Intel Processor and 2.0-GB RAM. The operating system was Microsoft Windows XP. All the programs were written and executed in MATLAB 2010a except Max-Min AS[8] which is in the open source software OAT [9]. We compared BOA with Cross-PSO [10] and Max-Min AS and each experiment was done 30 times. The maximum scale of cross-region is 8. The experimental results are shown in table 4 and following figures.

Table 4. TSP Experimental Results

Experimental Results / TSP	the Best Result			the Worst Result			the Everage Number of Iterations		
	BOA	Cross-PSO	Max-Min AS	BOA	Cross-PSO	Max-Min AS	BOA	Cross-PSO	Max-Min AS
ULYSSES22	75.31	75.88	77.00	76.08	77.51	77.00	68.67	146.17	89
BAYG29	9074.15	9213.89	9081.43	9166.28	10769.39	9081.43	131.73	169.45	147
OLIVER30	423.74	497.06	433.14	424.69	457.94	433.14	89.23	195.67	169
EIL51	434.46	468.09	427.80	447.11	472.38	427.80	317.53	898.07	631
BERLIN52	7542.30	7916.41	7542.00	7835.04	8353.52	7542.00	232.45	766.86	415

4.4 Experimental Results Analysis

We can see from the above results that the improved BOA can effectively solve TSP. The performance of the algorithm is mainly reflected in the two aspects: final solution and average number of iterations. The performance of Cross-PSO algorithm is the worst compared with improved BOA and Max-Min AS both in the two aspects of the performance. The improved BOA gets the best solutions (also the optimal solutions) when solving the TSP problems: ULYSSES22, BAYG29 and OLIVER30. When

solving EIL51 and BERLIN52, Max-Min AS got the best final solution. The results show that Max-Min AS is more suitable for solving combinatorial optimization problems than other two algorithms. But the average number of iterations of Max-Min AS is bigger than that of BOA. That means BOA has better convergence rate. When solving BERLIN52 problem, BOA also got an approximate optimal solution.

5 Conclusions

In the area of continuous optimization problems solving, BOA has shown a good performance. In this paper, an improved BOA is presented for solving TSP, a typical discrete optimization problem. Two new evolution mechanisms named population migration and priori information cross-sharing are proposed to improve the performance of BOA. The improved BOA algorithm maintains the basic idea of BOA and overcomes the shortcomings that BOA with continuous distribution function can not be applied to solve the discrete optimization problems. The TSP experimental results show that the improved BOA algorithm is suit for solving discrete problems with high efficiency. In the future, we will explore more effective optimization mechanism to improve BOA and focus on large-scale TSP problems to test it.

Acknowledgments. The work of this paper has been supported by the Knowledge Innovation Program of the Chinese Academy of Sciences, and the national natural science foundation of china under grant No. 91024008, 41101516.

References

1. Dorigo, M., Birattari, M., Stützle, T.: Ant Colony Optimization– Artificial Ants as a Computational Intelligence Technique. IEEE Computational Intelligence Magazine 11(4), 28–39 (2006)
2. Kennedy, J.: The Particle Swarm: Social Adaptation of Knowledge. In: 1997 IEEE International Conference on Evolutionary Computation, pp. 303–308. IEEE Press, New York (1997)
3. Zhang, X., Wang, R., Song, L.: A Novel Evolutionary Algorithm—Seed Optimization Algorithm. Pattern Recognition and Artificial Intelligence 21(5), 677–681 (2008)
4. Wang, P., Cheng, Y.: Relief Supplies Scheduling Based on Bean Optimization Algorithm. Economic Research Guide (8), 252–253 (2010)
5. Zhang, X., Sun, B., Mei, T., Wang, R.: Post-disaster Restoration Based on Fuzzy Preference Relation and Bean Optimization Algorithm. In: 2010 IEEE Youth Conference on Information, Computing and Telecommunications, pp. 253–256. IEEE Press, New York (2010)
6. Li, Y.: Solving TSP by an ACO-and-BOA-based Hybrid Algorithm. In: 2010 International Conference on Computer Application and System Modeling, pp. 189–192. IEEE Press, New York (2010)
7. Mona Lisa TSP Challenge, `http://www.tsp.gatech.edu/data/ml/mona-lisa100K.tsp`
8. Stützle, T., Hoos, H.: MAX-MIN Ant System. Future Generation Computer Systems 16(8), 889–891 (2000)
9. Optimization Algorithm Toolkit, `http://optalgtoolkit.sourceforge.net`
10. Su, J., Wang, J.: Improved Particle Swarm Optimization for Traveling Salesman Problem. Computer Engineering and Applications 46(4), 52–75 (2010)

Cloud Droplets Evolutionary Algorithm on Reciprocity Mechanism for Function Optimization

Lei Wang, Wei Li, Rong Fei, and Xinghong Hei

Faculty of Computer Science and Engineering, Xi'an University of Technology, China
{leiwang,liwei,annyfei,heixinhong}@xaut.eud.cn

Abstract. For the problems of solving difficult problems in evolutionary algorithms such as easily falling into local optimum, premature convergence because of selective pressure, a complex and larger calculation and a lower accuracy of the solution, this paper proposes cloud droplets evolutionary model on reciprocity mechanism (CDER). The main idea of CDER is to simulate the phase transition of the cloud in nature which has vapor state, liquid state and solid state, and to combine the basic ideas of evolutionary computation to realize the population evolution. The condensation growth and collision growth of cloud droplets correspond to the competitive evolution and reciprocal evolution of species in nature. Experiments on solving the function optimization problems show that this model can enhance the individual competition and survival ability, guarantee the population diversity, accelerate the convergence speed and improve the solution precision through the iterative process of competition mechanism and reciprocity mechanism.

Keywords: reciprocity mechanism, competition mechanism, cloud droplets, evolutionary algorithms, phase transition.

1 Introduction

One of the typical characteristics of intelligent computing is to learn and simulate various forms of intelligent behavior in nature so that we can explore many ways and means to solve the problems. Cloud theory which brings forward by Academician Li Deyi is a method combining fuzzy and randomness [1-2], the model has the character of uncertainty with certainty, stability with variation in knowledge representation. Therefore, it reflects the basic principles of species evolution in nature. Zhang [3] proposes an evolution algorithm based on cloud model. The algorithm is called CEBA in this paper. The algorithm is simple and easy to realize. It has been achieved good results on uncertainty and ambiguity problems appeared in the process of the evolution. However, the algorithm still has some shortcomings. Firstly, in order to achieve rapid refinement in local exploitation, the algorithm reduces the evolutionary range. Therefore the population diversity and the exploration ability will fall, and the algorithm may converge prematurely because of selective pressure. Secondly, the algorithm adopts a mutation strategy to solve the premature convergence problem, but the mutation strategy simply enlarges the parameters. Therefore, the algorithm will

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 268–275, 2012.

have much unnecessary search and reduce the efficiency of the search. Finally, in the evolutionary process, the algorithm only emphasizes the survival of fittest while ignores another evolutionary strategy, reciprocal evolution. Therefore, the algorithm has lower exploitation capacity, slower convergence in the later stage of evolution, and lower convergence accuracy. For these problems, learn from the formation of cloud droplets in nature, especially inspired by the reciprocal evolutionary strategy, this paper puts forward the cloud droplets evolutionary model on reciprocity mechanism.

2 Cloud Droplets Variation and Evolutionary Mechanism

2.1 Cloud Droplets Variation

Cloud formation is the phase transition process. When the water vapor is saturated and stayed at some hygroscopic cloud condensation nuclei, the initial cloud droplet is condensed by heterogeneous nucleation. Cloud droplets constantly absorb water vapor to make themselves to condense and sublimate. When the cloud droplets move close to each other, larger cloud droplets are formed by colliding and coalescing between cloud droplets. When the temperature is below 0℃, ice phase is produced and a large number of supercooled water droplets are existed in the clouds. Through the sublimation of water vapor, ice crystals quickly grew up into snow crystal. The precipitation particles are eventually formed by way of the Bergeron process. If a lot of supercooled water droplets participate in the collision, snow crystals are transformed into spherical snow pellets. If they fall into the warm area where the temperature is above 0℃, they will melt into rain.

2.2 Cooperative Evolutionary Mechanism

Darwin has pointed out in “Origin of Species” that the evolution of biological organisms is the result of competition, which explained many natural phenomena successfully [4]. Martin Nowak from Harvard University ranks the cooperation as the third important factor of the species evolution, alongside of mutation and natural selection. Evolution is a unity of the cooperation and the opposition [5]. Every gene, every cell and every organism of the individual should strengthen its own evolutionary process at the expense of beating its rivals. Therefore, individuals often conflict with each other because of limited resources [6]. However, cooperation is existed in the evolution. Therefore, Nowak summarizes the cooperation evolution in five mechanisms, kin selection, direct reciprocity, indirect reciprocity, network reciprocity, and group selection.

J. B. S. Haldane puts forward that the altruism was existed in the evolution [7]. From the study of the social insect, Hamilton [8] proposed kin selection. Direct reciprocity refers if I help you now, you may help later. More classical game strategy is tit for tat, TFT [9]. Indirect reciprocity is a kind of more prevalent reciprocity form. Indirect reciprocity refers to helping someone to establish a good reputation, which

will be rewarded by others later. Indirect reciprocity will favor the cooperation evolution [10] if the probability, q, of someone's reputation exceeds the cost-to-benefit ratio of the altruistic act, namely q>c/b. Network reciprocity is a new form of reciprocity. Cooperation can form network clusters so that they can help each other. Group selection is a minimalist stochastic model [11]. The population is divided into several groups. Cooperators help others in their own group while defectors do not help.

In a word, collaborative is a most basic feature in all biological systems. In view of the cooperation and competition relation in the biological evolution, the algorithm realizes population evolution by cloud phase transformation, the competition and reciprocal evolution.

3 Cloud Droplets Evolutionary Model on Reciprocity Mechanism

Thermodynamic theory points out that if the various parts of the object do not change in status without any influences of outside conditions for a long time, it is called equilibrium state. Metastable equilibrium state is stable for infinitely small changes, and is an unstable state for large disturbance. If an object has different properties in different parts, it is called the non-uniform system or heterogeneous system. Each part is called a phase. From the cloud formation process it can be seen that the cloud has three phase, gaseous (water vapor), liquid (water droplets) and solid (snow crystal, snow pellets).

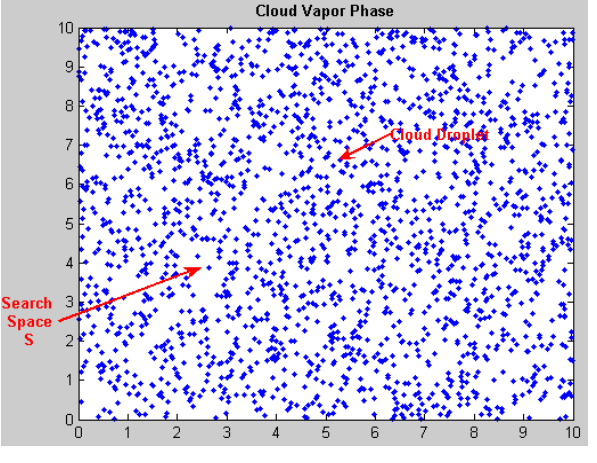


Fig. 1. Cloud vapor phase

Firstly, there are 10 populations, each population has 50 individuals. We use the cloud generator to generate 500 individuals, and each individual is called the cloud droplets. At this time, cloud droplets throughout the search space, such state is called

cloud vapor phase (as shown in Fig. 1). According to the laws of species evolution, the competition is in dominant while reciprocity is in disadvantaged in the search process. The populations realize the survival of the fittest in competitive evolution, good cloud droplets win and survive, and the system is in an unstable state. For reciprocal evolution, cloud droplets adapt to each other, strengthen the viability and reproductive ability of populations and maintain the orderly and diversity of the ecosystem. The system is in metastable equilibrium.

In the formation and development stage of cloud in nature, the cloud droplets can grow by condense (or sublimation). In addition, the big cloud droplets will collide with the small cloud droplets because of the different size and gravity. Then the collision growth phenomenon is happened. In this model, the condensation and collision growth of the cloud droplet corresponds to the cloud liquid phase (as shown in Fig. 2).

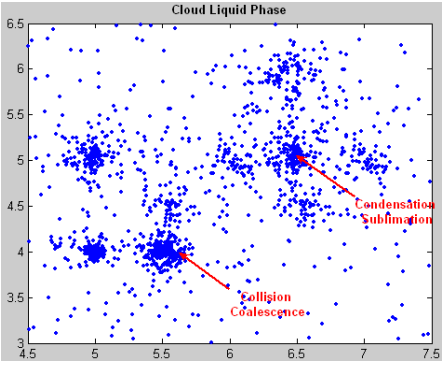


Fig. 2. Cloud liquid phase

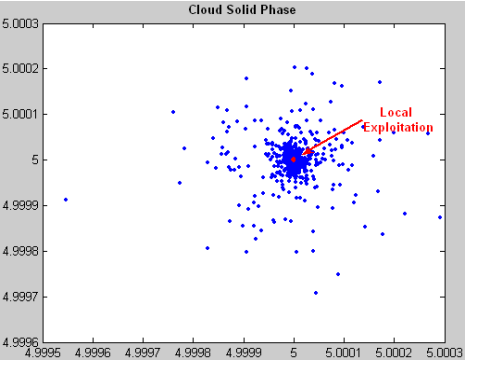


Fig. 3. Cloud solid phase

As the temperature continues to decrease, water vapor transfers to the ice crystals and ice crystals grow up. Ice crystals can quickly grow up from the snow crystal by condensation, and ultimately formed the precipitation particles. In this model, the formation of ice crystals, snow crystals corresponds to the cloud solid phase (as shown in Fig.3). It indicates that the population has found the current optimal solution area in the cloud solid phase. The population is in the local exploitation phase. In the exploitation process, if the number of successful evolution is more than a certain threshold, then the algorithm finds the optimal solution. The cloud is in the solid phase and the system is in steady state. If the failure evolutionary number exceeds a certain threshold, the cloud is changed from solid phase into liquid phase. The population search in a wider range. In cloud liquid phase, if the failure evolutionary number exceeds a certain threshold, the cloud is changed from liquid phase into vapor phase. Cloud droplets make the phase transition according to the degree of evolution until they find the optimal solution (as shown in Fig.4).

This algorithm uses indirect reciprocity mechanism as the cooperation model. The rule is that the populations will favor the cooperation evolution if the probability, q, of population's reputation exceeds the cost-to-benefit ratio of the altruistic act. Otherwise, it will favor the competitive evolution.

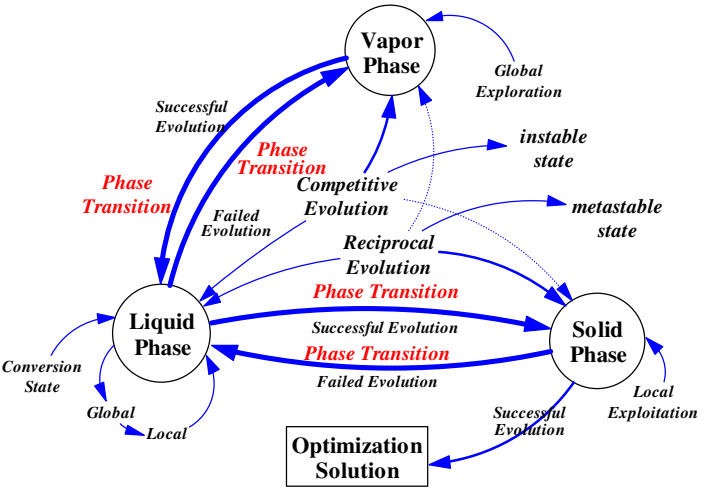


Fig. 4. Optimization process of CDER model

From the view of mathematics, CDER can be abstracted as a 5-tuple model CE = (VP, LP, SP, CR, A). CE represents the cloud droplets evolutionary system on reciprocal mechanism. The system is composed of five parts, cloud vapor phase VP, cloud liquid phase LP, cloud solid phase SP, competitive and reciprocal evolution CR, and the algorithm A. The core idea of the algorithm is as follows: cloud droplets carry out mutual phase transitions among cloud vapor phase, cloud liquid phase and cloud solid phase and according to the degree of evolution in every evolutionary generation. In the phase transition process, the cloud droplets realize adaptive competitive and reciprocal evolution. The algorithm is finished as soon as it converges to the optimal solution or very close to the optimal solution.

The cloud droplets are produced by cloud generator in CDER model. The cloud digital features are expressed by Expectation (Ex), Entropy (En) and Hyper-entropy (He). Ex is the expectation of all cloud droplets distributed in the domain. En is the uncertainty measure of the qualitative concept. He is the discrete degree of entropy. (The concept of Ex, En and He are derived from literature [1]).

4 CDER Model for Function Optimization

In order to test the performance of the proposed algorithm, a set of 10 benchmark functions [12] are selected. The comparison is taken among the CDER algorithm, opt-aiNet algorithm [13] and CEBA algorithm. The benchmark functions are listed in Table 1. The computer of experiment is Acer notebook 2.0GHz frequency and 1.5GB memory. The software is Matlab 7.0. We carry out experiments from the search space(S), the convergence speed (CS), the computation time (CT), the function evaluation number (FEN) and the best global optimum (BGO). Each experiment independently run 50 times.

Table 1. Comparison with opt-aiNet and CEBA

F	*Name of the Problem*	F	*Name of the Problem*
f_1	*Sphere model*	f_6	*Griewangk's function*
f_2	*Axis parallel hyperellipsoid*	f_7	*Shekel's Family(m = 5)*
f_3	*Schwefel's problem 1.2*	f_8	*Shekel's Family(m = 7)*
f_4	*Step function*	f_9	*Shekel's Family(m = 10)*
f_5	*Rastrigin's function*	f_{10}	*Easom function*

There are single-mode functions and multi-mode function of the selected functions. One function is 2-D function (f10). Others are high-dimensional functions. These functions can be divided into two categories according to the location of the global optimal solution. One category is the geometric center position in the domain, and the other is not. We can draw some conclusions from experimental results, see Table 2.

(1) Better convergence. Convergence analysis has become an index sign which usually evaluate the algorithm performance. In the tested functions, CDER can converge to the optimal solution that the algorithm has found within the limited number of function evaluations, while the other two functions have a gap between their solution and the optimal solution. This indicates that the algorithm opt-aiNet and algorithm CEBA have general effect on premature convergence problem while algorithm CDER can effectively ensure the diversity of the population by adaptive competitive and reciprocal evolution and avoids premature convergence.

(2) Convergence speed. Convergence rate are measured from three aspects. The first index sign is the convergent generations of the algorithm or it satisfied other termination conditions. In the 10 tested functions, CDER algorithm has fewer generations than opt-aiNet algorithm and CEBA algorithm. The data (see f1) show that CDER algorithm requires about 37 generations to converge, opt-aiNet algorithm needs about 144 generations to converge, while CEBA algorithm requires about 41 generations to converge. The second index sign is program running time. We can see from Table 2 that the time CDER algorithm spent is less than opt-aiNet algorithm and CEBA algorithm in tested functions. Take function f2 as example, the time of CDER algorithm spent is equivalent to 22% of opt-aiNet algorithm, and is equivalent to 60% of CEBA algorithm. Therefore, we can think that CDER algorithm not only saves more time but also converges quickly and more satisfies the engineering requirements. The third index sign is the number of function evaluation. The function evaluation times of CDER algorithm are less than opt-aiNet algorithm and CEBA algorithm, for example (f1), evaluation times of CDER algorithm is 31% of opt-aiNet algorithm and 87% of CEBA algorithm. In addition, the accuracy of the optimal solution found by CDER algorithm is higher than opt-aiNet algorithm and CEBA algorithm.

(3) Robustness. We have chosen various test functions in our experiment. They are representative and widely used. The test scale is so large that it can eliminate the influence of algorithm brought by the subjective and objective conditions. We have found that the performance indexes variance of the CDER algorithm is generally smaller than the opt-aiNet algorithm and CEBA algorithm after analyzing the table 2

comprehensively. This fully shows that CDER algorithm has stable performance, strong robustness, less susceptible to the influence of the function features and initialized population. The algorithm has a wide range of application.

Table 2. Comparison with opt-aiNet and CEBA

Fun	Algorithm	S	CS		CT (s)		FEN		BGO	
		ave	ave	SD	ave	SD	ave	SD	ave	SD
f_1	opt-aiNet	90000	144	41	0.248	0.037	61278	18646	2.035e-008	2.123e-008
	CEBA	30000	41	2.03	0.946	0.142	21990	959	9.427e-034	2.810e-033
	CDER	20000	37	1.53	0.071	0.019	19120	718	1.045e-060	5.520e-060
f_2	opt-aiNet	90000	140	42	0.258	0.022	61960	19764	3.744e-008	5.085e-008
	CEBA	20000	39	2.38	0.099	0.195	19250	1126	3.477e-033	1.251e-032
	CDER	20000	37	1.56	0.059	0.017	19150	735	9.864e-061	4.447e-060
f_3	opt-aiNet	90000	121	52	0.248	0.019	50443	22433	5.264e-013	8.839e-013
	CEBA	20000	37	2.50	0.098	0.019	19010	1197	1.068e-030	3.964e-030
	CDER	20000	37	1.57	0.065	0.018	19000	738	2.221e-058	9.308e-058
f_4	opt-aiNet	150000	100	96	0.340	0.050	60000	20906	3.120e-005	2.649e-004
	CEBA	12500	18	0.91	0.070	0.047	1400	459	0	0
	CDER	12500	17	0.34	0.044	0.015	1360	170	0	0
f_5	opt-aiNet	90000	78	8.07	0.454	0.017	8170	4080	2.706	2.362
	CEBA	12500	20	1.59	0.059	0.016	10620	799	0	0
	CDER	12500	11	0.98	0.046	0.016	5900	479	0	0
f_6	opt-aiNet	100000	87	48	0.287	0.049	18466	3611	1.800	1.770
	CEBA	30000	24	2.42	0.179	0.030	24800	2416	0	0
	CDER	30000	12	1.28	0.103	0.018	12960	1414	0	0
f_7	opt-aiNet	108000	148	9.44	3.287	1.008	88800	22080	-9.1268	1.417e-004
	CEBA	30000	45	7.48	2.892	0.872	22500	3350	-9.7265	4.427e-005
	CDER	20000	35	5.65	1.103	0.116	17650	2656	-10.1532	9.104e-007
f_8	opt-aiNet	108000	154	11.30	3.462	0.512	92400	11030	-9.4498	1.473e-002
	CEBA	30000	43	9.24	2.761	0.312	21500	3210	-9.7472	2.638e-003
	CDER	20000	32	7.35	1.083	0.027	16475	2596	-10.4029	3.546e-005
f_9	opt-aiNet	108000	132	8.56	3.848	1.326	79200	10100	-9.0512	2.742e-004
	CEBA	30000	48	7.84	2.982	0.904	24000	3240	-9.2538	2.104e-004
	CDER	20000	36	5.5	1.205	0.112	18075	2586	-10.5364	3.203e-006
f_{10}	opt-aiNet	108000	119	27.81	0.568	0.021	68790	22626	-9.902e-01	2.310e-003
	CEBA	40000	19	16.1	0.212	0.029	20140	15727	-9.996e-01	3.705e-004
	CDER	40000	13	2.46	0.145	0.018	13540	2467	-1	0

5 Conclusion and Outlook

This paper discusses cloud droplets evolutionary model for function optimization problems after deeply researching the cloud formation in nature and two typical evolutionary paths in evolution. The model simulates the survival strategy and

evolutionary path of the population evolution with mutual transformation of vapor, liquid, solid of cloud to solve the function optimization problems. The performance of CDER algorithm is relatively better compared to opt-aiNet algorithm and CEBA algorithm. Theoretical analysis and simulation results show that the algorithm is better in global convergence, solution quality and so on.

The cloud droplets evolutionary model which combines the cloud phase with reciprocal ideas has a more wide range of applications. However, the settings of expectations, entropy and hyper-entropy have a certain impact on CDER algorithm. Thus, how to set up scientific and effective parameters is one of the key factors affecting the algorithm efficiency. In addition, the optimization problems involved in the project are mostly multi-objective optimization problems. People often want to achieve the optimal goal with a small price. For example, investment questions, people often want to put the funds as little as possible, bear minimal risk and get the maximum benefit. Therefore, these issues will be the direction for us to make a future research.

Acknowledgements. This work is supported by the National Natural Science Foundation of China (Grant No. 61073091), the Science Foundation in Shaanxi Province (Grant No. 2010JM8028), and the Special Science Research Program in Shaanxi Province (Grant No. 2010JK704).

References

1. De Yi, L., Chang Yu, L.: Study on the Universality of the Normal Cloud Model. Engineering Sciences 6, 28–33 (2004)
2. De Yi, L., Chang Yu, L., Yi, D., Xu, H.: Artificial Intelligence with Uncertainty. Journal of Software 15, 1583–1592 (2004)
3. Guang Wei, Z., Rui, H., Yu, L., De Yi, L., Gui Sheng, C.: An Evolutionary Algorithm Based on Cloud Model. Chinese Journal of Computers 31, 1082–1091 (2008)
4. Pennisi, E.: On the Origin of Cooperation. Science 325, 1196–1199 (2009)
5. Nowak, M.A.: Five Rules for the Evolution of Cooperation. Science 12, 1560–1563 (2006)
6. Thompson, J.N., Cunningham, B.M.: Geographic Structure and Dynamics of Coevolutionary Selection. Nature 417, 735–738 (2002)
7. Haldane, J.B.S.: The Causes of Evolution. Longmans Green & Co., London (1932)
8. Hamilton, W.D.: The Genetical evolution of social behaviour. Journal of Theoretical Biology 7, 17–52 (1964)
9. Axelrod, R., Hamilton, W.D.: The Evolution of Cooperation. Science 211, 1390–1396 (1981)
10. Nowak, M.A., Sigmund, K.: Evolution of Indirect Reciprocity by Image Scoring. Nature 393, 573–577 (1998)
11. Traulsen, A., Nowak, M.A.: Evolution of Cooperation by Multilevel Selection. Proceedings of the National Academy of Sciences of United States of America, 10952–10955 (2006)
12. Rahnamayan, S., Tizhoosh, H.R., Salama, M.M.A.: Opposition-Based Differential Evolution. IEEE Transactions on Evolutionary Computation 12, 64–79 (2008)
13. De Castro, L.N., Timmis, J.: An Artificial Immune Network for Multimodal Function Optimization. In: Proceedings of IEEE Congress on Evolutionary Computation, Honolulu, USA, pp. 699–704 (2002)

A Filter and Fan Based Algorithm for Slab Rehandling Problem in MPA of Steel Industry

Xu Cheng and Lixin Tang

The Logistics Institute, Northeastern University, Shenyang, 110819, China
{xcheng,lxtang}@tli.neu.edu.cn

Abstract. Slab Rehandling Problem is a new style of warehousing problem stemmed from Material Preparation Area in slab yard in steel industry which is essential to the operations efficiency of the slab yard and also coordination between continuous-casting and hot-rolling stages. SRP is to reassign storage locations of slabs in MPA under the constraint of no further shuffles during the retrieving process, with the objective of minimizing the number of rehandles in reassigning process and maximizing the empty stacks cleared up. Few literatures studied exactly the same problem as SRP. For its combinatorial and dynamic nature, a basic heuristic and a *cut strategy*-embeded filter & fan algorithm are proposed to solve it separately. Experiments on real data collected from steel industry proved the effectiveness and efficiency of the algorithm proposed. A lower bound of the problem is also proposed as a measurement of the algorithm proposed.

Keywords: Steel Industry, Slab Rehandling Problem, Filter & Fan.

1 Introduction

This paper studies the Slab Rehandling Problem (SRP) in the slab yard, which is a key logistics problem between the slab storage stage and the hot rolling mill in steel industry. In slab yard, slabs produced from continuous-casting are stored in it before sent into the furnace for hot-rolling process. Slabs which are chosen in hot rolling plan (rolling plan for short, which is a sequence of slabs with precedence relationships defined between them according to customer demands and hot-rolling rules) will be sent to MPA first and then carried onto the conveyor connected to furnace by the precedence sequence defined by rolling plan (Fig. 1). As a material preparation buffer between storage area of slab yard and the furnace, MPA plays a key role in coordinating the slab-preparation-pace and slab-heating-pace.

As slab yard receives the rolling plan, chosen slabs which always stacked in different sub-yards of the slab yard are sent to MPA. For no further shuffles when retrieving them onto the conveyor, any slab sent to MPA will be assigned to a stack which is of the same rolling plan and on top of later retrieved slabs. If no such stack exists in MPA, then open a new stack for it. Based on such stacking strategy, many unnecessary stacks are occupied in MPA. The slabs in MPA are sent into furnace one by one, according to the precedence relationships, and one plan after another. But

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 276–283, 2012.

since furnace is a kind of high energy-consumption facility in steel industry, which should be timely fed on slab and full filled for unnecessary energy consumption. So such unnecessary stacks mentioned above should be reassigned and some of them should be emptied to storage more slabs of different plans so as to avoid any vacancy in furnace. Slab Rehandling Problem (SRP) is to aggregate such unnecessary stacks into a minimum number of stacks, aiming at least rehandles in the whole rehandling process under the restriction of no further shuffling during retrieving them into furnace. A proper rehandling scheme can empty more stacks within a shorter time. And more slabs will be stored in MPA which can help match feeding frequency of furnace, decreasing energy consumption and also smooth the coordination between storage yard and hot-rolling process.

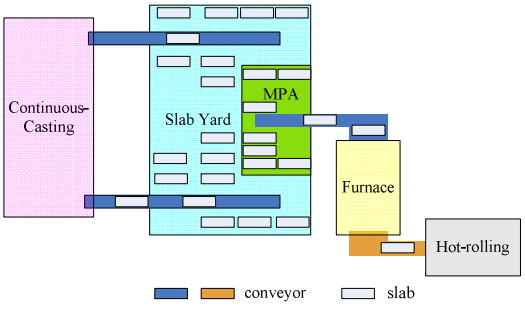


Fig. 1. Schematic Diagram of the Slab Yard

Rehandling problems have been addressed in container yard a lot. The container rehandling problem is to rehandle the export containers within the yard in advance, so that no extra re-handles will be needed during the loading operation onto the ship and the optimization goal is usually to minimize the number of container movements during the rehandling process [2,6]. Since the buffer of container yard is relatively larger than that of slab yard, operations of emptying stacks during the rehandling process is not considered in the above problem. The stacking problem in König*et al.* [1] is actually a pre-marshalling problem in which all incoming slabs are first stored in temporary stacks, and then marshaled to target stacks so that the slabs can be retrieved later from the target stacks without further shuffling. Precedence relations are defined among target stacks, a given target stack can not be disposed until its precedence target stacks having been disposed, and no shuffles permitted in target area. Slab Stack Shuffling (SSS) problem is to choose appropriate slabs for a sequence of rolling items in a hot rolling plan, from their respective candidate slab sets (families) with a view to reducing the resulting shuffling workload[3,4]. After the decision of SSS, slabs which are chosen in hot rolling plan are sent to MPA of the slab yard. And the SRP in this paper is to decide the locations of these slabs in MPA with the objective of emptying more stacks within minimum number of rehandles, under the constraints of no further shuffles during the retrieving process.

The SRP problem studied in this paper consists of three major properties distinguished from the previous warehousing or container terminal problems. (1) The rehandling object of SRP is only the slabs stored in MRP, and any rehandle must obey

the precedence relations defined between them. (2) The crane can move at most two slabs at a time. (3) SRP is a combinational optimization problem since the objective function takes both the number of shuffles and the number of emptied stacks into consideration. (4) With the huge number of possible options and later ones depending on earlier decisions, SRP has astronomical number of states during the whole rehandling decision process. Such features and complexity of SRP motivated us to develop a Filter & Fan based heuristic to solve the problem approximately.

The rest of the article is organized as follows: Section 2 describes the SRP and proposes two theorems about the properties of the optimal solution of SRP. Section 3 proposes a basic heuristic and Sections 4 presents a Filter & Fan based heuristic for SRP separately. Sections 5 gives the lower bound of the problem and Section 6 reports the experiment result on the real data collected from steel industry. Finally, Section 7 concludes the study.

2 Problem Description

In MPA, slabs are stacked according to the precedence relationships defined by rolling plan. A rolling plan includes several priority groups. The slabs which are of smaller priority index have higher priority to be retrieved, which is to say a slab indexed as 1 will be retrieved firstly. Slabs of the same priority group are always of the same width, weight and steel grade, and they can be sent to furnace at an arbitrary sequence. Slabs of the same plan are stacked together, separately from slabs of other plans. Slabs of a hot-rolling plan and the configuration of related stacks in MPA are shown as follows in Fig. 2.

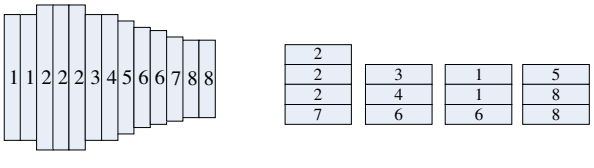


Fig. 2. A hot-rolling plan and its stacking configuration in MPA

In Fig. 2, slabs are stacked according to the priority index. For each stack in MPA, slabs of smaller priority index are stacked on higher tier which for no further shuffles occur when retrieving them on to the conveyor.

Once all slabs of a plan have been sent and stacked in MPA, none of slabs from other plans can be stacked on top of these occupying stacks, and the whole rehandling process considered in SRP is carried out in a plan. And also, from perspective of production safety in practice, aggregation process should be finished before retrieving process. Any slab can not be sent to furnace until the rehandling process is over. So any dynamic sent-in or sent-out slab is not in consideration of this paper.

For illustrating the rehandling (aggregation) process, an example of 4 stacks configuration is given in Fig. 3. There are 14 slabs staked in 4 stacks, each slab represented by a square. The containers are retrieved in ascending order of their priority index marked on the squares. The series of Figs. 3(a)–3(e) gives one way to rehandle the stacks. There are 6 rehandles: slab 3 and 4 from stack b to stack d

(Fig. 3a to 3b), two slab 2 from stack a to stack d, one slab 2 from stack a to stack d, two slab 1 from stack c to stack d (Fig. 3b to 3d), and last twice rehandles for slab 6 from stack b and c to stack a separately (Fig. 3d to 3e). Certainly, the initial configuration and the scale of priority index both affect the total number of rehandles and final vacancy stacks move out. Given an initial configuration, different moves will definitely bring out different final configurations, different number of vacancy stacks and number of rehandles.

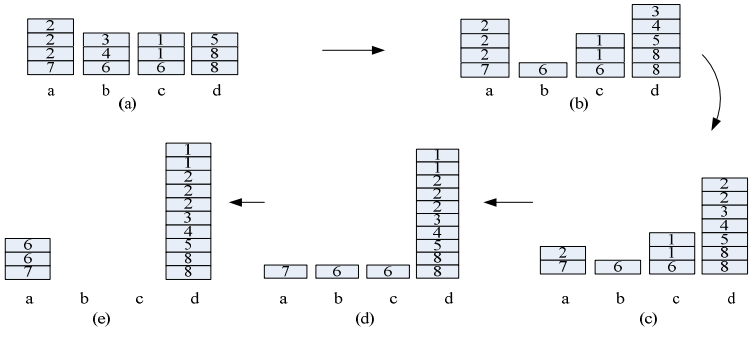


Fig. 3. A rehandling process of SRP

For a given configuration of MPA, Slab Rehandling Problem (SRP) is to aggregate the underused stacks, aiming at the minimum number of stacks finally taken and also the number of rehandles during the whole rehandling process. Each rehandle movement is operated by crane. For given initial and final configurations, the crane which can hold at most two slabs once will finish the rehandles more quickly than the crane which can only hold one slab at a time. But it also makes the SRP more difficult since the number of slabs that the crane should carry on for each movement becomes decision variables too.

Three assumptions mentioned below generally hold during the operations of rehandling moves in SRP.

Assumption 1: Restrictions on stacks that the rehandled slabs moved on: any rehandled slab should be stacked on top of later departing ones. In other words, in a stack, the higher slab ranks smaller than any lower one. Based on such restriction, we would like to assign the rehandled slabs to the best storage slots such that the possibilities of more stacks would be emptied is maximized.

Assumption 2: Restrictions on two slabs moved together by crane: two slabs can be moved together by crane iff they have the same source stack and destination stack, and the below one should rank smaller than the slab which is on top of the destination stack. This assumption ensures no more shuffles occur.

Assumption 3: Restrictions on the emptied stacks: since empty stacks in MPA is precious and will be more useful for other rolling plans to take, any rehandling move should not occupy any vacancy stack, even the new emptied ones.

As innocent as the problem posed, the problem is hard for its astronomical number of states. With the huge number of possible options and later ones depending on earlier

decisions, even the modeling of the stack emptying process is hard. The objective function is as follows:

$$\min \textit{stacks}+1/k\ \textit{rehandles}. \tag{1}$$

where *stacks* denotes the number of stacks in the final configuration of MPA, *rehandles* denotes the number of rehandles taken from the initial configuration to the optimized final configuration (the whole rehandling process) . k is a parameter, $k \in Z^+$, which can be set according to different practical requirement. Since the main purpose of SRP is to aggregate existing stacks, resulting in more empty stacks for other plans to take, then item of *stacks* is more important to its objective function. The introduction of rehandles is to keep rehandling process in a reasonable time.

Based on descriptions above, two theorems are proposed for the optimal solution of SRP.

Theorem 1: if there are n consecutive slabs of the same original stack and about to move to the same destination stack. Then in the optimization solution, such movements should be rehandled within $\lceil n/2 \rceil$ times.

Proof: for n consecutive moves, minimum rehandles occur only if the crane takes two slabs in each movement until no or one slab left, which is $\lceil n/2 \rceil$ times. □

Theorem 2: if a stack contains all the slabs which marked with the highest index, in the optimization solution such stack could not be emptied.

Proof: Because of the precedence relationships defined in stacking process, any slab must stack on the one whose index is no smaller than it. Then slabs with the highest index cannot stack on any slabs except the slab of the same index. Since all the slabs of the highest index are in a same stack and such slabs cannot move on other stacks, so such stack cannot be emptied. □

3 A Basic Heuristic

Consider *S(1,2,…j…,S)* slabs stored in *D(1,2,…i…,D)* stacks. p_j is priority index of slab j and h_i is height of stack i. Without loss of generality, slabs are ranked from 1 to r ($p_j \in \{1,2,\ldots,r\}$, since different slabs may be of the same priority, so $r \leq S$) with smaller rank retrieved earlier. Define C as the set of stacks in which top one or two slabs can be moved onto other stacks. See Fig. 4 as an example.

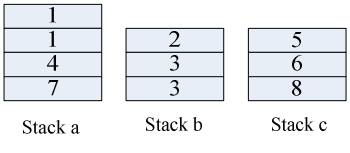


Fig. 4. An example of configuration in MPA

Top slab of stack a is indexed as 1, the highest priority, can be moved to any top of other stacks. The top slabs of stack b is indexed as 2 or 3, can be moved to stack c since 2 or 3 is less than 5. Top slabs of tack c is indexed as 5 or 6, larger than any

other stacks, so stack c cannot be moved to any other stacks. Then $C=\{a,b\}$ and $c \notin C$. The process of the heuristic method is followed:

Step 1: Initialization. Set *Iteration* = 0, *max_Interation*.

Step 2: Feasible rehandles check. Check C, if $C = \varnothing$ there is no feasible rehandle can be taken, go to *Step e*; else index the stacks in descending order of $P_i = \sum_{j \in \tau_i} p_j \Big/ h_i, i \in C$, ($\tau_i$ is the slab set of stack i).

Step 3: Origination-stack decision. Choose stack $i^{ori} = \left\{ i \left| P_i = \max_{i \in C} \sum_{j \in \tau_i} p_j \Big/ h_i, i \in C \right. \right\}$. If more than one stack have the same max value, then choose the one with bigger priority index of the top slab. If more than one stack can be chosen, then choose an arbitrarily stack among them.

Step 4: Destination-stack decision. Evaluate candidate stacks $i^{des} \in C \backslash i^{'}$ by $p^{des} - p^{ori} + |h_{max} - h_i^{des'}|$, in which p^{des} denotes the index of the top slab of i^{des}, p^{ori} denotes the larger index of slabs of the origination stack, and $h_i^{des'}$ the updated height of i^{des}. Choose the stack with the minimum evaluation function value and take the move (a rehandle movement includes an origination stack, a destination stack and also the number of slabs should be moved), update the configuration of MPA, *iteration* = *iteration* + 1;

Step 5: If *Iteration* < *max_Interation*, go to *Step b*;

Step 6: Stop.

4 A Filter and Fan-Based Algorithm

Since a feasible solution of SRP is a sequence of moves, and later scenarios depending on earlier scenarios, which resulting in an astronomical number of states. A filter & Fan (FF) approach is developed to explore the neighborhood space for finding a near-optimal solution. The F&F model can be illustrated by means of a neighborhood tree where branches represent submoves and nodes identify solutions produced by these moves. The neighborhood tree is explored breadth-first, and then depth. We associate each node of neighborhood tree with a possible configuration of MPA, and define the neighborhood of a node as the set of all configurations that can be constructed from this node by one feasible move. Suppose S(0) (scenario 0) is the initial configuration of the stacks in MPA, S(1) are the set of *n1* best scenarios based on S(0) after taking a move. For each node in S(1), *n2* best nodes S(2) are then generated by taking a move based on scenario S(1). Then the best *n1* scenarios from *n1*n2* nodes are selected for the following generation process. Figure 6 shows an example of FF process.

The method incorporates two fundamental components: a *local search* to identify a local optimum and a *filter and fan search* to explore larger neighborhoods in order to overcome local optimality. Any time a new local optimum is found in one search strategy the method switches to the other strategy and keeps alternating this way until the *filter and fan search* fails to improve the current best solution.

For the astronomical number of states in the neighborhood tree, a *cut strategy* is proposed to accelerate the search process. Based on the solution gained from basic heuristic, any node in the neighborhood satisfies any of the following condition will be cut off:

1. For the same number of emptied stacks, the node which takes more than l_1 ($l_1 > 1$) times moves of initial solution will be removed;
2. For the same number of rehandles, the node which empties less than l_2 ($l_2 = 1, 2, \ldots$) stacks of the initial solution emptied.

Cut strategy cuts the descendants of such nodes mentioned above and keep the tree an appropriate size and also accelerate finding a near-optimal solution as a result.

5 A Lower Bound of SRP

A lower bound on the optimal objective function value of SRP is proposed as follows. For S slabs in MPA, the minimum number of stacks occupied in the final configuration is $\lceil S/h_{\max} \rceil$, and coordinately, the minimum number of reshuffles for this final configuration equals to empty the $q = M - \lceil S/h_{\max} \rceil$ lowest stacks. Suppose the height of q stacks is $h'_1, h'_2, ..., h'_q$ separately, then the minimum shuffles is $\sum_{i=1}^{q} \lceil h'_i/2 \rceil$. The lower bound of SRP Z^{low} is as follows:

$$Z^{low} = \lceil S/h_{\max} \rceil + 1/k \sum_{i=1}^{q} \lceil h'_i/2 \rceil \tag{2}$$

It is worth to mention that the initial configuration of MPA has a great effect on both the performance of the algorithm proposed and also lower bound. In most conditions, this lower bound is too tight to be achieved only if an MPA has an initial configuration that all slabs can be moved onto first $\lceil S/h_{\max} \rceil$ highest stacks directly without shuffling.

6 Computational Experiments

Experiments on real data collected from Steel Enterprise are tested on an Intel Dual Core of 2.5 G Hz and 1.94G RAM. The experiment results are shown in table 1.

Table 1. Computational Experiments ($k = 8$, $l_1 = 1.2$, $l_2 = 2$)

Index	Basic Heuristic	FF	Lower Bound	Improvement (%) (BH_FF)	Gap (%) (FF_LB)	CPU Time(s) (FF)
1	1.750	1.750	1.75	0	0	1.43
2	2.250	1.375	1.25	38.8	12.5	0.23
3	4.375	3.000	2.25	31.4	33.3	2.74
4	6.500	4.375	3.50	32.7	25.0	2.03
5	8.125	6.750	3.75	16.9	80.0	5.18
Average	4.600	3.450	2.50	25.0	30.16	2.322

Experiment results show that FF algorithm makes a relatively large improvement comparing to the basic heuristic by 25% of SRP. Inevitably, FF algorithm takes a relatively longer runtime for 2.322 seconds, compared to average running time on basic heuristic (0.1s). Actually, time consuming on a movement taken by crane is much longer (2 or 3 minutes) than that of spent on FF algorithm, so FF algorithm is acceptable in practice.

Based on the experiment results above, both the effectiveness of the basic heuristic algorithm and the lower bound are affected by the initial configuration of MPA quite a lot. Except for test index 1, lower bounds for other 4 tests can not be achieved for some realistic reasons, such as no enough slot for rehandling, given stack can not be emptied and so on. Initial configuration-oriented algorithm should proposed to solve SRP more effectively and of better pertinence.

7 Conclusions

SRP is originally stemmed from MPA of slab yard in steel industry which is to accumulate stacks in MPA under the constraint of precedence relationships defined among them and also no further rehandles during the retrieving process, aiming at maximizing the empty stacks cleared up and minimizing the rehandles. For its astronomical number of states, combinatorial and dynamic nature, a basic heuristic and a FF based algorithm are proposed to solve it separately. In FF based algorithm, a *cut strategy* is developed to keep the neighborhood tree an appropriate size and also accelerate the search process. The experiment results with practical data show that solutions of FF have an improvement of 25% comparing the solutions obtained by basic heuristic on average.

Acknowledgments. This research is partly supported by State Key Program of National Natural Science Foundation of China (71032004), the Fundamental Research Funds for the Central Universities (N090104002, N100704002).

References

1. König, F.G., Lübbecke, M., Möhring, R., Schäfer, G., Spenke, I.: Solutions to Real-World Instances of PSPACE-Complete Stacking. In: Arge, L., Hoffmann, M., Welzl, E. (eds.) ESA 2007. LNCS, vol. 4698, pp. 729–740. Springer, Heidelberg (2007)
2. Wan, Y.W., Liu, J.Y., Tsai, P.C.: The Assignment of storage locations to containers for a container stack. Naval Research Logistics 56, 699–713 (2009)
3. Tang, L.X., Liu, J.Y., Rong, A.Y., Yang, Z.H.: Modelling and a genetic algorithm solution for the slab stack shuffing problem when implementing steel rolling schedules. IJPR 40, 1583–1595 (2002)
4. Tang, L.X., Ren, H.Z.: Modelling and a segmented dynamic programming-based heuristic approach for the slab stack shuffling problem. Computers & Operations Research 37, 368–375 (2010)
5. Greistorfer, P., Rego, C.: A simple filter-and-fan approach to the facility location problem. Computers & Operations Research 33, 2590–2601 (2006)
6. Kim, K.H., Hong, G.P.: A heuristic rule for relocating blocks. Computers & Operations Research 33, 940–954 (2006)

An Improved Artificial Immune Recognition System Based on the Average Scatter Matrix Trace Criterion

Xiaoyang Fu[1] and Shuqing Zhang[2]

[1] Department of Computer Science and Technology, Zhuhai College of Jilin University, Zhuhai 519041, China
dvndavidfu@vip.163.com
[2] Northeast Institute of Geography and Agroecology, Chinese Academy of Sciences, Changchun 130012, China

Abstract. This paper proposed an improved artificial immune recognition system (IAIRS) based on the average scatter matrix trace (ASMT) criterion. In essence, the artificial immune recognition system (AIRS) is an evolving algorithm. Through clonal expansion, affinity maturation, resource competition and immune memory etc, a set of new samples (memory cells) is produced. The ASMT of memory cells will be decreased and the minimized ASMT can be as the optimal criterion of AIRS. The IAIRS algorithm is demonstrated on a number of benchmark data sets effectively.

Keywords: artificial immune recognition system, scatter matrix trace, pattern classification.

1 Introduction

In 2001 year, Timmis at al. [1] proposed a resource limited artificial immune system (AIRS) based on clonal selected theory adopts the conception of artificial recognition balls (ARBs). Comparison with linear and nonlinear classifiers, it was be demonstrated on a number of benchmark data sets effectively [2-3]. It has shown to be successful for the area of remote sensing [4]. Its basic idea can be developed to evolve multiplayer neural networks [5].

In essence, AIRS is an evolving algorithm. Through clonal expansion, affinity maturation, resource competition and immune memory etc., a set of new samples: memory cells (MC) will be produced. Replacing the training samples with MC, the data will be classified more effectively by k-NN algorithm. From a machine learning point of view, AIRS algorithm is what provides for the data reduction capabilities and generalizations since the evolved memory cells in the system are not necessarily identical to any training samples.

Scatter matrix trace (SMT) is one of the simplest and most widely used criterions for clustering [6]. From a geometry point of view, vectors in feature space draw from a normal population tend to fall in a single cloud. The within-cluster scatter matrix is used to measure the compactness of these clouds. The smaller the scatter matrix trace, the higher the density of the clouds and the better the classification performance.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 284–290, 2012.

This paper proposed an improved artificial immune recognition system (IAIRS) based on the average scatter matrix trace (ASMT) criterion. The ASMT value of evolved memory cells will be decreased through training and the minimized ASMT can be as the optimal criterion of IAIRS. IAIRS algorithm is demonstrated on a number of benchmark data sets effectively.

2 AIRS Algorithm

AIRS algorithm [2][4] includes the following five steps:

2.1 Normalization and Initialization

All training samples (antigens, *ags*) are firstly normalization such that the distances among them are in the range [0, 1]. Secondly, calculate the affinity threshold (AT) which is the average affinity over-all training samples.

$$AT = \frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} affinity\ (ag_i, ag_j)}{n(n-1)/2} \tag{1}$$

Where *n* is the number of all training samples.The final step is the seeding of memory cells and initial ARB population. This is performed by randomly choosing training antigens to be added to the set of memory cells (MC) and to the set of ARBs (AB).

2.2 ARB Generation

1. Find mc_{match}
   mc_{match}=*arg max stimulation (ag,mc), mc*∈ *MC,* mc_{match}*.c=ag.c*
   Where *stimulation(x,y)=1-affinity(x,y).*
2. Hyper clonal expansion

The mc_{match} generates new *ARBs* to place into the *AB* set by hyper clonal expansion.The number of hyper clones, *NClones* is defined as:

*NClones=hyperClonalRate*clonalRate*stimulation(ag,* mc_{match}*)*

2.3 Competition for Resources and Nomination of Candidate Memory Cell

2.3.1 Normalization ARBs Stimulation Level and Calculation the Resources

1. Find the maximum stimulation *max.stim* and minimum stimulation *min.stim* among all the ARBs.
2. For each $ab \in AB$, normalize its stimulation, *ab.stim.*
3. For each $ab \in AB$, calculate *ab's* resources, *ab.resources* and the resources of all ARBs, *ResAlloc*.
4. Metadynamics of ARBs

Comparing *ResAlloc* with *TotalNumRes* which is defined as the total numbers of resources allowed, if *ResAlloc*>*TotalNumRes*, then resources are removed from the weakest ARBs until *ResAlloc* in the system returns to *TotalNumRes*.

2.3.2 Stopping Criterion for Training Procedure

Calculate the average stimulation level for each ARB class group, S_i. If each $S_i(i=1,2,\cdots c)$ is less than a given stimulation threshold (*ST*), the process moves to step 2.3.3, otherwise, jump to step 2.3.5.

2.3.3 Clonal Expansion and Affinity Maturation

For each $ab \in AB$, allow each *ab* in AB the opportunity to produce mutated offspring. The number of clones, *NClones=clonalRate*stimulation(ag,ab).*

2.3.4 Re-judging Stopping Criterion

Calculate each S_i, if these are less than *ST*, the process repeats from step 2.3.1 until the stopping criterion is met.

2.3.5 Developing the Candidate Memory Cell

Select the highest affinity *ab* of the same class as the training *ag* from AB set, as candidate memory cell, $mc_{candidate}$.

2.4 Evolving MC Pool

$$\text{If } stimulation(ag, mc_{candidate}) > stimulation(ag, mc_{match}) \quad \text{then } MC \leftarrow MC + mc_{candidate} \tag{2}$$

$$\text{If } affinity(mc_{match}, mc_{candidate}) < AT * ATS \quad \text{then } MC \leftarrow MC - mc_{match} \tag{3}$$

Equation (2) is the necessary condition for $mc_{candidate}$ to be added into MC pool, if both equation (2) and equation (3) are satisfied, the mc_{match} shall be replaced by the $mc_{candidate}$.

The training on this particular *ag* is completed until now. The next *ag* in the training set is selected and the training process proceeds from step 2.2 to step 2.4. This process continues until all *ags* have been trained in the proposed algorithm.

2.5 Classification

Replacing training samples with MC set, the tested samples can be classified by k-NN algorithm.

3 Average Scatter Matrix Trace (ASMT) Criterion

Let vectors in the feature space *D* are divided into c subsets. D={$D_1 \cup D_2 \ldots \cup D_c$}

The scatter matrix S_i of the i^{th} class feature vectors is defined as:

$$S_i = \sum_{x \in D_i} (x - m_i)(x - m_i)^t \tag{4}$$

Where m_i is the mean vector of the i^{th} class vector *x*

Within cluster scatter matrix is defined as:

$$S_w = \sum_{i=1}^{c} S_i \tag{5}$$

The simplest scalar measure of a scatter matrix is its trace (the sum of its diagonal elements), and definition of scatter matrices (Eqs.4 and 5) yield the equation (6) as follows:

$$t_r S_w = \sum_{i=1}^{c} t_r S_i = \sum_{i=1}^{c} \sum_{x \in D_i} \left\| x - m_i \right\|^2 = J_e \tag{6}$$

Thus the Scatter matrix trace criterion is nothing more or less than the sum of squared-error criterion.

Because of the number of memory cells is usually changed for different training, it is necessary to introduce the average scatter matrix trace. For each class group *i*, $\overline{S}_i = t_r S_i / n_i$, Where n_i is the number of the i^{th} class mc cells.

The total average scatter matrix trace $\overline{S}$ is defined as $\overline{S} = \sum_{i=1}^{c} \overline{S}_i$

The essence of AIRS algorithm is the process of evolving memory cells from *ARBs* and replacing the training samples with MC. The evolving algorithm results in more density of feature vectors in MC so that $\overline{S}$ is decreased. Thus the minimized ASMT can be as the optimum criterion of AIRS.

The IAIRS algorithm has been improved as follows:

1. In step 2.1, replacing *AT* with AT_i

$$AT_i = \frac{\sum_{j=1}^{n_{i-1}} \sum_{k=j+1}^{n_i} affinity \quad (ag_j, ag_k)}{n_i (n_i - 1) / 2} \tag{7}$$

Where n_i is the number of the i^{th} class antigens, i =1, 2 … c.

2. In step 2.4, replacing *AT* with AT_i which has the same class as the training *ag*.

3. Let m represent the number of epochs, ATS_0 represents *ATS* initial value, ΔATS represents an increment of *ATS*, θ represents criterion threshold. Doing step 2.1 to step 2.4 represent an evolving epoch. After an epoch, calculate $\overline{S}$, $\Delta \overline{S} = \overline{S}_{m+1} - \overline{S}_m$, and $ATS_{m+1} = ATS_m + \Delta ATS$. Repeat step 2.1 to step 2.5 until stopping criterion of multi-training $\Delta \overline{S} < \theta$ is met.

4. Find the best MC from multi-loop training. The best MC has both relatively good accuracy and data reduction capability.

4 Experiments and Discussion

Iris, Ionosphere and Diabetes data are come from website (http://archive.ics.edu/ml) Vowel data are come from MLT Lincoln Laboratory (http: //www.ll.mit.edu/IST/ lnknet). Their main characters of the data organized as following table 1.

Table 1. The main characters for Iris, Ionosphere, Diabetes and vowel

Data sets	Size of samples	classes	feature dimensions
Iris	150	3	4
Ionosphere	350	2	34
Diabetes	750	2	8
Vowel	300	10	2

According to m-fold cross-validation, all samples are randomly divided into five sets. The classifier is trained five times, each time with different set hold out as a validation set and other sets for training. The test accuracy is an average of five runs.

In AIRS algorithm, *hyperClonalRate*=2, *clonalRate*=10, *mutationRate*=0.1, *TotalNumRes*=200. *ST*=0.6~0.9, *ATS*=0.1~0.6, *k*=3~7 (depends on classified data).

In IAIRS algorithm, θ=0.001, ATS_0=0.1, ΔATS =0.1, other parameters are the same as AIRS.

Table 2 is the performance comparison of classifiers for Iris, Ionosphere, Diabetes and Vowel.

Table 2. The performance comparison of classifiers for Iris, Ionosphere, Diabetes and Vowel

Data sets	AIRS		IAIRS	
	Accuracy	Memory cells	Accuracy	Memory cells
Iris	95.4	60.8/(43%)	96.0	64.4/(46%)
Ionosphere	84.0	104.0/(63%)	85.7	132.0/(53%)
Diabetes	71.6	349.6/(42%)	72.6	192.4/(68%)
Vowel	67.7	113.4/(53%)	68.6	123.4/(49%)

Where accuracy (allscore)=((*n-miss)/n*)*100%, n is the number of tested samples, miss is the number of misclassified points.

As seen from Table 2, the overall classification accuracy of IAIRS is better than AIRS for Iris, Ionosphere, Diabetes and Vowel. In data reduction capabilities, the IAIRS matches the AIRS comparatively.

Comparing IAIRS with AIRS, the evolving mechanism adopting artificial immune algorithm is the same. Their different is the determinant criterion Equation (3). The IAIRS algorithm employs AT_i which has the same class as the training *ag*, not *AT* which is calculated overall training *ags*. It results in more reasonable $mc_{candidate}$

replacing mc_{match} to be added into MC pool. On the other hand, the criterion (Eq.3) is affected by the product of *ATS* and *AT*. The larger the product, the higher the opportunity of $mc_{candidate}$ replacing mc_{match} , the stronger the data reduction capabilities while the generalization would be worse probably. Through multi-training which the *ATS* is adjusted automatically, IAIRS can find the best MC individual with optimal equilibrium between data reduction capabilities and system classified accuracy.

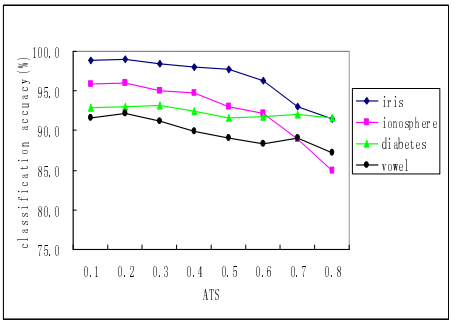


Fig. 1. The affect of altering ATS on accuracy

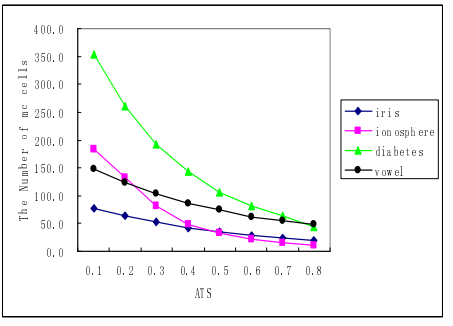


Fig. 2. The affect of altering ATS on data reduction

Fig.1and Fig.2 show the affect of altering *ATS* on classification accuracy and data reduction in IAIRS respectively. The test results that the number of mc cells is descent monotonously with increase of the *ATS* in the training epochs are consistent with analysis above. The tested accuracy of the four data changes relatively small while *ATS* changes from 0.1 to 0.5. Observed from the both figures, the evolving MC can get the best equilibrium between data reduction capabilities and system generalization while *ATS* is ranged from 0.3 to 0.5.

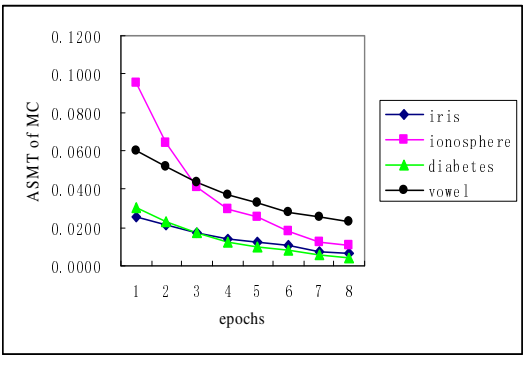


Fig. 3. The change of ASMT on different epoch

As seen from Fig. 3, the value of ASMT is descent monotonously with increase of the epochs for all data as well as the number of mc cells. The minimum of ASMT is consistent with the classifier's performance improvement(classified accuracy and data reduction capability) in the range of 1 to 5 epochs. Thus, it is reasonable that the

minimum of ASMT can be regarded as the stopping criterion of the multi-training. After 6^{th} epoch, though the value of ASMT is still descent, and the data reduction capability is stronger, but the classification accuracy is getting worse because the SMT criterion is assumed that the samples form c groups fairly well separated clouds of points, which result in limitation of using SMT criterion. [6].

5 Conclusion

This paper proposed an improved artificial immune recognition system (IAIRS) based on the average scatter matrix trace (ASMT) criterion. The IAIRS algorithm modifies the criterion employed by AIRS which the $mc_{candidate}$ replacing mc_{match} into MC pool. The improved algorithm, which takes multi-training that the ATS is modified automatically instead of one shoot for all antigens by AIRS, employs the minimum average scatter matrix trace as a stopping criterion while the variation of classification accuracy is in a reasonable range. The IAIRS algorithm can find the best MC individual which replaces the training samples and gets the better classification performance. IAIRS algorithm has been demonstrated on the benchmarked data effectively. The tested results show that the IAIRS possesses better classification accuracy and data reduction capabilities.

Acknowledgments. Project supported by the Joint Research Fund for Chinese Academy of Sciences with Guangdong Province, China (Grant No. 2009B091300149).

References

1. Timmis, J., Neal, M.: A Resource limited Artificial Immune System. Knowledge Based Systems 14(3/4), 121–130 (2001)
2. Watkins, A., Boggess, L.: A New Classifier based on Resource Limited Artificial Immune System. In: Ebherhart, R. (ed.) Congress on Evolutionary Computation. Part of the World Congress on Computational Intelligence, Honoluu, HI, pp. 1546–1551. IEEE, Piscataway (2002)
3. Watkins, A., Timmis, J.: Artificial Immune Recognition System (AIRS): An Immune-Inspired Supervised Learning Algorithm. Kluwer Academic Publisher, Netherland (2003)
4. Zhang, L., Zhong, Y., Huang, B., Li, P.: A Resource Limited Artificial Immune System Algorithm for Supervised Classification of Multi/hyper-spectral Remote Sensing Imagery. International Journal of Remote Sensing 28(7-8), 1665–1686 (2007)
5. Fu, X., Zhang, S., Pang, Z.: A Resource Limited Immune Approach for Evolving Architecture and Weights of Multilayer Neural Network. In: Tan, Y., Shi, Y., Tan, K.C. (eds.) ICSI 2010, Part I. LNCS, vol. 6145, pp. 328–337. Springer, Heidelberg (2010)
6. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. John Wiley & Sons, Beijing (2001) ISBN:0-471-05669-3

A Danger Feature Based Negative Selection Algorithm

Pengtao Zhang and Ying Tan

Key Laboratory of Machine Perception (MOE), Peking University
Department of Machine Intelligence,
School of Electronics Engineering and Computer Science,
Peking University, Beijing, 100871, China
ytan@pku.edu.cn

Abstract. This paper proposes a danger feature based negative selection algorithm (DFNSA). The DFNSA divides the danger feature space into four parts, and reserves the information of danger features to the utmost extent, laying a good foundation for measuring the danger of a sample. In order to incorporate the DFNSA into the procedure of malware detection, a DFNSA-based malware detection (DFNSA-MD) model is proposed. It maps a sample into the whole danger feature space by using the DFNSA. The danger of a sample is measured precisely in this way and used to classify the sample. Eight groups of experiments on three public malware datasets are exploited to evaluate the effectiveness of the proposed DFNSA-MD model using cross validation. Comprehensive experimental results suggest that the DFNSA is able to reserve as much information of danger features as possible, and the DFNSA-MD model is effective to detect unseen malware. It outperforms the traditional negative selection algorithm based and the negative selection algorithm with penalty factor based malware detection models in all the experiments for about 5.34% and 0.67% on average, respectively.

Keywords: danger feature, negative selection algorithm, feature extraction, malware detection, artificial immune system.

1 Introduction

With the development of immunology, more and more immune mechanisms have begun to be applied in computer security. Forrest et al. first proposed a negative selection algorithm (NSA) to detect the abnormal modification on protected data [1] and later to monitor the UNIX process [2]. Furthermore, they proposed some design principles for computer immune system, such as anomaly detection, diversity and adaptability [3].

The traditional NSA (TNSA) generates a detecting feature library, in which any feature does not match any self, by deleting all the features matching self. It assumes that all the self are harmless and all the non-self are harmful. However, some self are harmful, for example, cancer cells, and some non-self are harmless, taking food as an example.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 291–299, 2012.

In order to overcome the drawback of the TNSA in defining the harmfulness of self and non-self, the danger theory (DT) was proposed [4]. According to the DT, the immune system reacts to danger, instead of reacting to non-self, and the internal conversation between tissues and the cells of the immune system controls immunity. The DT explains the autoimmune reaction perfectly.

Based on the DT, Aickelin et al. proposed the danger zone to translate the DT into the field of computer security [5]. From then on, many artificial immune models are proposed, for details, please refer to [6] [7] [8] [9] [10].

Pengtao Zhang et al. proposed a negative selection algorithm with penalty factor (NSAPF) based malware detection (NSAPF-MD) model [11]. The detecting feature library of this model consists of all the non-self danger features, where the features matching self are punished using a penalty factor. It performed well in their experiments. However, this model needs to select a proper penalty factor. What is more, it merely takes advantage of non-self danger features, instead of all the danger features extracted in a training set.

In this paper, a danger feature based negative selection algorithm (DFNSA) is proposed, which reserves the information of danger features to the utmost extent, laying a good foundation for measuring the danger of a sample. On this basis, a DFNSA-based malware detection (DFNSA-MD) model is proposed. It makes use of all the danger features and maps a sample into the whole danger feature space by using the DFNSA. In this way, the proposed DFNSA-MD model measures the danger of a sample precisely and archives good performance.

The remainder of the paper is organized as follows. In Section 2, we introduce the DFNSA. In Section 3, the DFNSA-MD model is presented in detail. Section 4 gives the detailed experimental setup and results. Finally, we conclude the paper with a detailed discussion.

2 Danger Feature Based Negative Selection Algorithm

2.1 Danger Feature

Definition: A danger feature is a feature with dangerous properties, which are able to identify its corresponding dangerous operations. It is the basic element for an immune system to decide whether an immune response should be produced.

In the malware detection field, a danger feature is a code segment which executes a dangerous operation, such as formatting diskette, self-replicating.

There are many expressions for a danger feature. For example, we could use binary string, sequences of assembly codes to express a danger feature in the malware detection field. Generally speaking, a danger feature could appear in both non-self and self. It is the foundation of measuring the danger of a sample.

The danger features can be classified into four categories: (1) danger features only appearing in non-self; (2) danger features appearing in both non-self and self, but tending to appear in non-self; (3) danger features appearing in both non-self and self, but tending to appear in self; (4) danger features merely appearing in self.

2.2 DFNSA

The flow chart of the DFNSA is shown in Fig. 1, where the NCDFL denotes the non-self candidate danger feature library, which is taken as non-self, and the SCDFL means the self candidate danger feature library, which is self.

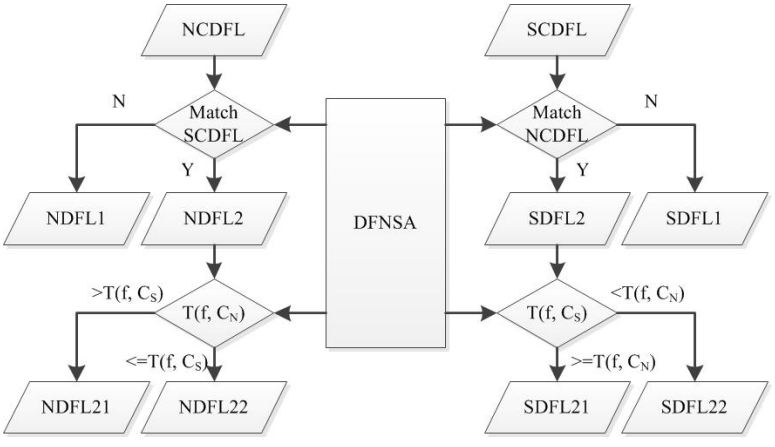


Fig. 1. The flow chart of the DFNSA

Based on the matching of non-self and self, the DFNSA splits the non-self features, which do not match any self, into the non-self danger feature library 1 (NDFL1), and the other non-self features, which match self, into the NDFL2. According to the class tendency of danger features, the NDFL2 is further divided into the NDFL21 and NDFL22, in which the features tend to appear in non-self and self, respectively. The features in the NDFL22 are extracted from non-self, but tend to appear in self, so they are considered to be invalid and deleted.

The measure of the class tendency of a feature is defined as $T(f, C) = P(f, C)$, where $P(f, C)$ denotes the proportion of feature f appearing in class C. if $T(f, C_N) > T(f, C_S)$, then f is considered to tend to appear in non-self, otherwise self. The C_N and C_S denote the classes of non-self and self, respectively.

In a similar way, the SCDFL is firstly split into the SDFL1 and SDFL2 by the DFNSA. Then the SDFL2 is further divided into the SDFL21 and SDFL22. The SDFL22 is deleted with the same reason as the NDFL22.

Definition: if a danger feature f_1 matches a danger feature f_2, the two features are equivalent to each other, written as $f_1 = f_2$.

According to the above definition, since NDFL21 = SDFL22 and NDFL22 = SDFL21, deleting the NDFL22 and SDFL22 merely deletes redundant information, without losing any information of danger features. The proof about NDFL21 = SDFL22 and NDFL22 = SDFL21 is given below.

Proof. $\because \forall f_S \in SDFL2, \exists f_N \in NDFL2, f_S = f_N$
$\therefore T(f_S, C_S) = T(f_N, C_S) = P(f_S, C_S)$ and $T(f_S, C_N) = T(f_N, C_N) = P(f_N, C_N)$
$\therefore$ if $T(f_S, C_S) >= T(f_S, C_N)$, then $f_S \in SDFL21$ and $f_N \in NDFL22$,
if $T(f_S, C_S) < T(f_S, C_N)$, then $f_S \in SDFL22$ and $f_N \in NDFL21$
$\therefore \forall f_S^{'} \in SDFL21, \exists f_N^{'} \in NDFL22, f_S^{'} = f_N^{'}$
and $\forall f_S^{'} \in SDFL22, \exists f_N^{'} \in NDFL21, f_S^{'} = f_N^{'}$,
Similarly $, \forall f_N^{'} \in NDFL21, \exists f_S^{'} \in SDFL22, f_N^{'} = f_S^{'}$
and $\forall f_N^{'} \in NDFL22, \exists f_S^{'} \in SDFL21, f_N^{'} = f_S^{'}$
$\therefore NDFL21 = SDFL22, NDFL22 = SDFL21$

The DFNSA divides the danger feature space into four parts, and reserves the information of danger features to the utmost extent, laying a good foundation for measuring the danger of a sample. The four categories of danger features are stored in the NDFL1, NDFL21, SDFL21 and SDFL1, respectively.

Comparing to the NSAPF, the DFNSA does not need to optimize a penalty factor, dramatically dropping down the training time of the DFNSA, and takes full advantage of all the danger features extracted in a training set.

3 DFNSA-Based Malware Detection Model

In this paper, a danger feature is defined as a code segment which executes a dangerous operation, and expressed as a binary string. The malware and benign programs are taken as non-self and self, respectively.

3.1 Danger Feature Extraction

Malware Instruction Library. This paper defines an instruction as a binary string of length 2 bytes. The class tendency of an instruction i to malware is measured using Eq. 1. The top $P\%$ instructions with the highest tendency value make up the malware instruction library (MIL).

$$I^i = \frac{I_n^i/I_n}{I_n^i/I_n + I_s^i/I_s}, F^i = \frac{F_n^i/F_n}{F_n^i/F_n + F_s^i/F_s}, T^i = \sqrt{(I^i)^2 + (F^i)^2} \tag{1}$$

where I_n^i and I_s^i denote the instruction frequencies of an instruction i in non-self and self, respectively, and F_n^i and F_s^i are the document frequencies of i in non-self and self. I_n and I_s indicate the number of instructions in the non-self and self, respectively. F_n and F_s are the number of samples in non-self and self. I^i and F^i measure the tendency of i to the non-self in the perspectives of instruction frequency and document frequency. T^i is the tendency of i to the non-self.

Since the instructions in the MIL tend to appear in malware, they are dangerous. If the length of a binary string constructed by these instructions exceeds a threshold R bytes, we believe the binary string contains enough danger information and is a danger feature. All the danger features make up the danger feature space.

NCDFL and SCDFL. On the basis of the MIL, the NCDFL and SCDFL are generated by traversing all the malware and benign programs in a training set, respectively. The way to traverse a sample is described below.

A sliding window of length 2 bytes is used to traverse a sample to extract candidate danger features. It moves forward 1 byte at a time. When the window encounters an instruction contained in the MIL, it begins to generate a feature. If the instructions in two adjacent windows do not belong to the MIL, the current feature is terminated as the next feature would not connect with it. If the length of the current feature exceeds R bytes, it is taken as a candidate danger feature. The sliding window keeps on moving to the end of the sample.

This paper sets $R = 4$. The length of a candidate danger feature would be adjusted based on the specific sample and MIL as described above, so R would not affect the result significantly. The frequency of a feature is taken as its weight.

Detecting Feature Library. Taking the NCDFL and SCDFL as the inputs of the DFNSA, four danger feature libraries are generated: NDFL1, NDFL21, SDFL1 and SDFL21, which make up the detecting feature library (DFL) of the proposed DFNSA-MD model. The features in the DFL are the basic elements to construct the danger feature vector of a sample.

3.2 Danger Feature Vector

In this paper, a sample is expressed as a danger feature vector to measure the danger of the sample. The danger feature vector is defined as

$$< \frac{M_{NDFL1}}{L_{NDFL1}}, \frac{M_{NDFL21}}{L_{NDFL21}}, \frac{M_{SDFL1} + M_{SDFL21}}{L_{SDFL1} + L_{SDFL21}} >$$

where M_i denotes the matching value of a sample and a library i, and L_i is the sum of weights of features in a library i, i =NDFL1, NDFL21, SDFL1, SDFL21.

The r-bit continuous matching is taken as the feature matching criteria. Here $r = R * 8$, i.e., the matching part of two features is also a danger feature. The matching value of a sample and a danger feature library is the sum of weights of the features in the library which match any feature of the sample.

The danger feature vector maps a sample into the whole danger feature space, and characterizes a sample efficiently and completely, making the DFNSA-MD model perform well. Every sample in a training set is expressed as a danger feature vector, which is taken as the input of a classifier.

4 Experiments

4.1 Datasets

The experiments in this paper are conducted on three public malware datasets: CILPKU08, Henchiri and VXHeanvens datasets. The three datasets and their composition documents can download from www.cil.pku.edu.cn/resources/.

The benign program dataset used here consists of files of Windows XP and a series of applications, which are the main paunching bag of malware.

Table 1. Experimental platform

CPU	Core 2 Duo 3.00 GHz
RAM	8 GB
Operating system	Win 7 64-bit

4.2 Experimental Setup

The support vector machine (SVM), realized in LibSVM [12], is taken as the classifier of the proposed DFNSA-MD model, and the area under the receiver operating characteristic curve (AUC) is utilized as the performance evaluation criteria. The information of the experimental platform is shown in Table 1.

In the experiments of Section 4.4, eight groups of experiments are taken on the three public malware datasets using 5-fold cross validation, and the 95% confidence intervals are computed to look into the stability of the proposed DFNSA-MD model. Both the CILPKU08 and Henchiri datasets mainly consist of computer viruses, so two experiments are carried on in the two datasets directly, ignoring the categories of malware. The VXHeavens dataset contains 7128 malware which fall into six categories, so we split this dataset into six smaller datasets: backdoor, constructor, miscellaneous, trojan, virus and worm. The miscellaneous includes DoS, Nuker, Hacktool and Flooder, while the malware in the other five smaller datasets, respectively, fall into a category. Six experiments are taken in the six smaller datasets.

In all the experiments, there is no overlap between a training set and a test set. That is to say, to a training set, the malware in a test set are unseen malware. This setting increases the reliability of the experiments.

The TNSA-based malware detection (TNSA-MD) model and the NSAPF-based malware detection (NSAPF-MD) model are imported for comparison.

4.3 Selection of Parameters

This section selects the instruction proportion: $P\%$ used in the MIL, using liner search, where $P = 0.5, 1.0, ..., 10.0$. We do not try larger P, since when $P = 10$, the MIL contains 6553 instructions and already covers a huge danger feature space. The experimental results are shown in Fig. 2.

Fig. 2 illustrates that, with the growth of P, the performance of the DFNSA-MD model shows steady downward trend as the MIL contains more and more instructions with unremarkable tendencies to malware. When $P = 1$, the DFNSA-MD model obtains the optimal AUC = 0.9039.

Generally speaking, the instruction proportion $P\%$ varies with different datasets. Hence we just set the optimization interval of P as $[0.5, 3]$ in the rest of experiments, instead of setting $P = 1$. In the rest of experiments, the P, which makes the DFNSA-MD model perform best in a training set, is set as the optimal P.

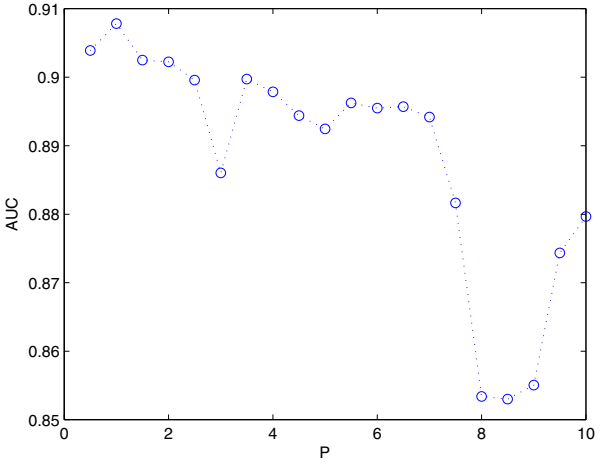


Fig. 2. The experimental results of the selection of parameters

Table 2. Experimental results

	TNSA-MD model	NSAPF-MD model	DFNSA-MD model
CILPKU08	0.9684 ± 0.00568	0.9688 ± 0.00907	0.9761 ± 0.00781
Hechiri	0.9634 ± 0.00755	0.9679 ± 0.01404	0.9808 ± 0.00428
Backdoor	0.8100 ± 0.02060	0.8190 ± 0.01764	0.8247 ± 0.01024
Constructor	0.9095 ± 0.03120	0.9202 ± 0.01545	0.9244 ± 0.01213
Miscellaneous	0.8243 ± 0.01603	0.8255 ± 0.01912	0.8394 ± 0.01028
Trojan	0.7901 ± 0.01332	0.8729 ± 0.01897	0.8735 ± 0.01714
Virus	0.6275 ± 0.01738	0.8746 ± 0.01187	0.8774 ± 0.01784
Worm	0.8252 ± 0.03697	0.8430 ± 0.04788	0.8489 ± 0.04101

4.4 Experimental Results

The experimental results of the proposed DFNSA-MD model are listed in Table 2. The experimental results of the TNSA-MD and NSAPF-MD models are also given in Table 2 for comparison.

From Table 2, the NSAPF-MD model is 4.67% better than the TNSA-MD model in all the experiments on average by making advantage of danger features extracted from malware. The detailed analysis will be given in Section 5.

The DFNSA-MD model outperforms the TNSA-MD and NSAPF-MD models for about 5.34% and 0.67% in all the experiments on average, respectively, without any losing in any experiment. The DFNSA-MD model makes use of all the danger features extracted from a training set, regardless of their categories. Hence the DFNSA-MD model is considered to be able to measure the danger of a sample more precisely, and achieves the best performance.

Table 3. The composition of the DFLs of the three models

	Detecting feature library
TNSA-MD model	NDFL1
NSAPF-MD model	NDFL1, NDFL21, NDFL22
DFNSA-MD model	NDFL1, NDFL21, SDFL1, SDFL21

The 95% confidence intervals of the three models are relatively small from Table 2, indicating that the results of these models are very stable and believable.

5 Discussions

5.1 Comparison of Detecting Feature Library

Table 3 lists the composition of the DFLs of the TNSA-MD, NSAPF-MD and DFNSA-MD models. It is easy to see that the DFL of the TNSA-MD model is the smallest DFL, consisting of NDFL1, i.e., the features merely appearing in non-self. Since the TNSA discards lots of danger features which are believed helpful, the performance of the TNSA-MD model is relatively bad.

The DFL of the NSAPF-MD model consists of NDFL1, NDFL21 and NDFL22, i.e., all the danger features appearing in non-self. The NSAPF reserves the non-self danger features which match self danger features by punishing these features, and obtains a larger DFL. Based on this DFL, the NSAPF-MD model detects malware by measuring the danger of a sample, and achieves good results.

The DFNSA-MD model owns the largest DFL which consists of all the danger features extracted from a training set. The DFNSA divides the danger feature space into four parts, and reserves the information of danger features to the utmost extent. It makes the danger feature vector of a sample contain as much information as possible and measure the danger of a sample better. In this way, the DFNSA-MD model outperforms the TNSA-MD and NSAPF-MD models in all the experiments.

5.2 Comparison of Detecting Time

The detecting time of a sample is proportionate to the number of the features in a DFL. We analyze the average detecting time of the three models for a sample in the virus dataset, in which the average size of a sample is 104 KB.

- The DFL of the TNSA-MD model is the smallest DFL, so it is faster than the other two models to detect a sample, just assuming 0.05 seconds on average.
- The size of the DFL of the NSAPF-MD model lays between that of the TNSA-MD and DFNSA-MD models, taking 0.12 seconds on average for detecting a sample.
- The DFNSA-MD model has the largest DFL, which consists of all the danger features extracted in a training set, so its detecting time is the longest, 0.15 seconds on average, basically meeting the demand of a real-time system.

6 Conclusions

In this paper, the DFNSA has been proposed and applied to detect malware. The DFNSA divides the danger feature space into four parts, and reserves the information of danger features to the utmost extent. Comprehensive experimental results suggest that the DFNSA is able to reserve as much information of danger features as possible, and the DFNSA-MD model is effective to detect unseen malware by measuring the danger of a sample precisely. It outperforms the TNSA-MD and NSAPF-MD models for about 5.36% and 0.67%, respectively.

In future work, we want to find a better way to measure the danger of a sample by importing the danger theory and text categorization methods.

Acknowlegements. This work is supported by the National Natural Science Foundation of China under grants No. 61170057 and 60875080.

References

1. Forrest, S., Perelson, A.S., Allen, L., Rajesh, C.: Self-nonself discrimination in a computer. In: IEEE Computer Society Symposium on Research in Security and Privacy, Oakland, pp. 202–212 (1994)
2. Forrest, S., Hofmeyr, S.A., Somayaji, A., Longstaff, T.A.: A sense of self for Unix processes. In: IEEE Symposium on Security and Privacy, Oakland, pp. 120–128 (1996)
3. Somayaji, A., Hofmeyer, S., Forrest, S.: Principle of a computer immune system. In: New Security Paradigms Workshop, Cumbria, pp. 75–82 (1998)
4. Matzinger, P.: The danger model: a renewed sense of self. Science's STKE 296(5566), 301–305 (2002)
5. Aickelin, U., Bentley, P., Cayzer, S., Kim, J., McLeod, J.: Danger Theory: The Link between AIS and IDS? In: Timmis, J., Bentley, P.J., Hart, E. (eds.) ICARIS 2003. LNCS, vol. 2787, pp. 147–155. Springer, Heidelberg (2003)
6. Ji, Z., Dasgupta, D.: Real-Valued Negative Selection Algorithm with Variable-Sized Detectors. In: Deb, K., et al. (eds.) GECCO 2004, Part I. LNCS, vol. 3102, pp. 287–298. Springer, Heidelberg (2004)
7. Li, Z., Liang, Y.W., Wu, Z.J., Tan, C.Y.: Immunity based virus detection with process call arguments and user feedback. In: Bio-Inspired Models of Network, Information and Computing Systems, Budapest, pp. 57–64 (2007)
8. Li, T.: Dynamic detection for computer virus based on immune system. Sci. China Inf. Sci. 39(4), 422–430 (2009) (in Chinese)
9. Wang, W., Zhang, P.T., Tan, Y., He, X.G.: A hierarchical artificial immune model for virus detection. In: International Conference on Computational Intelligence and Security, Beijing, pp. 1–5 (2009)
10. Wang, W., Zhang, P., Tan, Y.: An Immune Concentration Based Virus Detection Approach Using Particle Swarm Optimization. In: Tan, Y., Shi, Y., Tan, K.C. (eds.) ICSI 2010. LNCS, vol. 6145, pp. 347–354. Springer, Heidelberg (2010)
11. Zhang, P.T., Wang, W., Tan, Y.: A malware detection model based on a negative selection algorithm with penalty factor. Sci. China Inf. Sci. 53(12), 2461–2471 (2010)
12. LibSVM, http://www.csie.ntu.edu.tw/~cjlin/libsvm/

Alpha Matting Using Artificial Immune Network

Zhifeng Hao[1], Jianming Liu[1], Xueming Yan[2], Wen Wen[1], and Ruichu Cai[1,3]

[1] Faculty of Computer Science, Guangdong University of Technology, Guangzhou, 510006, China
[2] School of Computer Science, South China Normal University, GuangZhou, 510631, China
[3] State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, 210093, China
Jim8757@163.com

Abstract. Alpha matting refers to the problem of softly extracting the foreground from an image.To solve the matting problem initialized with a trimap (a partition of the image into three regions: foreground, background and unknown pixels), an approach based on artificial immune network is proposed in this paper.The method firstly uses Artificial Immune Network(aiNet) to map the color feature for unknown region, attaining the color subset both on the foreground and background color distributions,then estimate the alpha matte for unknown region, and finally apply guided filter to improve the matting results. Experiments on several different image data sets show that the proposed method produces high-quality matting results.

Keywords: alpha matting, Artificial Immune Network, feature map.

1 Introduction

Matting and compositing were originally developed for film and video production. In 1984, Porter and Duff [1] introduced the digital analog of the matte—the alpha channel—and showed how synthetic images with alpha could be useful in creating complex digital images. The most common compositing operation is the over operation, which is summarized by the compositing equation: the alpha matting model may be stated as

$$I = \alpha F + (1-\alpha)B \tag{1}$$

where α varies between [0,1] and represents the blending coefficient between the foreground object, F, and the background object, B. The α value has been variously interpreted as a probability that the pixel belongs to the foreground object, a partial differential equation solution, or as an interpolation between the known foreground and background objects. If we constrain the alpha values to be either 0 or 1 in Eq.1, the matting problem degrades to another classical problem: binary image/video

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 300–309, 2012.

segmentation, where each pixel fully belongs to either foreground or background. In addition, the matting problem is highly ill-posed, because the number of unknowns (F, B, and α) is much larger than the number of equations. Therefore, a user-specified trimap which indicates the known foreground /background and the unknown pixels is often required.

In this paper we present a new approach for extracting the alpha matte from a natural image. From information processing perspective, aiNet ,as a artificial immune system, which is a highly distributed, adaptive, and self-organizing information processing system, together with its learning, memory, feature extraction, and pattern recognition features, and offers rich metaphors for its artificial counterpart [17]. The proposed method firstly map the color feature on both the foreground and background color distributions with Artificial Immune Network(aiNet) attaining the color subset for each pixel in unknown region ,then estimate the alpha matte by the color subset,in final,apply guilded filter smoothing to further improve the matting results. A variety of experiments show that our method produces both visually and quantitatively high-quality matting results.

In the following, we introduce and analyze the related work in section 2. Section 3 describes our approach to alpha matting using feature map network in details, Section 4 gives an experimental comparison, Conclusion and future works are given in Section 5.

2 Related Work

Recently, various matting techniques and systems have been proposed to efficiently extract high quality mattes from both still images and video sequences [3]. Currently existing matting methods can be categorized as sampling-based or affinity-based.

Sampling-based methods firstly estimate the foreground and background color and then compute the alpha matte. Although the concept sounds simple, implementing such an algorithm that works well for general images is difficult. Some approaches [4], [6], [7], [8] ignore some of these difficulties by making ad hoc assumptions, and some [5]try to solve them in mathematical ways.These methods perform well in condition that the true foreground and background color are in the sample set. However, the true foreground/background colors are not always covered, because these methods only collect samples near each unknown pixel, and the number of samples is rather limited.

Unlike Sampling-based methods, affinities in affinity-based approaches are always defined in a small neighborhood, usually between immediately connected pixels or pixels in a 3×3 window. In such a small window, the pixel correlations are usually strong, thus the local smoothness assumption typically holds, even for moderately

complex images [9], [10], [11], [13].On the other hand, the defined affinities regularize the resulting matte to be locally smooth, thus fundamentally avoid matte discontinuities [12], [14]. However,there are two possible drawbacks of affinity-based approaches. Firstly, most approaches focus on first estimating alpha values, and only then estimate true foreground colors for unknown pixels based on pre-computed alphas, rather than estimating hem jointly for an optimal solution. Secondly, the alpha matte is estimated in a propagation fashion, from known pixels to unknown ones, thus small errors could be propagated and accumulated to produce bigger errors.

Both Sampling-based methods and affinity-based approaches have their own advantages and disadvantages. Sampling-based methods work better when dealing with distinct foreground and background color distributions along with carefully specified trimaps, but tend to generate large errors when their underlying assumptions are violated. On the other hand, affinity-based approaches are relatively insensitive to different user inputs and always generate smooth mattes. However, they tend to produce inaccurate results for images containing long and furry foreground structures. By combining these two methodologies through an optimization process, an advanced system can be developed, which achieves good trade-off between accuracy and robustness. But it is slow for mega-pixel images and not applicable for predrawn trimaps. More efficient method for high quality matting is still demanded.

3 The Proposed Algorithm for Matting

There are three major parts of our alpha matting algorithm: feature map network, alpha estimation and post-processing.we firstly map the pixels in unknown region into known foreground region and background region separately by feature map network, attaining the color subset both on the foreground and background color distributions;then estimate the alpha matte for unknown region through Theoretical Derivation ; and finally apply guided filter to improve the matting results.Details of feature map network is presented in the following .

3.1 Artificial Immune Network for Feature Map

The most remarkable roles of immune system are the protection of the organism against the attack of antigen. The primary problem the immune system faced with is the recognition of these antigen. After recognizing (identifying) an antigen, the immune response arises to avoid or block the antigen, and immune memory which is the feature map of the antigen is memorized in immune system[18].

In Immune feature map network, antigen refers to data in region Ω , and final antibody refers to data in resulting color subspace Ψ_i handling by aiNet[19]. The aiNet workflow is shown in Fig.1 :

Step 1. Initialization: create an initia.l random population of network antibodies in region Ω ;
Step 2. Antigenic presentation: for each antigenic pattern, run the following steps.

Antibodies activation: for each network element, determine its affinity with the antigen presented. Select a number of antibodies whose affinity with the antigen are more than a pre-specified threshold(activating threshold) as activated set; save the remaining antibodies in a unactivated set.

Clonal expansion: reproduce (clone) activated antibodies proportionally to their affinity; eliminate the antibodies whose clones is less than a pre-specified threshold(selecting threshold);

Affinity maturation: select a number of highest affinity antibodies in activated set and place them into a memory set. Mutate each clone of the remaining activated antibodies inversely proportional to affinity;

Clonal suppression: by incorporating memory antibodies with mutated antibodies, reconstruct the memory set, and then determine the concentration of each antibodies; eliminate antibodies whose concentration is more than a pre-specified threshold;

Network construction: incorporate the clonal memory set with unactivated antibody set, reconstruct the antibody population.

Network suppression: eliminate all network antibodies whose activated counting numbers are less than a pre-specified threshold(living threshold);

Step 3. Cycle: repeat steps 2 until a pre-specified number of iterations is reached.
Step 4. Output: the resulting data are output, and afforded to be sample data for further application.

Fig. 1. The workflow of Artificial Immune Network

As an example of each pixel in Ω mapping to Ω_F , several key steps of aiNet are described in details as follows:

***Affinity Definition*:** the affinity of antigen $Ag_i (i \in \Omega)$ and antibody $Ab_j (j \in \Omega_F)$ is computed according to Eq. (2).

$$aff_{ij} = 1/(1 + \mathrm{d}(Ag_i, Ab_j)) \tag{2}$$

where d(.) denotes the color space distance between Ag_i and Ab_j .

***Clonal Expansion*:** an initial weight value w_j is given to each activated antibody, and keeping the total weight value of activated set unchangeable. Then clone each antibody activated antibodies according to Eq.(3)and eliminate those antibodies whose weight value are lowest;

$$\mathrm{w}_j = \mathrm{int}(w_0 \times N_{act} \times \frac{aff_{ij}}{\sum_{j=1}^{N_{act}} aff_{ij}}) \tag{3}$$

Where w_0 is the initial weight value of antibody; N_{act} is the total number of activated antibodies.

***Affinity maturation*:** the main function of this step is to mutate the antibodies in order to improve the affinities of these antibody, and activated antibodies are mutated according to Eq.(4)

$$\mathrm{Ab}_k = Ab_k + \beta(Ag_i - Ab_k) \tag{4}$$

Where β is the mutation rate, and $\beta = \mathrm{e}^{1-\frac{\mathrm{N_{act}} * aff_{ij}}{N_{act}}}$ is inversely proportional to antibody's affinity.

Clonal suppression: the concentration of antibody is evaluated according to Eq(5)

$$C_j = \frac{\sum_{k=1}^{N_{act}} S_{jk}}{N_{act}} \tag{5}$$

Where $S_{\mathrm{jk}} = \begin{cases} 1 & Ab_{jk} / \max\{Ab_{jk}\} \geq \eta \\ 0 & Ab_{jk} / \max\{Ab_{jk}\} < \eta \end{cases}$; $\mathrm{Ab_{jk}}$ is the affinity of $\mathrm{Ab_j}$ and $\mathrm{Ab_k}$ and η is the concentration suppression threshold.

***Network suppression*:** at the beginning of algorithm, initialize each activated antibody with an activated counting number 0. During the iterations, the activated counting number is plus 1 whenever the antibody is activated. If the activated counting number is less than a pre-specified number, the antibody will be eliminated from the antibody population.

3.2 Alpha Matte Estimation

Matting algorithms typically assume that each pixel I_i in an input image is a linear combination of a foreground color F_i and a background color B_i in the compositing equation (1).In this work, we define Ω_F , Ω_B and Ω as "definitely foreground", "definitely background" and "unknown" regions respectively, we all kown that

$$\alpha_i = \begin{cases} 1, i \in \Omega_F \\ 0, i \in \Omega_B \end{cases}$$

Then we deformation the equation (1) by simple operation as follow:

$$\alpha_i = \frac{I_i - B_i}{F_i - B_i}, (i \in \Omega) \tag{6}$$

Here, I_i is a certain value, and we consider F_i and B_i as the variables in function (1) . In order to get an approximate gradient field of matte, we take the partial derivatives on equation (1):

$$\Delta\alpha_i = \frac{(B_i - I_i)}{(F_i - B_i)^2}\Delta F_i + \frac{(F_i - I_i)}{(F_i - B_i)^2}\Delta B_i \tag{7}$$

Where $\Delta = (\frac{\partial}{\partial F}, \frac{\partial}{\partial B})$ is the gradient operator. This is the differential form of the matting equation, for R, G, B channels individually. Owing to

$$\Delta\alpha_i \approx \alpha_i(F + \Delta F, B + \Delta B) - \alpha_i(F, B) \tag{8}$$

we can get the approximate matting equation:

$$\alpha_i = \alpha_i(F + \Delta F, B + \Delta B) - \frac{(B_i - I_i)}{(F_i - B_i)^2}\Delta F_i - \frac{(F_i - I_i)}{(F_i - B_i)^2}\Delta B_i \tag{9}$$

Then the above formula may be rewritten as:

$$\alpha_i = \frac{I_i - (B_i + \Delta B_i)}{F_i - B_i + \Delta F_i - \Delta B_i} - \frac{(B_i - I_i)}{(F_i - B_i)^2}\Delta F_i - \frac{(F_i - I_i)}{(F_i - B_i)^2}\Delta B_i \tag{10}$$

It means that the matte is estimated for each pixel, given the values of F_i, B_i, ΔF_i and ΔB_i .So we map each pixel I_i to Ω_F and Ω_B respectively with aiNet(Aritifical Immune Network) to attain the color subspace ψ_i^F and ψ_i^B respectively,which is the feature map of I_i individually.

Then we try to find the most likely eatimation for F_i and B_i by feature map network. Each pixel $I_i(i \in \Omega)$ is mapped to Ω_F ,We got the color subspace $\psi_i^F\{\varphi_k \mid k = 1, 2, ..., m\}$, ψ_i^F stands for the characteristic of I_i in color space Ω_F ,using it to express the value of F_i and ΔF_i as follows:

$$F_i = \frac{1}{m}\sum_{k=1}^{m} F_{\varphi_k} = \frac{1}{m}\sum_{k=1}^{m} I_{\varphi_k} (\varphi_k \in \psi_i^F \subseteq \Omega_F) \tag{11}$$

$$\Delta F_i = \frac{1}{m}\sum_{k=1}^{m} \left\| F_{\varphi_k} - F_i \right\| \tag{12}$$

We use analogues term to that of region Ω_B and got the color subspace $\psi_i^B\{\varphi_k \mid k = 1, 2, ..., n\}$,

$$B_i = \frac{1}{n}\sum_{k=1}^{n} I_{\varphi_k} (\varphi_k \in \psi_i^B \subseteq \Omega_B) \tag{13}$$

$$\Delta B_i = \frac{1}{n}\sum_{k=1}^{n} \left\| B_{\varphi_k} - B_i \right\| \tag{14}$$

Then in Eq.(10),we substitute the Eq.(11), Eq.(12), Eq.(13) and Eq.(14),then we can attain the rough alpha matte.

3.3 Post-Processing

To refine the result of alpha matting based on feature map network ,we apply a kind of post-processing(smoothing) by considering neighboring pixels can further improving the matting results. We adopt the fast guided filter proposed in [15], the filtering output on pixel *I* is expressed as a weighted average:

$$\alpha_{out} = \sum_j W_{ij}(I)\alpha_i \tag{15}$$

where i and j are pixel indexes. The filter kernel W_{ij} is a function of the guidance image *I* and independent of α .This filter is linear with to α . The kernel weights $W_{ij}(I) = \frac{1}{|w|^2}\sum_{k:(i,j)\in w_k}(1+\frac{(I_i-\mu_k)(I_j-\mu_k)}{\sigma_k^2+\varepsilon})$ can be explicitly expressed by:

$$W_{ij}(I) = \frac{1}{|w|^2}\sum_{k:(i,j)\in w_k}(1+\frac{(I_i-\mu_k)(I_j-\mu_k)}{\sigma_k^2+\varepsilon}) \tag{16}$$

Here, μ_k and σ_k^2 are the mean and variance of I in a window w_k , $|w|$ is the number of pixels in w_k , ε is a regularization parameter.

4 Experimental Results

To evaluate the performance of the proposed approach, we compare it with the Robust matting, Closed-form matting and Shared matting by MSE, using eight test Image sets of Levin et al. Here, MSE is the absolute color differences comparing approximate alpha matte with true alpha matte,The mean squares error (MSE) curves are illustrated in Fig.2.

As shown in Fig.2, our approach with feature map network model produces better (for some degree) results than other approaches: lower MSE value are obtained on almost 8 test image sets except the test image set 2 and the test image set 7; besides, our algorithms estimate better mattes with lower MSEs than Shared Matting, especially the image with hair curls and small holes as shown in Fig.4. The first eight rows in Table 1 shows the detailed MSE values for each test dataset, and the last row shows the average MSE. Results in table 1 demonstrate that our algorithm have advantages not only in accuracy, but also in robustness and efficiency.

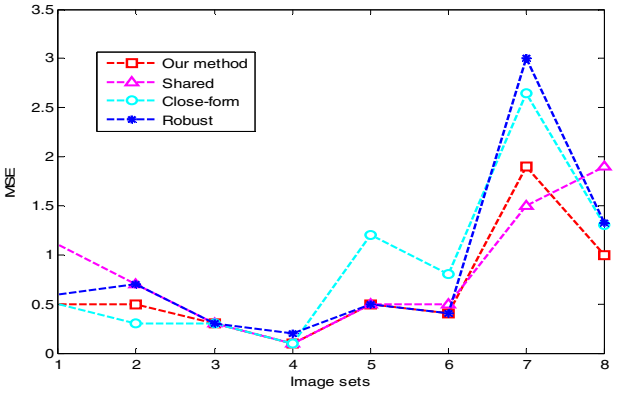
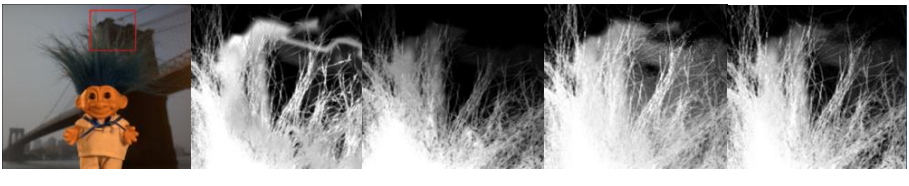


Fig. 2. MSE curves for 8 test image sets

Table 1. The MSE values for all algorithms on all the test images along with the average MSE

MSE	Robust	Close-form	Shared	Our mehold
T1	0.6	0.5	1.1	0.5
T2	0.7	0.3	0.7	0.5
T3	0.3	0.3	0.3	0.3
T4	0.2	0.1	0.1	0.1
T5	0.5	1.2	0.5	0.5
T6	0.4	0.8	0.5	0.4
T7	3.0	2.6	1.5	1.9
T8	1.3	1.3	1.9	1.0
Ave	0.878	0.893	0.825	0.656

Fig.3 and Fig.4 show the results of mattes with different algorithms on one test image.Specially, partial mattes are extracted in region with red border by Robust matting, Closed-form matting, Shard matting and our algorithm, respectively in Fig.3. Our approach produces high accurate results because the more color characters can handled by feature map network.



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

Fig. 3. (a). Original image. Following images show partial matte extracted in region with red border by (b). Robust matting. (c). Closed-form matting. (d).Shard matting(e). Our algorithm.

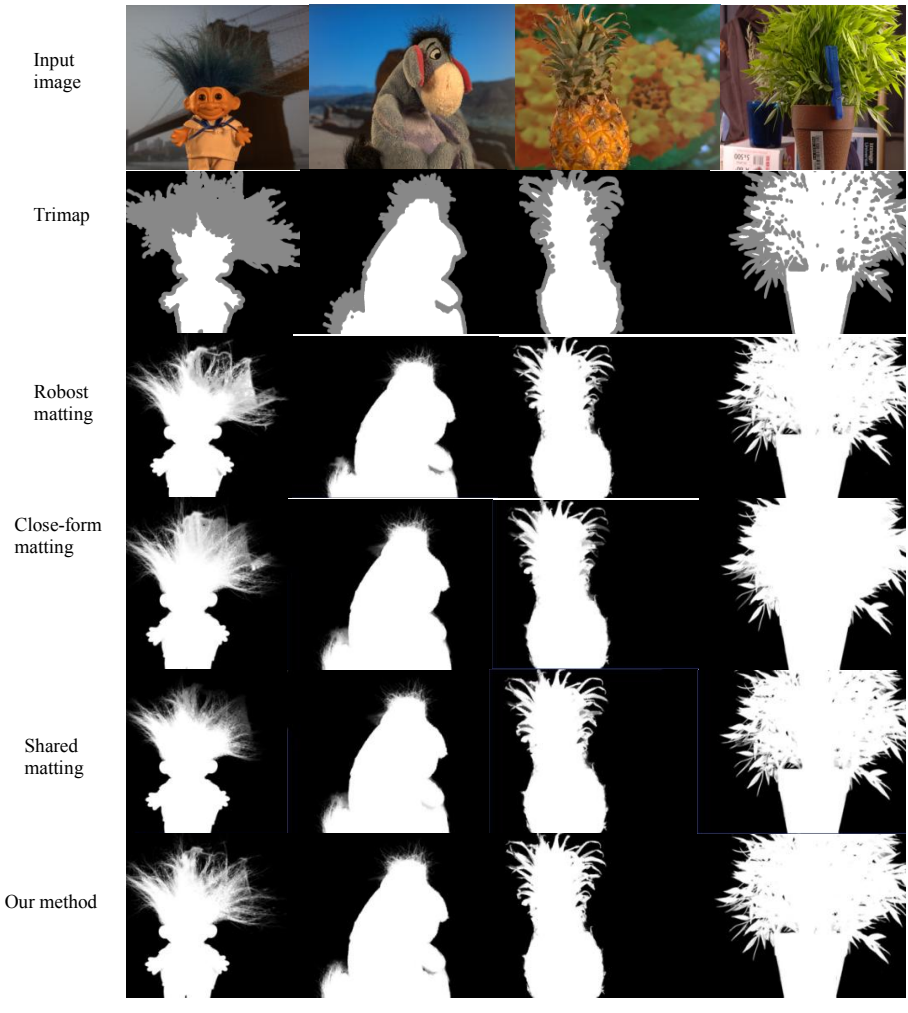


Fig. 4. Nature image Matting.these four image mattes are extracted by Robust matting, Closed-form matting ,Shard matting and Our algorithm respectively

5 Conclusions and Future Work

By applying artificial immune network to feature map color model, we propose a new matting approach. It can easily learn more general color model in both linear and nonlinear cases. The proposed method inspire us that works in the area of neural network learning,such as RBF network or other supervised learning methods, might also be implemented in the matting problem.

Acknowledgements. This work has been supported by Natural Science Foundation of China (61070033, 61100148), Natural Science Foundation of Guangdong province

(9251009001000005, S2011040004804), Key Technology Research and Development Programs of Guangdong Province (2010B050400011), Opening Project of the State Key Laboratory for Novel Software Technology (KFKT2011B19), Foundation for Distinguished Young Talents in Higher Education of Guangdong (LYM11060, LYM09068).

References

1. Porter, T., Duff, T.: Compositing digital images. In: 11th Annual Conference on Computer Graphics and Interactive Techniques, pp. 253–259. ACM Press, New York (1984)
2. http://www.alphamatting.com
3. Jue, W., Cohen, M.F.: Image and Video Matting: A Survey. Computer Graphics and Vision 3 (2007)
4. Ruzon, M.A., Tomasi, C.: Alpha estimation in natural images. In: Computer Vision and Pattern Recognition, pp. 18–25. IEEE Press, New York (2000)
5. Chuang, Y., Curless, B., Salesin, D., Szeliski, R.: A bayesian approach to digital matting. In: Computer Vision and Pattern Recognition, pp. 264–271. IEEE Press, New York (2001)
6. Rhemann, C., Rother, C., Gelautz, M.: Improving color modeling for alpha matting. In: BMVC (2008)
7. Wang, J., Cohen, M.: Optimized color sampling for robust matting. In: Computer Vision and Pattern Recognition, pp. 1–8. IEEE Press, New York (2007)
8. Gastal, E.S.L., Oliveira, M.M.: Shared sampling for real-time alpha matting. Computer Graphics Forum, 575–584 (2010)
9. Bai, X., Sapiro, G.: A geodesic framework for fast interactive image and video segmentation and matting. In: 10th IEEE International Conference on Computer Vision, pp. 1–8. IEEE Press, New York (2007)
10. Rother, C., Kolmogorov, V., Blake, A.: Grabcut - interactive foreground extraction using iterated graph cut. In: Proceedings of ACM SIGGRAPH, pp. 309–314. ACM Press, New York (2004)
11. Sun, J., Jia, J., Tang, C.K., Shum, H.Y.: Poisson matting. In: Proceedings of ACM SIGGRAPH, pp. 315–321. ACM Press, New York (2004)
12. Zheng, Y., Kambhamettu, C., Yu, J., Bauer, T., Steiner, K.: Fuzzymatte:A computationally efficient scheme for interactive matting. In: Computer Vision and Pattern Recognition, pp. 1–8. IEEE Press, New York (2008)
13. Levin, A., Lischinski, D., Weiss, Y.: A closed form solution to natural image matting. In: Pattern Analysis and Machine Intelligence, pp. 228–242. IEEE Press, New York (2008)
14. Levin, A., Ravacha, A., Lischinski, D.: Spectral matting. In: Computer Vision and Pattern Recognition, pp. 1–8. IEEE Press, New York (2007)
15. Wang, J., Cohen, M.: Optimized color sampling for robust matting. In: Computer Vision and Pattern Recognition, pp. 1–8. IEEE Press, New York (2007)
16. He, K., Sun, J., Tang, X.: Guided Image Filtering. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6311, pp. 1–14. Springer, Heidelberg (2010)
17. Hart, E., Timmis, J.: Application areas of AIS: the past, the present and the future. Applied Soft Computing, 191–201 (2008)
18. Castro, L.N., Timmis, J.I.: Artificial immune systems as a novel soft computing paradigm. Soft Computing, 526–544 (2003)
19. Ge, H., Yan, X.: A Modified Artificial Immune Network for Feature Extracting. In: Tan, Y., Shi, Y., Chai, Y., Wang, G. (eds.) ICSI 2011, Part I. LNCS, vol. 6728, pp. 408–415. Springer, Heidelberg (2011)

Forecasting Mineral Commodity Prices with Multidimensional Grey Metabolism Markov Chain

Yong Li[1,*], Nailian Hu[1], and Daogui Chen[2]

[1] School of Civil and Environmental Engineering,
University of Science and Technology Beijing, Beijing 100083, China
yongli9898@hotmail.com, hnl@ustb.edu.cn
[2] Minmetals Exploration & Development Co. Ltd, Beijing, 100044, China
chendg@minmetals.com

Abstract. The price with large random fluctuation in mineral market has made it very difficult to do an accurate forecast. To overcome this problem, a multidimensional grey metabolism Markov forecasting method is proposed based on the theories of Grey forecast and Stochastic process. The forecasting effect of the model is tested through a case study and analysis with MATLAB software. The research results indicate that the forecasting precision of the proposed method is high and not limited to forecasting step length. So the method can be used to do a long term forecasting for mineral commodity prices without considering economic crisis.

Keywords: Mineral commodity, Price forecasting, Multidimensional grey, Metabolism, Markov chain.

1 Introduction

The prices of mineral commodity have an important significance for guiding mineral produce and adjusting mineral industrial structure. The international mineral market is affected by many factors. It is very difficult to know clearly about the action mechanism of the factors for mineral prices. All of these lead that the investment for mineral market has a great uncertainty. The uncertainty mainly expresses in the difficulty for price prediction. Moreover, because of the particularity of mineral market, mineral commodity prices fluctuate widely and mining investment can not be controlled effectively. So it is critical for us to forecast the prices by using proper theories and methods.

The prediction of mineral commodity prices is an systemic problem with big uncertainty. To overcome this complex problem, many researchers have made a great effort to improve forecasting techniques by using plenty of methods and models, such as Grey-Markov models [1], time series ARIMA models [2][3] and [4], BP neural network [5] and [6], nonlinear regression models [7], AFINS [8], and other methods [9][10] and [11]. However, the forecasting performance of the techniques are not so satisfactory that it should be improved. The reason is that grey forecasting models has a good extrapolation capability, but their fitting effect is not so well. Time series

* Corresponding author. Tel.:+86 13810948239.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 310–317, 2012.

ARIMA models, BP neural network, nonlinear regression models and AFINS fit the actual values well but their extrapolation capability is limited to forecasting step length greatly. So in this paper, we attempt to use a multidimensional metabolism method to do mineral commodity price forecasting research based on GM(1,1) model and Markov chain.

2 Theories and Methodology

2.1 Grey Forecasting Theory and GM(1,1) Model

Deng(1982) proposed grey theory which is a multidisciplinary and generic theory for dealing with the systemic problems with poor, incomplete or uncertain messages [12]. Based on the theory, The accumulated generation operation (AGO) is defined as Eq.(1) by assuming the original data sequence by $X^{(0)}=(x^{(0)}(1), x^{(0)}(2),\ldots, x^{(0)}(n))$.

$$X^{(1)}=x^{(1)}(k)=\{x^{(1)}(1),x^{(1)}(2)\cdots,x^{(1)}(n)\}=\sum_{i=1}^{k}x^{(0)}(i) \qquad (k=1,2,...,n) \tag{1}$$

Then the first-order difference equation can be expressed as Eq.(2):

$$\frac{dx^{(1)}}{dt}+ax^{(1)}=b \tag{2}$$

Where a is a development coefficient, b is a grey action coefficient. Therefore, the solution of Eq. (2) can be obtained as Eq.(3) by using the least square method.

$$x^{(1)}(k+1)=[x^{(1)}(0)-\frac{b}{a}]e^{-ak}+\frac{b}{a} \qquad (k=1,2,\cdots,n) \tag{3}$$

Where, $\begin{bmatrix}a\\b\end{bmatrix}=(B^TB)^{-1}B^TY_n$, and $B=\begin{bmatrix}-\frac{1}{2}[x^{(1)}(1)+x^{(1)}(2)] & 1\\ -\frac{1}{2}[x^{(1)}(2)+x^{(1)}(3)] & 1\\ \cdots & \cdots\\ -\frac{1}{2}[x^{(1)}(n-1)+x^{(1)}(n)] & 1\end{bmatrix}$, $Y_n=\begin{bmatrix}x^{(0)}(2)\\x^{(0)}(3)\\\cdots\\x^{(0)}(n)\end{bmatrix}$

Then grey forecasting function can be defined as Eq.(4):

$$x^{(1)}(k+1)=[x^{(0)}(1)-\frac{b}{a}]e^{-ak}+\frac{b}{a} \qquad (k=1,2,\cdots,n) \tag{4}$$

Where $x^{(1)}(0)=x^{(0)}(1)$. Finally we can get the forecasting sequence $X^{(f)}$ by restoring computation with Eq.(5):

$$x^{(f)}(k)=x^{(1)}(k+1)-x^{(1)}(k) \qquad (k=1,2,\cdots,n) \tag{5}$$

2.2 Markov Chain and Probability Matrix

Assuming an original data series $X^{(0)}=\{x^{(0)}(1), x^{(0)}(2),\ldots, x^{(0)}(n)\}$ is a discrete stochastic process, if it satisfies the following conditions:

(1) For each t (t = 1,2,…,*n*), the state space of $X^{(0)}$ is a set of integers (denoted by *I*); (2) For any r +1 non-negative integers t_1, t_2, ..., m ($0 < t_1 < t_2 < ... < t_r < m < n$) and any positive integer k, and the state i_1, i_2, i, ..., i_r, i, $j \in I$. Then

$$P\{x^{(0)}(t_1)=i_1, x^{(0)}(t_2)=i_2, ..., x^{(0)}(t_r)=i_r, x^{(0)}(m)=i\}>0 \ \ and$$
$$P\{x^{(0)}(m+k)=j | x^{(0)}(t_1)=i_1, x^{(0)}(t_2)=i_2, ..., x^{(0)}(t_r)=i_r, x^{(0)}(m)=i\}$$
$$=P\{x^{(0)}(m+k)=j \mid x^{(0)}(m)=i\}$$

Thus the series $X^{(0)}$ is a Markov Chain[13][14] and [15]. $P\{x^{(0)}(m+k)=j| \ x^{(0)}(m)=i\}$ is the transition probability when the state shifts from i at time m to state j at time $m+k$. The transition probability can be denoted by P_{ij} (m, $m+k$) and it has the following properties.

$$\begin{cases} P_{ij}(m,m+k)=p\{x^{(0)}(m+k)=j \mid x^{(0)}(m)=i\} \\ P_{ij}(m,m+k)=\dfrac{N_{ij}(m,m+k)}{N_i(m)} \\ P_{ij}(m,m+k)\geq 0 \\ \sum\limits_{j\in I} P_{ij}(m,m+k)=1 \\ i,j\in I, \quad k=1,2,\cdots. \end{cases} \tag{6}$$

Where, I is the state space of $X^{(0)}$, $N_{ij}(m,m+k)$ is the times when the state shifts from state i at time m to state j at time $m+j$, $N_i(m)$ is the number of state i at time m. The matrix consisting with $P_{ij}(m, m+k)$ as elements is the k step transition probability matrix of $X^{(0)}$. We denote it by $P(k)$.

$$P(\mathrm{k})=\begin{bmatrix} P_{11}^{(k)} & P_{12}^{(k)} & \cdots & P_{1j}^{(k)} \\ P_{21}^{(k)} & P_{22}^{(k)} & \cdots & P_{2j}^{(k)} \\ \vdots & \vdots & \vdots & \vdots \\ P_{i1}^{(k)} & P_{i2}^{(k)} & \cdots & P_{ij}^{(k)} \end{bmatrix} \qquad (i=1,2,\cdots;j=1,2,\cdots;j=1,2,\cdots.) \tag{7}$$

2.3 Prediction Algorithm Description

With the original sequence $X^{(0)}=X^{(0)}(1)=\{x^{(0)}(1),x^{(0)}(2),\ldots,x^{(0)}(\mathrm{n})\}$, multidimensional grey fitting calculation is carried out based on grey forecasting model. At first, the first set of fitting values is computed based on $X^{(0)}$. Then taking out the first data of $X^{(0)}(1)$, we get a new data sequence $X^{(0)}(2)=(x^{(0)}(2),\ldots, x^{(0)}(\mathrm{n}))$, and the second set of fitting values is obtained based on $X^{(0)}(2)$. With repeating this work, many sets of sequence $X^{(0)}(\lambda)=(x^{(0)}(\lambda), x^{(0)}(\lambda+1)\ldots, x^{(0)}(\mathrm{n}))$ ($\lambda=1,\cdots,n$) are finished till the length of the new sequence is no less than $4(n-\lambda\geq 4)$[12]. Based on $X^{(0)}(\lambda)$, a cluster of fitting sequence can be calculated and denoted by $X^{(f)}(\lambda)=\{x^{(f)}(\lambda), x^{(f)}(\lambda+1),\ldots, x^{(f)}(n)\}$. Here n-λ is the dimension of the sequence $X^{(f)}(\lambda)$. Then the mean absolute errors(MAE) of

the fitting values and original values are calculated by using Eq.(8). With comparing the values of *MAE*, the best value of n-λ can be determined and noted by τ. Hence $X^{(f)}(n\text{-}\tau)$ is the best fitting sequence of the sequence cluster $X^{(f)}(\lambda)$, and its curve fits the actual curve best.

$$MAE = \frac{1}{n\text{-}\lambda}\sum_{\lambda}^{n} | x^{(f)}(\lambda) - x^{(0)}(\lambda) | \quad (\lambda = 1, 2, \cdots, n) \tag{8}$$

With treating the best fitting curve as a baseline and a given value d, a set of state intervals can be divided as following:

$$\begin{cases} \phi = \{\phi_1, \phi_2, \cdots, \phi_\alpha, \phi_{\alpha+1}, \cdots, \phi_{\alpha+\beta-1}, \phi_{\alpha+\beta}\} \\ \phi_1 = [X^{(f)}(n-\tau) - \alpha d, X^{(f)}(n-\tau) - (\alpha\text{-}1)d] \\ \phi_2 = [X^{(f)}(n-\tau) - (\alpha-1)d, X^{(f)}(n-\tau) - (\alpha\text{-}2)d] \\ \quad \vdots \\ \phi_\alpha = [X^{(f)}(n-\tau) - d, X^{(f)}(n-\tau)] \\ \phi_{\alpha+1} = [X^{(f)}(n-\tau), X^{(f)}(n-\tau) + d] \\ \quad \vdots \\ \phi_{\alpha+\beta} = [X^{(f)}(n-\tau) + (\beta-1)d, X^{(f)}(n-\tau) + \beta d] \end{cases} \quad (\alpha = 1, 2, \cdots, \text{n} \ ; \beta = 1, 2, \cdots, n.) \tag{9}$$

Where, Φ is a state interval set, α is the number of intervals below the baseline, β is the number of intervals above the baseline. The values of d, α and β are decided by actual data sequence. If the state space I=Φ=$\{\Phi_1, \Phi_2, \ldots, \Phi_{\alpha+\beta}\}$, $P(k)$ will be a $(\alpha+\beta)$-order matrix. Assuming the initial state vector of a stochastic time series is S^0=(S_1, S_2,…, $S_{\alpha+\beta}$), the state vector after one step transition can be denoted by S^* and computed with Eq.(10).

$$S^* = S^0 \times P(1) = (S_1, S_2, \cdots, S_{\alpha+\beta}) \times \begin{bmatrix} P_{\phi_1\phi_1} & P_{\phi_1\phi_2} & \cdots & P_{\phi_1\phi_{\alpha+\beta}} \\ P_{\phi_2\phi_1} & P_{\phi_2\phi_2} & \cdots & P_{\phi_2\phi_{\alpha+\beta}} \\ \vdots & \vdots & \vdots & \vdots \\ P_{\phi_{\alpha+\beta}\phi_1} & P_{\phi_{\alpha+\beta}\phi_2} & \cdots & P_{\phi_{\alpha+\beta}\phi_{\alpha+\beta}} \end{bmatrix} \tag{10}$$

$$= (\sum_{i=1}^{\alpha+\beta} S_i P_{\phi_i\phi_1}, \sum_{i=1}^{\alpha+\beta} S_i P_{\phi_i\phi_2}, \cdots, \sum_{i=1}^{\alpha+\beta} S_i P_{\phi_i\phi_{\alpha+\beta}}) \qquad (\alpha = 1, 2, \cdots, n \ ; \beta = 1, 2, \cdots, n.)$$

Then the prediction interval can be computed based on the values of S^0 and S^*. With assuming the prediction interval is $[\Phi_{h-1}, \Phi_h]$(h=1,2,…, $\alpha+\beta$), final prediction value Y_t is the mean value of Φ_{h-1} and Φ_h, namely Y_t=(Φ_{h-1}+ Φ_h)/2. we take out the old data of $x^{(0)}$(n-τ) and set $x^{(0)}(n+1)$= Y_t for constructing a new data sequence{$x^{(0)}$(n-τ+1), $x^{(0)}$(n-τ+2),…, $x^{(0)}(n+1)$}. With repeating this work for continuing to do next step forecasting, we can achieve our forecasting objective.

3 A Case Study

In order to demonstrate the effectiveness of the proposed method, the historical price records of international copper market from 1990 to 2011 are used as our research

data. we use the data from 1990 to 2007 to forecast the four price values of 2008-2011 and test the forecasting effect with actual price values of 2008-2011. First we use the data from 1990 to 2007 to calculate the best fitting sequence. With MATLAB software as a tool, the results of multidimensional grey fitting calculation are shown in Table 1 and Fig. 1. It is clear that the best value of dimension is τ=15. This indicates that the fitting sequence with 15 dimension fits the actual prices best and its curve can reflect copper price changing trend.

Table 1. MAE for multidimensional grey fitting calculation

Items	λ	Data sequences	Dimensions	MAE
1	2001	2001～2007	7	4700.4
2	2000	2000～2007	8	3772.7
3	1999	1999～2007	9	2689.8
4	1998	1998～2007	10	2017.9
5	1997	1997～2007	11	1521.8
6	1996	1996～2007	12	1216.4
7	1995	1995～2007	13	1042.4
8	1994	1994～2007	14	1030.5
9	1993	1993～2007	15	1027.0
10	1992	1992～2007	16	1052.9
11	1991	1991～2007	17	1101.6
12	1990	1990～2007	18	1145.8

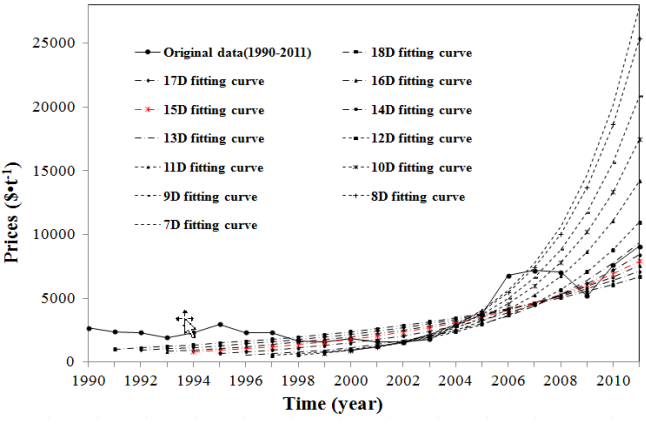


Fig. 1. The curve cluster of multidimensional grey fitting calculation

With the 15 dimension fitting curve of sequence(1993-2007) as a baseline, the initial state intervals and vectors of the sequence (1993-2007) are determined by using Eq.(9), state transaction matrixes are computed by using Eq.(6) and Eq.(7), and the new vectors after one step transaction are computed according to Eq.(10). Based on the new vectors, we find the forecasting interval. The middle point value of the

forecasting interval is the final forecasting value of 2008. With taking out the first data of sequence(1993-2007) and adding the forecasting value of 2008 to the end of sequence(1993-2007), a new sequence(1994-2008) is constructed. Then we can obtain the forecasting value of 2009 by repeating the above forecasting calculation steps. With repeating the above work again and again. The other two sequences (1995-2009, 1996-2010) and two forecasting values of 2010 and 2011 can be obtained. The forecasting process is shown in Fig. 3, Fig. 4, Fig. 5, Fig. 6 and Table 2 is the forecasting results.

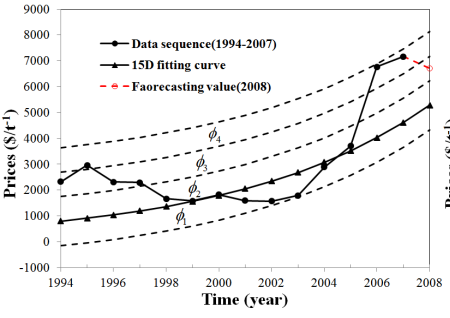


Fig. 2. Copper price forecasting(2008)

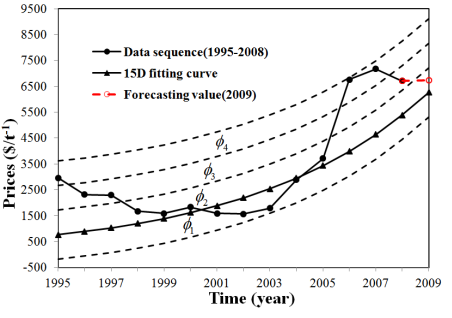


Fig. 3. Copper price forecasting(2009)

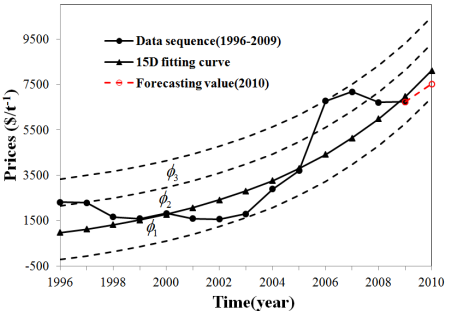


Fig. 4. Copper price forecasting(2010)

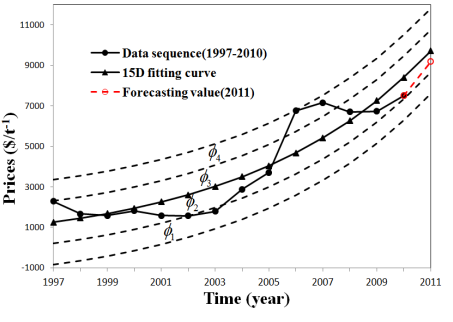


Fig. 5. Copper price forecasting(2011)

Table 2. Multidimensional grey metabolism Markov forecasting results(2008-2011)

Year	Number of intervals	Forecasting intervals	Forecasting values
2008	4(d=950)	[6230.9,7180.9]	6706.4
2009	4(d=950)	[6260.6,7210.6]	6733.1
2010	3(d=1180)	[6923.2,8103.2]	7513.2
2011	4(d=1050)	[8670.8,9720.8]	9195.8

4 Analysis and Evaluation

The four forecasting values and intervals are shown in Table 2. It is obvious that the actual values of 2008, 2009, 2010 are all in the corresponding forecasting intervals

except the value of 2009. However, considering the world economic crisis in 2009, the forecasting results can also be accepted.

In order to evaluate the forecasting effect of the proposed method further, we compared the method with ARIMA model. The actual price values, the forecasting price values of the two methods and their corresponding relative errors are shown in Table 3. A plot of the price values appears in Fig. 6. It is clear that the forecasting curve of our proposed method fits with the original curve much better than ARIMA model but the value of 2009. Moreover, the relative errors of our proposed method are 4.59%, 30.22%, 0.95% and 1.36% while the errors of ARIMA model are 6.18%, 22.97%, 13.46% and 20.29%. As is shown in Fig. 7, the relative errors of the proposed method are much lower than ARIMA model except the error value of 2009. Hence we get the conclusion that the proposed method performs excellently and not limit to forecasting step length in mineral copper price prediction.

Table 3. Forecasting price values and relative errors (2008-2011)

Year	Actual values	Multidimensional grey metabolism Markov	RE(%)	ARIMA model	RE(%)
2008	7029.4	6706.4	4.59	7464.0	6.18
2009	5170.5	6733.1	30.22	6358.3	22.97
2010	7585.6	7513.2	0.95	6564.8	13.46
2011	9072.3	9195.8	1.36	7231.5	20.29

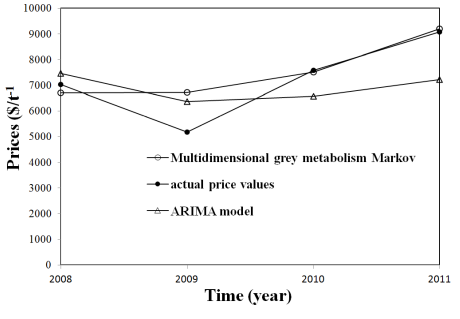


Fig. 6. Graph of actual and forecasting values

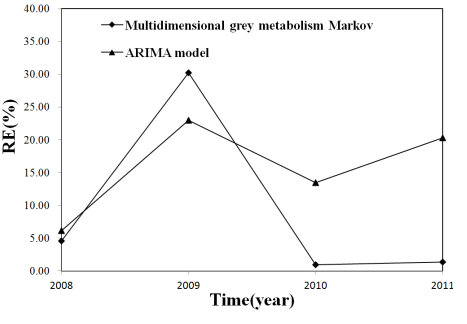


Fig. 7. Graph of forecasting relative errors

5 Conclusions

The multidimensional grey metabolism Markov forecasting method has been proposed based on grey forecasting model and stochastic process theory in this paper. With the international copper prices as a data sequence, the method is successfully applied to predict the four copper prices of 2008-2011. According to the forecasting values and relatives errors, our proposed method performs much better than ARIMA model in mineral copper price forecasting. Hence we get the conclusion that our proposed method not only performs an excellent forecasting effect but also is not limited to the forecasting steps in mineral market. Multidimensional grey metabolism

Markov forecasting method is proposed as a precisely and satisfactorily method for long term forecasting for mineral commodity prices. However, the influence of economic crisis for price forecasting can not be eliminated in our research.

References

1. Chengbao, W., Yanhui, C., Lihong, L.: The Forecast of Gold Price Based on the GM (1, 1) and Markov Chain. In: IEEE International Conference on Grey Systems and Intelligent Services, pp. 739–743. IEEE Press, Nanjing (2007)
2. Dooler, G., Lenihan, H.: An assessment of time series methods in metal price forecasting. Reso. Poli. 30, 208–217 (2005)
3. Jianhong, C., Xueyan, Y., Shan, Y., Lang, L.: Analysis and Forecast About Mineral Product Price Based on Time Series Model. Kunming. Univ. Sci. Technol. 34(6), 9–14 (2009)
4. Biao, Y., Zhouquan, L., Guang, L., Yiwei, Y.: Open pit mining limit dynamic optimization based on economic time series forecasting. China Coal Soc. 36(1), 29–33 (2011)
5. Chunmei, L.: Price Forecast For Gold Futures Based on GA-BP Neural Network. In: IEEE International Conference on Management and Service Science. IEEE Press, Wuhan (2009)
6. Lian, Z., Dandi, M., Zongxin, L.: Gold Price Forecasting Based on Improved BP Neural Network. Compu. Simul. 27(9), 200–203 (2010)
7. Xiao, F., Jin, W.: Mineral Price Forecasting Based on Nonlinear Regression. Nonf. Metals Sci. Eng. 2(3), 72–75 (2011)
8. Shijiao, Y., Changzhen, W., Jianyong, D., Xiaoyu, H.: Comparative Analysis of Price Prediction of Molybdenum Products Based on ANFIS and BPNN. In: IEEE International Conference on Computational Intelligence and Industrial Application, pp. 126–129. IEEE Press, Wuhan (2010)
9. John, E.T.: Long-term trends in copper prices. Mining Eng. 54(7), 25–32 (2002)
10. Shahriar, S., Erkan, T.: An overview of global gold market and gold price forecasting. Res. Pol. 35, 178–189 (2010)
11. Roberts, M.C.: Duration and characteristics of metal price cycles. Res. Poli. 34, 87–102 (2009)
12. Julong, D.: Grey Forecasting and Decision. Huazhong University of Science & Technology Press, Wuhan (1986)
13. Zhaoben, F., Boqi, M.: Stochastic Process. Science Press, Beijing (2004)
14. Davis, M.H.A.: Markov Models and Optimization. St. Edmundsbury Press, Landon (1993)
15. Freedman, D.: Markov Chains. Holden-Day, San Francisco (1983)

A Web-Service for Automated Software Refactoring Using Artificial Bee Colony Optimization

Ekin Koc[1], Nur Ersoy[2], Zelal Seda Camlidere[3], and Hurevren Kilic[4]

[1] Golbasi Bahcelievler Mah. 91 Sok. Ankara, Turkey
[2] Innova, Middle East Technical University, Teknokent, Ankara, Turkey
[3] Locksmith Software Technologies: LST Yazilim, Ankara, Turkey
[4] Computer Engineering Department., Gediz University, Menemen, Izmir, Turkey
{antimon,nersoy88,zelalseda}@gmail.com,
hurevren.kilic@gediz.edu.tr

Abstract. Automated software refactoring is one of the hard combinatorial optimization problems of search-based software engineering domain. The idea is to enhance the quality of the existing software under the guidance of software quality metrics through applicable refactoring actions. In this study, we designed and implemented a web-service that uses discrete version of Artificial Bee Colony (ABC) optimization approach in order to enhance bytecode compiled Java programming language codes, automatically. The introduced service supports 20 different refactoring actions that realize intelligent ABC searches on design landscape defined by an adhoc quality model being an aggregation of 24 object-oriented software metrics.

Keywords: Discrete Artificial Bee Colony Optimization, Search-Based Software Engineering, Software Quality, Web-Services.

1 Introduction

Artificial bee colony (ABC) search is a popular, population based metaheuristic search algorithm inspired from the collective intelligent behavior of honey bees [1]. The algorithm has been designed to work for unconstrained and constrained numerical optimization problems [2]. The basic idea is to refer each solution within the solution space as a food source. While using a population of food sources and various bee types (including employed, onlooker and scout) to search for better food sources (i.e. solutions), ABC mimics the behavior of honey bees on a search space.

Automated software refactoring is known to be one of the hardest problems of optimization oriented software engineering [3][4]. The difficulty is mainly due to candidate solution representation, objective function description and necessity of functional behavior preservation of software. In [5], the problem is formulated as a combinatorial optimization problem whose objective function is characterized by an aggregate of object-oriented metrics or pareto-front solution description. Efficient problem formulation; realistic individual representation and applicable change operator description are three basic factors that affect the performance of possible optimization

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 318–325, 2012.

effort. In automated software refactoring literature, metaheuristic-based direct/indirect optimization techniques including simple hill climbing (HC), simulated annealing (SA), multiple first ascent hill climbing (MFAHC) and genetic algorithms (GA) have been studied [6]. A multi-level design exploration approach to the problem has been introduced in [7]. In our recent empirical study [8], performance results of alternative search algorithms including pure random (RND), steepest descent (SD), multiple first descent (MFD), simulated annealing (SA), multiple steepest descent (MSD) and artificial bee colony (ABC) searches have been reported by the authors of this paper. In this study, we have concluded that MSD and ABC algorithms are the most suitable approaches for efficient solution of the problem. In our next study, we showed that different from MSD search, population-based, scalable and being suitable for parallel execution characteristics of ABC search [9] makes it a good choice for designing and implementing a high performance, highly scalable web-service. Detailed comparative performance results between ABC and other above mentioned approaches (including classical MSD) can be found in [8] and [9].

In the light of the above observations, the authors designed and implemented a web-service for automated software refactoring problem using ABC optimization as suitable optimization choice. To the best of our knowledge, the proposed solution introduces the first web-service on the automated refactoring problem domain. Note that, a fixed demand to the number of nodes to be expanded at each generation makes ABC algorithm a good choice for the web-service implementation. Furthermore, the quality gain is increased steadily against increasing population sizes while the required computation time per run increases only in the order of food source size and number of generations. The intended users of the system are software developers who aim to improve their code-quality through such automated service that can be made an extended part of their development environment. In Section 2, we give information about developed web-service including the assumptions and limitations. Domain specific considerations and application details of the applied ABC optimization technique can be found in Section 3. Finally, the conclusions are in Section 4.

2 The Refactoring Web-Service

There are two basic ingredients for the formulation of search-based optimization problems [10]: The way we represent candidate solutions and the way we measure fitness (or quality) of the solution. These ingredients also shape the proposed web-service solution in which better candidate design solutions are found under the guidance of predefined aggregate quality/fitness metric, in parallel manner. Selection of suitable refactoring actions and applying them at the design level enables the design space movement while taking care about functional behavior preservation of the input Java bytecode under analysis. Simply, we create web-service that delivers software refactoring features over web to compatible clients using standard web-services.

2.1 Basic Features

A Web-service is a specific kind of service that is identified by a URI which uses the XML-based standards, WSDL (Web-services Description Language) and UDDI

(Universal Description, Discovery and Integration) open Internet standards in order to form a software system designed to support interoperable client and service provider interaction over a network [11]. XML offers a widely adopted standard way of representing text and data in a format that can be exchanged across different operating systems, languages, and applications [12].

The proposed solution uses XML-RPC based web-services on the server side of the cloud service. It provides necessary client library to use Java for programming the web-service with XML-RPC, the data serialization methods are powerful enough to transfer data related to the design and to refactoring processes. Note that the XML-RPC based solution can easily be adapted and run in multiple environments and operating systems. The architecture of the web-service that serves software refactoring component features over web to clients is designed as a client/server communication model. At the server side of the web-service architecture, there is a standalone refactoring component instance from the client side. That component's instance waits for clients' requests over XML-RPC web-service technology and a design (any zip file containing the bin folder of any Java project) can be sent through Internet. When a design is received from a client by the server, server reads the design and creates a "context" for it. Context has a unique ID number. From this point, client has to notify the server with that ID number while establishing any communications and by this way among requests state can be maintained. The web-service provides all methods for extracting the design information, starting the refactoring phase and obtaining the results of the refactoring process. At the client side of the web-service architecture, all these methods can be called and the related information, according to user actions, can be obtained from the server. Then, the related information results are displayed (reflected) to user via graphical user interface during communication. The architecture is fully compatible with known cloud computing methods. Because the cloud computing providers supply virtual servers that can elastically be increased or decreased, multiple servers can be used. However, we do not create a cloud based solution at this point. Our implementation creates a basis for service oriented architecture.

2.2 Supported Functionalities

The client side of the web-service based refactoring engine provides a graphical user interface to interact with the remote server. So, we have decided to create a web site using PHP, Java Script, and HTML for implementation and CSS is used for design of the web application interface. A summary of the proposed web-service functionalities and related details as given below:

- All predefined and necessary methods for serving refactoring component features to clients can be called from the server with sending XML-RPC requests for PHP including related method names and parameters.
- Connection is initiated with the server when a user starts a session. At the beginning, server creates a unique context id for each user, and then this context id is stored for future use of the client demands.
- Any Java based software (a zip file containing all the compiled ".class" files of the Java project) for refactoring can be uploaded in order to start refactoring process at the initial page of web-service client. We used "base 64 encoding" for sending the uploaded design in order to encode the binary

data and ensure that it remains interaction without modification during transport. It is the most critical part since XML-RPC encapsulates the data to XML during transport so that the server can read the design incorrectly. Also, client's e-mail address is asked in order to send a notification about completion of the refactoring process and a link that client can see the results of the refactoring process details.

- Metrics table that gets predefined available metrics from the server can be shown in order to choose and keep the needed ones for refactoring process from the metric page. Moreover, each metric description can be seen when mouse is on the related metric information icon.
- Actions table that gets predefined available actions from the server can be shown in order to choose and keep needed ones for refactoring process from the action page. Moreover, each action description can be seen when mouse is on the related action information icon. Following the configurations of metrics & actions, Artificial Bee Colony parameters are fixed to desired values. After that, a unique "run id" is given to the client and it is placed in the "past/current runs" table with its starting "date", parameter values & run "status". Client can select more than one search requests which are queued as the next refactoring process. Each time the client start new refactoring, a new unique "run id" is given for the same client.
- Finally, when a refactoring process is finished an e-mail is sent to client's account showing (also, a "result button" appears in the "past/current runs" table when the refactoring is completed as its status) the details of the refactoring process such as run info, time taken, expanded designs, initial metric score, final metric score, quality gain, applied refactoring actions, a line chart for run progress of the refactoring action over time and a column chart for the change in metrics after the run. Note that if the connection cannot be established while initiating a connection to the server, an error page is shown to inform the client.

2.3 Assumptions and Limitations

The proposed web-service solution uses a Java bytecode manipulation and analysis framework ASM [13]. The framework is being used to evaluate compiled Java code in bytecode level, in order to extract design information from arbitrary Java software. Since ASM is operating in bytecode level, it is possible to extract high level design information, such as classes, methods, packages etc. along with low level implementation details like call dependencies, visibilities, method bodies and such. All abstract design representations of given benchmark programs has been constructed using ASM framework. The utilized design representation schema favors look-ups over modifications in terms of performance. The implemented 20 refactoring actions are: Move Up Method, Move Down Method, Move Method, Move Up Field, Move Down Field, Instantiate Method, Freeze Method, Make Class Abstract, Make Class Final, Make Class Non-Final, Inline Method, Remove Class, Remove Interface, Remove Method, Remove Field, Introduce Factory, Increase Method Security, Decrease Method Security, Increase Field Security and Decrease Field Security. The reversibility of the refactoring actions provides both bad and good design space

movements that lead to change in the quality of the design based on the results of the metrics evaluation. The reversible actions include Move-Up Method & Move-Down Method; Move-Up Field & Move-Down Field; Increase Method Security & Decrease Method Security; Increase Field Security & Decrease Field Security; Make Class Final & Make Class Non-Final.

Each refactoring action type has a checker function that can evaluate a given design to find all appropriate actions of the given type that can be applied to the design. In order to preserve the functionality of a given design, the checker has to iterate over all possible operations that looks possible at the first pass and performs a static analysis to ensure the action is legal in the source programming language and the run time behavior is guaranteed to be preserved. The static analysis method consists of several condition checks on a possible refactoring action that results in a filtering over actions. The service uses an aggregate of 24 object-oriented metrics selected from various sources including [14] and [15]. The aggregate does not consider any weight over adopted metrics and the related search is done on normalized metric values. Widely accepted software engineering practices suggests low complexity in order to increase maintainability [14]. As a consequence, we need to minimize complexity related metrics during the search for better design. On the other hand, we cannot say that increase in static methods of a class is good (or bad) to improve overall design quality. Since the optimization process cannot rely on subjective opinions, objectives for such metrics can be regarded as "unknown". As a consequence, we categorize each considered metrics either as "minimized" or "unknown" (see Table 1). Furthermore, those metrics that require value maximization are also treated as minimized metrics by their negated values.

In the normalization schema, instead of setting precise objectives for unknown metrics, we prefer to use the calculated metric values of a predetermined ideal design set as a feedback mechanism. The design under evaluation is compared against the norm of a supposed optimum design set and the distances are used as a metric score to be minimized during search. As the idea is to approach to the norm of optimum design set for unknown metrics, their distances are calculated against the mean value of the standard normal distribution. On the other hand, minimized metric distances are required to be calculated against the absolute 0. To achieve this, for such metrics, the normalized value of absolute 0 is used as the objective and the distance is calculated accordingly. Given metrics $M_1, M_2,\ldots,M_k$, ideal design set $D_1, D_2,\ldots, D_n$ and current design D_{cur}. The Distance function is defined as

$$\begin{aligned} Dist(M_i, D_{cur}) = {} & |NormVal(M_i(D_{cur}), i) - NormVal(0, i)|, && M_i \in Min. \\ & |NormVal(M_i(D_{cur}), i)|, && M_i \in Un. \end{aligned} \quad (1)$$

where Min. stands for the set of minimizing metrics, Un. stands for the set of unknown metrics and NormalVal calculates the normalized value of x against the values of i^{th} metric in the ideal design set such that

$$NormVal(x, i) = Norm_{0\text{-}1}(M_i(D_1), M_i(D_2), \ldots, M_i(D_n))_x \quad (2)$$

The overall evaluated metric score of D_{cur} is defined as

$$Eval(D_{cur}) = \sum_{i=1}^{k} Dist(M_i, D_{cur}) \quad (3)$$

Besides from describing objective function, we need to describe the constraints that apply to the problem domain. They are the requirements of behavior preservation of the bytecode design under optimization. Simply, each of different functional requirements implemented over the code defines a constraint. Note that the state variables do not take their values from a continuous domain but from a set which is described by possible discrete refactoring actions through the search process, dynamically.

Table 1. List of considered metrics

Minimized	Unknown
Number of fields in a class; Number of methods in a class; **Average number of the field visibility of a class;** Average number of methods visibility of a class; **Nesting level of a class;** Number of methods of a class in a package; **Number of classes in a package;** Number of interfaces in a package.	Number of constant fields in a class; **Number of setter methods in a class;** Number of getter methods in a class; **Average number of the static methods of a class;** Number of interfaces a class implements; **Number of children of a class;** Number of descendants of a class; **Number of ancestors of a class;** Number of elements on which this class depends; **Number of elements that depend on this class;** Number of times the class is externally used as attribute type; **Number of attributes in the class having another class or interface as their type;** Number of times the class is externally used as parameter type; **Number of parameters in the class having another class or interface as their type;** Nesting level of a package; **Ratio of abstract classes in a package.**

3 Application of ABC Search to the Domain

The software refactoring domain requires the usage of not continuous but discrete variation of ABC (i.e. DABC). As a consequence, in our implementation, we applied the discrete version of the algorithm designed for solving combinatorial optimization problems. Mathematical description of the known DABC search procedure can be found in [16]. However, our domain specific implementation requires the below adaptations:

- Initial food sources are being generated from the initial design by applying random refactoring actions. For each food source, a predefined number of random actions are carried out iteratively to generate pseudo random solutions. Note that the original algorithm creates random food sources within the boundaries of search space.
- In the algorithm, the depleted food sources are required to be replaced by random sources. However, our application requires producing a random solution while keeping the design functionality of the input program. Two possible solutions to this problem are, either creating a random food source from the initial design, or to consider the current best design. Using the current

best design as a starting point for creating random solutions has been decided to be more effective and the replacement approach has been implemented based on this assumption. Just like the initial population generation, all depleted food sources are replaced by pseudo random food sources, which are created from best solution by carrying out a predefined number of random refactoring actions on it.

- In its discrete search space, our solution finds the neighboring food sources by applying a single refactoring action to the current food source. Therefore, each random change represents a movement within search space onto a nearby food source.

While the adaptation is influenced by the discrete version, it was not possible to directly implement the given algorithm in refactoring domain. The DABC algorithm applies a preliminary local search to mutated food sources during employed and onlooker bee phases. Even though this nectar boosting procedure can be implemented in theory, it causes significant performance problems in refactoring domain, due to complexity and vast numbers of mutation possibilities of each solution. Another difference from the DABC algorithm is related to probabilistic selection of food sources by onlooker bees. While the original algorithm performs individual probability checks on each food source, the discrete version utilizes a tournament selection procedure by choosing two random food sources. However, it has been decided to adopt the original algorithm where possible, therefore, our solution performs individual probability checks for food source selection. Apart from these differences, the required modifications for discretization (such as initialization of food sources, mutations, using mutants from current best etc.) have been based on the DABC algorithm.

4 Conclusions

The study realized an automated software refactoring web-service based on parallel implementation of discrete artificial bee colony optimization technique. Due to the added parallelism, we were able to investigate different regions of design space defined by the proposed objective function. All searches were executed at the design level of abstraction and realized at the design space. The service supports 20 different refactoring actions that realize ABC searches on design landscape that are defined by an adhoc quality model being aggregation of 24 object-oriented software metrics. It produces a sequence of recommended refactoring actions together with code improvement history. For the time being, the design under evaluation is compared against the norm of a supposed optimum design set defined by 4 packages from base Java library. In future, the optimum design set can be enhanced by alternative choices. Finally, the input programs under consideration were limited in terms of number of classes and line of codes. This is mainly due to the high performance requirement of the problem under study. It is clear that some lightweight and efficient design representation still leads to design improvements for more complex and bigger input programs in terms of number of classes to handled.

Acknowledgments. This study has been realized under partial support of Scientific Research Project (SRP) Unit of Atilim University, Ankara, Turkey. So, the authors would like to thank to Atilim University.

References

1. Karaboga, D.: An idea based on honey bee swarm for numerical optimization, Technical Report, TR-06, Erciyes University, Kayseri, Turkey (2005)
2. Karaboga, D., Basturk, B.: A powerful and efficient algorithm for numerical function optimization: artificial bee colony algorithm. Journal of Global Optimization 39, 459–471 (2007)
3. Bouktif, S., Antoniol, G., Merlo, E., Neteler, M.: A novel approach to optimize clone refactoring activity. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, GECCO 2006, vol. 2, pp. 1885–1892. ACM Press, WA (2006)
4. Seng, O., Stammel, J., Burkhart, D.: Search-based determination of refactorings for improving the class structure of object-oriented systems. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, GECCO 2006, vol. 2, pp. 1909–1916. ACM Press, WA (2006)
5. Harman, M., Tratt, L.: Pareto optimal search based refactoring at the design level. In: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, GECCO 2007, pp. 1106–1113. ACM Press, NY (2007)
6. O'Keeffe, M., Cinneide, M.O.: Search based refactoring: an empirical study. Journal of Software Maintenance and Evolution: Research and Practice (2), 345–364 (2008)
7. Moghadam, I.H.: Multi-level Automated Refactoring Using Design Exploration. In: Cohen, M.B., Ó Cinnéide, M. (eds.) SSBSE 2011. LNCS, vol. 6956, pp. 70–75. Springer, Heidelberg (2011)
8. Koc, E., Ersoy, N., Camlidere, Z.S., Andac, A., Cereci, I., Kilic, H.: An empirical study about search-based refactoring using alternative multiple and population-based search techniques. In: Computer and Information Sciences II - Proceedings of 26th International Symposium on Computer and Information Sciences, ISCIS 2011, London, UK, pp. 59–66. Springer (2011)
9. Kilic, H., Koc, E., Cereci, I.: Search-Based Parallel Refactoring Using Population-Based Direct Approaches. In: Cohen, M.B., Ó Cinnéide, M. (eds.) SSBSE 2011. LNCS, vol. 6956, pp. 271–272. Springer, Heidelberg (2011)
10. Harman, M.: The current state and future of search based software engineering. In: Proceedings of Future of Software Engineering, FOSE 2007, pp. 342–357. IEEE Press, WA (2007)
11. Papazoglou, M.P., Traverso, P., Dustdar, S., Leymann, F.: Service-oriented computing. Communications of the ACM 46, 25–28 (2003)
12. XML Basics. XML News, `http://www.xmlnews.org/docs/xml-basics.html` (last accessed on February 2012)
13. OW2 Consortium, ASM, `http://asm.ow2.org/`, (last accessed on February 2012)
14. Chidamber, S.R., Kemerer, C.F.: A metrics suite for object oriented design. IEEE Trans. on Soft. Eng. 20, 476–493 (1994)
15. SDMETRICS tool, `http://www.sdmetrics.com/` (last accessed on February 2012)
16. Pan, Q., Tasgetiren, M.F., Suganthan, P.N., Chua, T.J.: A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem. Information Sciences 181(12), 2455–2468 (2011)

An Improved Artificial Bee Colony Algorithm Based on Gaussian Mutation and Chaos Disturbance

Xiaoya Cheng and Mingyan Jiang

School of Information Science and Engineering, Shandong University
250100 Jinan, China
chengxiaoya@mail.sdu.edu.cn, jiangmingyan@sdu.edu.cn

Abstract. Artificial Bee Colony (ABC) algorithm is a novel bio-inspired swarm intelligence approach which is competitive with other population-based algorithms and has the advantage of using fewer control parameters. However, basic ABC is easy to be prematurely convergent and be trapped into local optimum. In the later iteration, algorithm has low convergent speed and population diversity seriously decreases. In this paper, Gaussian mutation and chaos disturbance are introduced into ABC to overcome the shortcomings above. Applications of improved ABC algorithm on four benchmark optimization functions show marked improvement in performance over the basic ABC.

Keywords: Artificial Bee Colony (ABC) algorithm, Gaussian mutation, Chaos disturbance.

1 Introduction

Nowadays, there is a trend in the scientific community to model and solve complex optimization problems by employing swarm intelligence in nature. Artificial Bee Colony (ABC) algorithm is one of the most recently defined algorithms by Karaboga in 2005, motivated by the intelligent behavior of honey bees [1]. It is as simple as Particle Swarm Optimization (PSO) and Differential Evolution (DE) algorithms, and uses only common control parameters such as colony size and maximum cycle number. ABC algorithm is very simple and very flexible, compared to the existing swarm intelligence algorithms [2].

Due to its simplicity and easy implementation, ABC algorithm has captured much attention and has been applied to solve many practical optimization problems. Mustafa applied ABC algorithm for the truss structures optimization problems [3]. T.-J. Hsieh et al. presented an integrated system where wavelet transforms and recurrent neural network (RNN) based on ABC algorithm are combined to forecast stock markets [4]. ABC algorithm was also applied in clustering [5, 6], Chunfan Xu proposed an ABC algorithm with edge potential function to visual target recognition for aircraft at low altitude [7]. Singh applied ABC algorithm for the Leaf-Constrained Minimum Spanning Tree (LCMST) problem [8]. Karaboga N. used ABC algorithm to design Infinite Impulse Response (IIR) filters [9].

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 326–333, 2012.

As a new search algorithm, a lot of researches have gone into improving the performance. Bao Li and Haijun Ding used tournament selection [10] and boltzmann selection [11] to instead of roulette wheel selection of ABC algorithm, a hybrid version of the algorithm [12] has also been proposed. Chunfang Xu proposed an improved ABC optimization algorithm based on chaos theory to solve the Uninhabited Combat Air Vehicle (UCAV) path planning in various combat field environments [13].These methods reduce the possibility of local optimum to some extent.

In this paper, an improved ABC algorithm has been proposed. In the local search, Gaussian mutation is carried out for improving the searching efficiency and precision. Considered the characteristics of ergodicity and randomness of chaotic variables, chaos disturbance is introduced into basic ABC, which is helpful for bees to jump out of local optimum and increases the population diversity.

2 Basic ABC Algorithm

In ABC, the colony of artificial bees contains three groups of bees: employed bees associated with specific food sources, onlooker bees watching the dance of employed bees within the hive to choose a food source, and scout bees searching for food sources randomly. Both onlookers and scouts are also called unemployed bees. Initially, all food source positions are discovered by scout bees. Thereafter, the nectar of food sources are exploited by employed bees and onlooker bees, and the continual exploitation will ultimately cause them to become exhausted. Then, the employed bee which was exploiting the exhausted food source becomes a scout bee in search of further food sources once again. In ABC, the position of a food source represents a possible solution to the problem and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution. The number of employed bees is equal to the number of food sources (solutions) since each employed bee is associated with one and only one food source. The details can be described as follows [14].

$$\min\ f = f(x),\ x = (x_1, x_2, ..., x_m) \in S,\ \ S = [x_{iL}, x_{iH}]\ . \tag{1}$$

where, f represents the objective function, x is m -dimensional variable, $[x_{iL}, x_{iH}]$ indicates the upper and lower bounds of the i th-dimensional variable.

Suppose that the number of the employed bees and onlooker bees all is N . The main steps of the algorithm are given below:

a) Initialization

Produce $2N$ locations randomly, evaluate them and move the employed bees onto the N food sources with the more nectar amounts.

b) Employed bees explore new food sources around themselves by the (2)

$$V_{ij} = x_{ij} + R_{ij}(x_{ij} - x_{kj})\ . \tag{2}$$

where V_{ij} is a new location, R_{ij} is a random number in the range [-1,1], $k \in \{1,2,3,...,N\}$ and $k \neq i$.

c) Mark the N food sources with more nectar amounts from the candidate food sources between a) and b).

d) Onlookers explore new food sources

The onlookers are placed on food sources selected in the roulette wheel selection method. Then each onlooker bee explores the neighborhood of food source x_i as (2). Probability P_i is calculated as follows:

$$P_i = \frac{fit_i}{\sum_{i=1}^{N} fit_i} \quad . \tag{3}$$

here, fit_i is calculated using the following equation:

$$fit_i = \begin{cases} \dfrac{1}{1+f_i}, f_i \geq 0 \\ 1+abs(f_i), f_i < 0 \end{cases} \quad . \tag{4}$$

where, f_i is the fitness value of the solution.

e) If a solution cannot be improved for "limit" trials, it will be abandoned. The scout randomly produces a solution to replace the old one.

f) Select the N better solutions between candidate solutions generated in step c) and step d).

g) Record the best solution obtained till now and repeat step b) to f) until the max iterations.

3 Improved ABC Algorithm

Basic ABC algorithm is a simple, robust, and easily be controlled algorithm. However, as a random optimization algorithm, ABC algorithm has slow convergence characteristics and easily gets stuck on local solutions. In this paper, basic ABC algorithm is modified to get better optimization value.

3.1 Gaussian Mutation

Gaussian mutation consists in adding a random value from a Gaussian distribution to each element of an individual's vector to create a new offspring. This technique employs the following equation:

$$mutation(x) = x*(1+N(0,1)) \quad . \tag{5}$$

x is a numerical value which each object has. $N(0,1)$ is a random value extracted from a Gaussian (normal) distribution. $mutation(x)$ is the new value after Gaussian mutation for an individual.

The individuals are selected at the predetermined probability and their positions are determined at the probability under the Gaussian distribution. Wide-ranging searches are possible at the local search stage and searching efficiency is improved at the final stage by using Gaussian mutation in improved ABC algorithm.

3.2 Chaos Disturbance

When the bees trap into local optimum, chaos disturbance is introduced to help them overcome the prematurity.

Chaos is a common nonlinear phenomenon in our lives. The dynamic properties of chaos can be shown as following: (1) Chaos is highly sensitive to the initial value. (2) Certainty: Chaos is produced by the certain iterative formula. (3)Ergodicity: Chaos can go through all states in certain ranges without repetition.

Chaos is similar to randomness. But experimental studies assert that the benefits of using chaotic signals instead of random signals are often evident even though a general rule cannot be formulated [15]. So the performance of chaotic search is better than random search. Due to the easy implementation and special ability to avoid being trapped in local optima, chaos has been a novel optimization technique and chaos-based searching algorithms have aroused intense interests [16].

Chaos optimization is realized through chaos variables which can be obtained by many ways. Here the Tent Mapping method [17] is selected. The equation of Tent Mapping is shown as follows:

$$X_d = \begin{cases} 2 * x_d & 0 \le x_d \le 1/2 \\ 2 * (1 - x_d) & 1/2 \le x_d \le 1 \end{cases} \quad d = 1, 2, ..., D \quad . \tag{6}$$

X_d is the d-th dimensional chaos variable; x_d is a random uniformly distributed variable, $x_d \in [0,1]$.

Chaos disturbance includes the following major steps:

(1) Produce chaotic variables using chaotic mapping.
(2) The chaotic variable is mapped back to the solution space.

$$newX_d = \min_d + (\max_d - \min_d) * X_d \quad . \tag{7}$$

$\max_d$ and $\min_d$ are the maximum and minimum value of the d-th dimensional variable, respectively.

(3) Chaos disturbance is carried out by the following formula:

$$newX' = (X' + newX) / 2 \quad . \tag{8}$$

X' is the individual which needs to be chaos disturbed; $newX$ is the chaos disturbance variable, $newX'$ is the new individual which has been chaos disturbed.

3.3 Flow of Improved ABC Algorithm

Improved ABC algorithm with Gaussian mutation and chaos disturbance can be summarized as follows:

Step1. Generate the initial population (positions of food source).
Step2. Employed bees depend on the formula (2) to explore new food sources called offspring.

Step3. Compare the fitness value of every food source with its corresponding offspring and mark location of the better performance.
Step4. Onlookers depend on roulette wheel selection method to choose food source obtained in Step3 and explore new position around it, called offspring population.
Step5. Select the better performance food sources between marked locations and offspring population as true food sources, move employed bees to these locations.
Step6. Calculate the mean fitness value of all food sources. For all bees, if its fitness value is superior to the mean fitness value, Gaussian mutation is carried out for the corresponding bee; otherwise, chaos disturbance is applied.
Step7. Judge whether the terminating condition is satisfied. If not, go to Step2, otherwise output the optimal solution and end.

4 Function Optimization Experiments

In order to evaluate the performance of improved ABC algorithm, a series of experiments on four classical benchmark functions are carried out to compare improved algorithm with basic ABC algorithm. The benchmark functions used in the experiments are listed in Table 1.

Table 1. The benchmark functions

Function	Formulation	Range
Griewank	$f_1 = \frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} \cos(\frac{x_i}{i^{1/2}}) + 1$	[-100, 100]
Rastrigin	$f_2 = \sum_{i=1}^{d} [x_i^2 - 10\cos(2\pi x_i) + 10]$	[-100, 100]
Rosenbrock	$f_3 = \sum_{i=1}^{d-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	[-100, 100]
Sphere	$f_4 = \sum_{i=1}^{d} x_i^2$	[-100, 100]

In the experiments, the Dimension was set to 100, the number of population (NP) was 20, the value of the limit parameter was 100 and the maximum cycle number (MCN) was 2500. Looking for the minimum of four test functions is the content of experiments. In theory, the minimum is all 0. Each experiment was repeated 10 times, and the mean best function values are presented in Table 2. In order to show the performance and convergent speed of the algorithms more clearly, Fig. 1-4 indicate the progress of the mean best function values in 600 iterations.

As shown in Table 2, for four test functions, the optimal function value of improved ABC is better than basic ABC. The convergent speed is an important indicator to the algorithm. From Fig. 1-4, it is shown that the improved ABC has the higher convergent speed.

For basic ABC, bees search depends on the equation (2) in the neighborhood, which evolved step by step. However, in improved ABC, after one iteration completed at the beginning, superior bees improve by Gaussian mutation and inferior

Table 2. The mean best function values

Function	The optimal value	
	Basic ABC	Improved ABC
Griewank	1.6×10^{-6}	0
Rastrigin	8.6457	0.00187
Rosenbrock	225.84	98.9884
Sphere	3.1×10^{-7}	3.0×10^{-8}

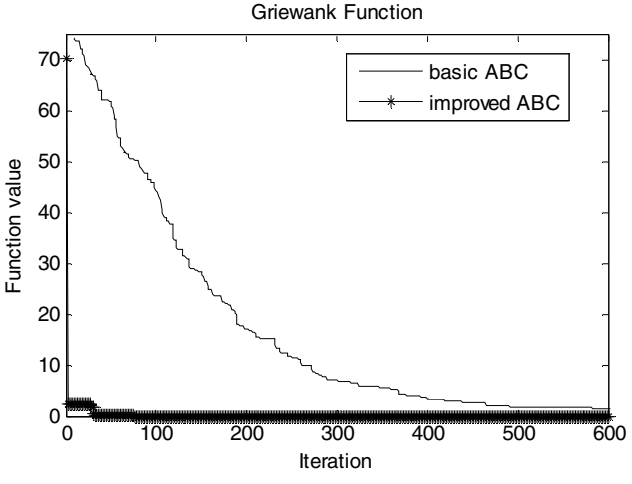


Fig. 1. The convergence curve of Griewank Function

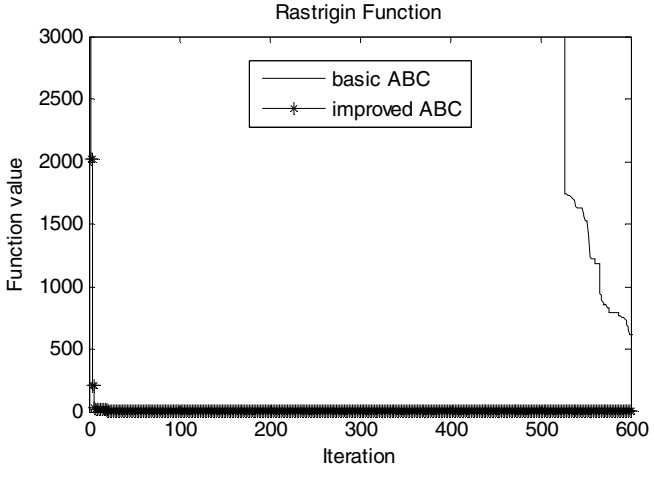


Fig. 2. The convergence curve of Rastrigin Function

bees change by chaos disturbance. It is salutatory and fast. So the convergent speed of improved ABC progresses very rapidly in the first period. Moreover, the better results can be obtained at the final stage.

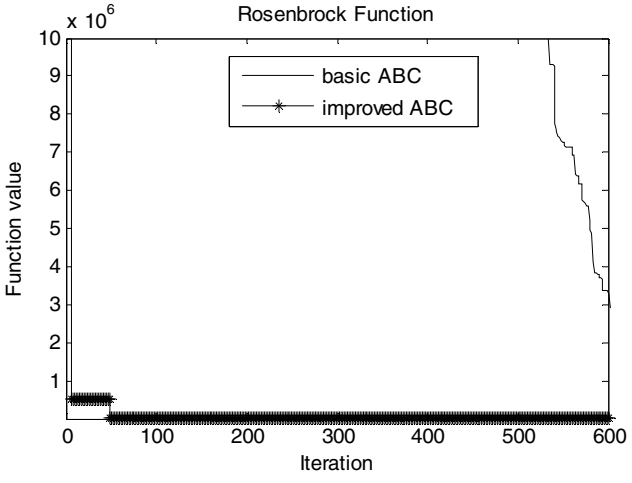


Fig. 3. The convergence curve of Rosenbrock Function

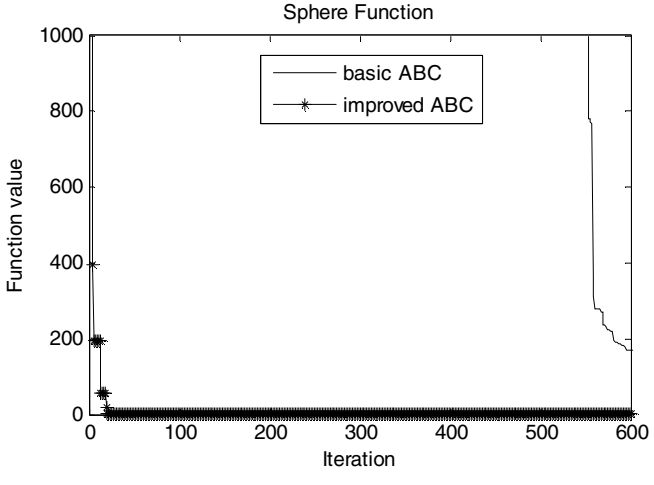


Fig. 4. The convergence curve of Sphere Function

5 Conclusions

In this paper, Artificial Bee Colony (ABC) algorithm was studied. Observing basic ABC algorithm's shortcomings, an improved ABC algorithm with Gaussian mutation and chaos disturbance has been proposed. Simulation results on a set of benchmark functions indicate that the searching properties including searching efficiency and precision of improved ABC algorithm are obviously better than that of basic ABC algorithm.

Acknowledgments. This work is supported by the Natural Science Foundation of Shandong Province (No. ZR2010FM040), Shandong Provincial key project (No.2009ZHZX1A0108, No.2010ZHZX1A1001).

References

1. Karaboga, D.: An Artificial Bee Colony (ABC) Algorithm for Numeric Function Optimization. In: IEEE Swarm Intelligence Symposium, pp. 181–184. IEEE Press, Indiana (2006)
2. Jiang, M., Yuan, D.: Artificial Fish Swarm Algorithm and Its Applications. Science Press, Beijing (2012)
3. Sonmez, M.: Artificial Bee Colony Algorithm for Optimization of Truss Structures. Applied Soft Computing Journal 10, 195–197 (2010)
4. Hsieh, T.J., Hsiao, H.F.: Forecasting Stock Markets using Wavelet Transforms and Recurrent Neural Networks: an Integrated System Based on Artificial Bee Colony Algorithm. Applied Soft Computing Journal 10, 156–162 (2010)
5. Karaboga, D.: A Novel Clustering Approach: Artificial Bee Colony (ABC) Algorithm. Applied Soft Computing 11, 652–657 (2011)
6. Zh, C., Ouyang, D., Ning, J.: An Artificial Bee Colony Approach for Clustering. Expert Systems with Applications 37, 4761–4767 (2010)
7. Chunfan, X.: Artificial Bee Colony (ABC) Optimized Edge Potential Function (EPF) Approach to Target Recognition for Low-altitude Aircraft. Pattern Recognition Letters 31, 1759–1772 (2010)
8. Singh, A.: An Artificial Bee Colony Algorithm for the Leaf-constrained Minimum Spanning Tree Problem. Applied Soft Computing 9, 625–631 (2009)
9. Karaboga, N.: A New Design Method Based on Artificial Bee Colony Algorithm for Digital IIR Filters. Journal of the Franklin Institute 346, 328–348 (2009)
10. Li, B., Zeng, J.: Self-adapting Search Space Chaos-artificial Bee Colony Algorithm. Application Research of Computers 27, 1331–1335 (2010)
11. Ding, H., Feng, Q.: Artificial Bee Colony Algorithm Based on Boltzmann Selection Policy. Computer Engineering and Applications 45, 53–55 (2009)
12. Kang, F., Li, J., Xu, Q.: Structural Inverse Analysis by Hybrid Simplex Artificial Neural Networks. In: 15th IEEE Proc. Signal Processing and Communications Applications, pp. 1–4. IEEE Press, SIU (2007)
13. Xu, C.: Chaotic Artificial Bee Colony Approach to Uninhabited Combat Air Vehicle (UCAV) Path Planning. Aerospace Science and Technology 26, 156–162 (2010)
14. Kang, F., Li, J., Li, H.: An Improved Artificial Bee Colony Algorithm. In: 2nd International Workshop on Intelligent Systems and Applications, pp. 15–21. IEEE Press, Wuhan (2010)
15. Bucolo, M., Caponetto, R., Fortuna, L., Frasca, M., Rizzo, A.: Does Chaos Work Better than Noise? Circuits and Systems Magazine 2, 4–19 (2002)
16. Wang, L., Zheng, D., Lin, Q.: Survey on Chaotic Optimization Methods. Comput. Technol. Automat. 20, 1–5 (2001)
17. Li, C., Zhang, X.: Design of Pseudo-random Sequence Generator Based on Chaos Anti-control Tent Map. Journal of Computer Applications 28, 48–51 (2008)

An Artificial Bee Colony Algorithm Approach for Routing in VLSI

Hao Zhang[1,2] and Dongyi Ye[1,2]

[1] Center for Discrete Mathematics and Theoretical Computer Science, Fuzhou University, Fuzhou 350003, China
[2] College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China
{zhanghao,yiedy}@fzu.edu.cn

Abstract. This paper presents an approach that applies the Artificial Bee Colony algorithm to the Two-Terminals-Net-Routing（TTNR） problem in VLSI physical design and compares its performance with the maze algorithm variant known as the state-of-the-art global routing algorithm. An effectively encoding method is described in this paper to solve the TTNR problem. In order to improve the convergence speed of the algorithm, some guiding solutions are employed as the initial solutions. The experimental results demonstrate that Artificial Bee Colony algorithm can find the less cost routing paths for TTNR problems than the maze algorithm.

Keywords: Artificial Bee Colony Algorithm, VLSI physical design, Global Routing, Two-Terminals-Net-Routing.

1 Introduction

Since design of Integrated Circuits went into nanometer regime, the challenges of very/ultra large scale integrated circuits (VLSI/ULSI) physical design have been escalating. Routing of the interconnects plays a more critical role in design cycle that arranges the routes of nets physically. Routing a two terminals net (TTNR), i.e., finding a right angle polyline in the routing region with minimal wirelength is a fundamental problem. TTNR is usually employed in the global routing and detail routing. Depending on the application background, the TTNR issue is divided into three categories: obstacle-free, weighted-path and obstacle-avoiding. In this paper, we focused our attention on the weighted-path routing.

Various routing algorithms have been proposed to solve this problem. Pattern routing is the simplest routing algorithm with the smallest number of bends, the fastest runtime, and the shortest wirelength. Pan et al. (2007) proposed Monotonic routing in [1], which can find the more complex paths than pattern routing. Maze routing algorithm is a kind of the slowest approach and is the most effective way for the extremely complex routing problem. Multi-source multi-sink maze routing (Pan 2007) and adaptive multi-source multi-sink maze routing [2] (Chang 2008) are the latest variants of mazing routing algorithm which consider the resource sharing in

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 334–341, 2012.

multi terminals net global routing. Heuristic and meta-heuristic algorithms such as boxed A*-search [3], the particle swarm optimization approach [4], the ant colony optimization based algorithm [5] and the genetic algorithm have also been presentd to solve this issue.

Artificial bee colony algorithm is a kind of bee swarm algorithm described by Karaboga (2005) which simulates the nectar behavior of honeybee swarm [7]. The performance of the ABC (standard version) is better than or similar to the standard versions of genetic algorithm, particle swarm optimization algorithm, and differential evolution algorithm and evolution strategies for multi-dimensional numeric problems [8]. To the best of our knowledge, none of these studies can achieve minimum wirelength with acceptable runtime for TTNR when the bend cost is considered. The primary focus of the present paper is on proposing an advanced Artificial Bees Colony optimization algorithm to improve the efficiency and wirelength with bend cost.

The rest of the paper is organized as follows: TTNR in VLSI is presented in Section 2. The overview of Artificial Bee Colony algorithm is introduced in the third section. Section 4 discusses the problem modeling and algorithm procedure of ABC-TTNR. Section 5 presents the simulation results and finally section 6 concludes this paper.

2 Two Terminals Net Routing (TTNR) Problem of Global Routing in VLSI

Weighted-path routing of two terminals net is a basic problem in global routing of VLSI physical design which is used to connect to two terminals with the least congestion cost and overflows-free path in the grid weighted graph. Not only every edge in the grid graph of global routing is weighted, but also the bends in the path are weighted for minimizing both wirelength between tiles and via usage between layers. So that the cost of a path is the sum of the edges' cost and the bends' cost. In this paper, we used the congestion cost function proposed in NTHU-2.0[9] to assign cost for every edge and bend. Expression (1) shows the full cost of the path P.

$$Cost(P) = \sum_{e \in E_P} \cos t_e + num * c_b \tag{1}$$

where E_P represents the edges set of the path P. $\cos t_e$ denotes the congestion cost of the edge e. num is the number of the bends which are employed in path P. c_b is a constant that expressed the cost of each bend.

3 Meta-Heuristic Algorithm of Artificial Bee Colony

Inspired by the collective behavior of the insect colonies, the swarm intelligence model of artificial bee colony was proposed by KARABOGA in [7]. Food sources around the hive represent the solutions in the solution space. The quantity of the food source indicates the quality (fitness) of the solution. The artificial bee colony contains three groups of artificial bees: employed bees, onlookers and scout bees. The job of the scouts in the algorithm is finding the stochastic solutions in the solution space. Employed bees recommend the present solutions to each onlooker which can chose a

solution based on the fitness and search the neighborhood of this solution for a better one. If a present solution cannot get improvement and the searching time has exceeded the bounding number, the solution is exhausted. Size of the population which is denoted by SP. SN is the number of the employed bees which the half of the SP and the other half of the population is consisted of onlookers. Each employed bee is associated with a solution (food source) and the number of present solutions is SN too. A bounding number of searching each solution's neighborhoods is set as limit. MCN is the maximum cycle number of the algorithm for an optimization problem. Fitness is defined as the quality of the solution which is closely related to the objective function. The number of the scout bees is set one in the algorithm.

The scout bee plays a global stochastic search role. Global search strategy is generating random solutions in solution space. Random function rand (0, 1) is invited which can generate a random real value in the open interval range of zero to one. The j th entry z_i^j of a random solution z_i which can be calculated by Equation (2).

$$z_i^j = z_{\min}^j + rand(0,1)*(z_{\max}^j - z_{\min}^j),\ \ \mathrm{j}\in\{1,2,...,\mathrm{D}\},\ \ i\in\{1,2...,SN\} \tag{2}$$

The onlooker and employed bee play a local search role around the relative present solution. Local search strategy is according to the associated solution z_i of the employed bee and the solution associated a random employed bee z_k . The j th entry v_i^j of the neighborhood solution v_i can be calculated by Equation (3). If the entry exceeds the bound, the bound value is accepted as the entry value. In this process, z_i is the position of the employed bee, $|z_k - z_i|$ is defined as the visual range around z_i .

$$v_i^j = z_i^j + rand(-1,1)*(z_i^j - z_k^j),\quad \mathrm{j}\in\{1,2,...,\mathrm{D}\},\ \ i,k\in\{1,2...,SN\} \tag{3}$$

The attractiveness probability of every present solution to the onlookers is calculated as Equation (4) where fit_i is the fitness of i th solution.

$$p_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \tag{4}$$

According to the above description, it is clear that ABC algorithm contains a few parameters which are used to control the size and characteristics. Furthermore, the global search capacity of ABC is especially excellent. In addition, ABC can be easily adopted to solve various optimization problems with a little adjustment.

4 ABC Algorithm for TTNR (ABC-TTNR)

4.1 Modeling in ABC

In order to adapt the ABC algorithm for solving TTNR problem, several parts of ABC should be altered such as: 1) Coding for the TTNR optimization problems, 2) The strategy of the initializing solutions set, 3) The local searching operator, 4) The dynamic attractiveness probability, 5) Discretization algorithm method.

A sequence encoding method is employed in ABC-TTNR, which divides the solutions of the problem into two parts: the row-based solutions and the column-based solutions. For example, the grey path belongs to a row-based solution, and the sequence can be expressed as $(1, y_s, y_s+1, \ y_s+1, y_s+1, y_s+3)$. The entries in the sequence are the y-coordinates of the used horizontal edges in the solution path except the first entry, which is a flag for distinguishing the row-based or column-based solution. The length of the sequence of the row-based solution is the difference of the y_t and y_s which denote with constant R (constant C for column-based solution) in expression (8). EXT denotes a constant of the range for extending the routing bounding box. The column-based solution is encoded in the same way.

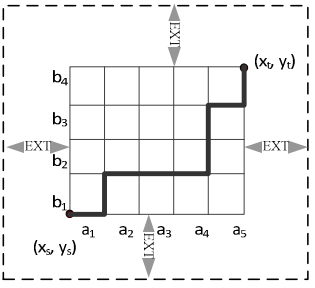


Fig. 1. Encoding for the ABC

The row-based solutions and column-based solutions are defined in expression (5). The present solutions set is consisted of the row-based solutions set and the column-based solutions set which sizes are denoted with SN_r and SN_c respectively.

$$
\begin{aligned}
&z_{ri} = (1, a_{i1}, a_{i2} \ldots\ldots a_{iR}) \in Z^n, R = |x_t - x_s|, a_i \in [-EXT, y_{\max} + EXT]_z, i \in [1, SN_r]_z \\
&z_{cj} = (0, b_{j1}, b_{j2} \ldots\ldots b_{jC}) \in Z^n, C = |y_t - y_s|, b_j \in [-EXT, x_{\max} + EXT]_z, j \in [1, SN_c]_z \\
&SN_r = \frac{R}{C+R} * SN, \qquad SN = SN_r + SN_c = 2*(C+R+2*EXT)
\end{aligned} \tag{5}
$$

Pattern routing ways are involved in the initializing solutions set in the initialization phase, which are nearby good solutions for the problem. Half of the initializing solutions are generated by pattern routing. The other half of the solutions set is produced stochastically in the solution space. The novel initializing strategy can improve the probability of finding the optimal solution and reduce the convergence time.

Local searching operator is replaced with a sequence operation which is divided into three steps. Firstly, an auxiliary solution is randomly chosen in the present solutions set except the current solution (the employed bee associated) which is

selected for searching. The auxiliary solution is the same base with the current solution. Secondly, a random entry number m of the sequence is chosen. The auxiliary solution and current solution are split into front part and behind part. Two new solutions are generated by exchanging the two front part sequences or behind part sequences. Finally, the greedy selection process is used to select the best solution among the new solutions and current solution as the updated current solution.

$$\begin{aligned} z_{current} &= (0, a^1, a^2 \ldots, a^m, \ldots a^C), \qquad z_{auxiliary} = (0, b^1, b^2 \ldots, b^m, \ldots b^C) \\ z_{new1} &= (0, a^1, a^2 \ldots, b^m, \ldots b^C), \qquad z_{new2} = (0, b^1, b^2 \ldots, a^m, \ldots a^C) \end{aligned} \tag{6}$$

A dynamic attractiveness probability is designed in expression (7) for adjusting the positive feedback of this algorithm during the different stages. At the beginning, the differences among the attractiveness probabilities of the present solutions are small which can avoid that all the onlookers are attracted by the best present solution and prevent from the algorithms premature. In latter stage, the differences among the attractiveness probabilities of the present solutions could be widened out which can help to convergence to the optimal solution in a short time. Fitness function fit_i in TTNR problem is defined as the reciprocal of the cost of the solution path.

$$p_i = \alpha * (\frac{fit_i}{fit_{\max}} - 1) + 1, \qquad \alpha = 0.6 * (\frac{cycle}{MCN})^2 + 0.3 \tag{7}$$

The global search operator should be lightly adjusted to meet the needs of discrete optimization problem. The entries of a sequence denote a coordinate which should be integers and an operation of taking integers downwardly is employed in the global search operator which is showed in Equation (8).

$$z_i^j = \left\lfloor z_{\min}^j + rand(0,1) * (z_{\max}^j - z_{\min}^j) \right\rfloor, \ \mathrm{j} \in \{1,2,...,\mathrm{D}\}, \ i \in \{1,2...,SN\} \tag{8}$$

4.2 Processes of ABC-TTNR

Detailed pseudo code of ABC-TTNR:

```
Load sample for routing
Initialize the population of row-based and column-based
solutions z_ri, z_cj, i=1...SN_r, j=1...SN_c
Evaluate the fitness population
cycle=1
repeat
    for each employed bee {
      Produce two new solutions by using (6)
      Calculate the fitness values of the new solutions
      Apply greedy selection process }
    Calculate the probability value p_i for the present
solutions by using (7)
```

```
    for each onlooker bee {
      Select a current solution w_i depending on p_i
      Produce two new solutions by using (6)
      Calculate the fitness values of the solutions
      Apply greedy selection process }
    If there is an abandoned solution for the scout then
replace it with a new solution which will be randomly
produced by (8) with the same base
Memorize the best solution so far
cycle = cycle + 1
until  cycle = MCN
```

5 Simulation Results

Eight TTNR problems with congestion costs are selected randomly which are generated in NTHU2.0. We cannot find the least cost paths for these examples with the variant of maze algorithm AMMMR and less cost paths can be found by the ABC-TTNR. Two simulation results are shown in Fig.2. “pin 1” and “pin 2” represent the coordinates of the two terminals in the net. “Cost” represents the congestion cost of the solution path, “Distance” represents the Manhattan distance of the solution path and “Bendnum” represents the number of the bends in the solution path. As shown in Fig.2. , the pins and bends are denoted by the solid squares. The interconnection schemes found by the AMMM are represented by the grey dotted lines. The schemes found by the ABC-TTNR are represented by the black lines. The coding sequences of the solutions are shown in Table.1, the numbers in the bracket represent the times of related parameter in the sequence. The comparison between solutions’ cost of ABC-TTNR and AMMM are showed in Table. 1.

The parameter EXT of ABC-TTNR is set to 20. Limit is the product of SN and the maximum of the C and R. Each example setting was manipulated for 100 runs. The minimum costs found by ABC-TTNR for examples are shown in Table.1.

We can see from Fig.2. that the ABC-TTNR can get better solutions than maze algorithm.

Table 1. The Solution Sequences of ABC-TTNR and Comparison with AMMM

Example ID	Solution of ABC-TTNR	Cost of ABC-TTNR	Cost of AMMMR
37825	(1,201[10],274[88])	180.7	194.6
148514	(0,327[38],350[40])	163.6	172.6
37826	(1,199[9],266[88])	170.1	180.0
148630	(0,327[42],350[33])	161.4	164.2
168454	(0,387[110])	120.0	129.8
148665	(0,308[37],307[38],340)	154.6	158.2
148743	(0,308[32],307[43])	158.0	177.4
136630	(0,308[32],307[67])	136.0	147.8

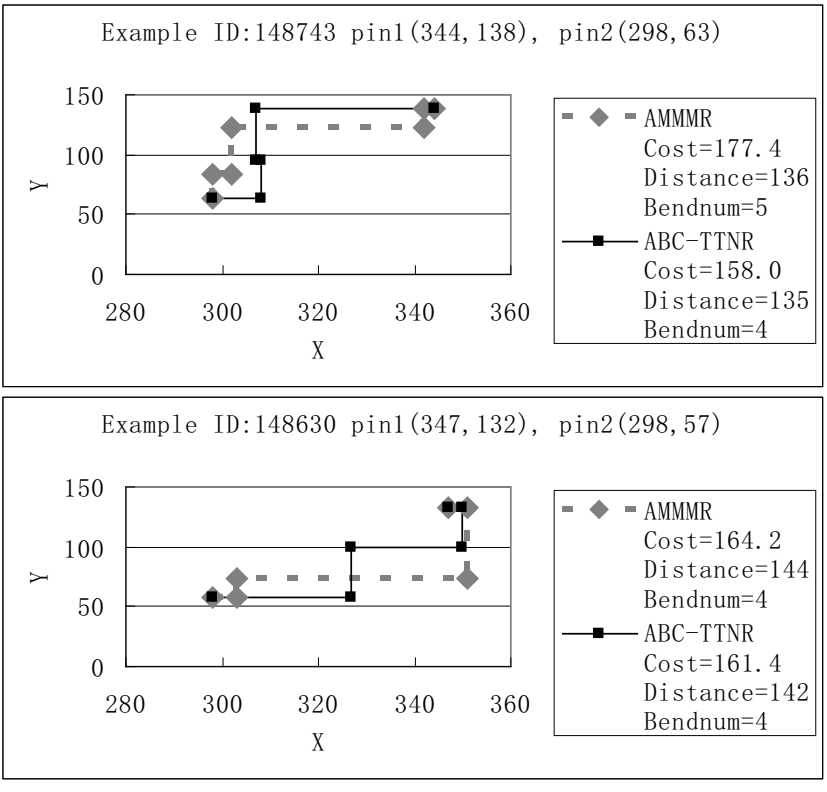


Fig. 2. Simulation Results of Examples

6 Conclusion

In this paper, we have designed and implement an artificial bee colony algorithm ABC-TTNR to solve the TTNR optimization problems. We have compared our approach against a variant of maze search approach AMMMR. Our approach can get less cost paths for the problems than that found by AMMMR. The experiment demonstrates the capability of artificial bee colony algorithm in solving a routing problem in VLSI. Our future research project is to solve the multi-terminal net routing problem in the weighted grid graph with an improved artificial bee colony algorithm.

Acknowledgement. This work was supported by the National Key Basic Research Special Foundation (NKBRSF) of China (No. 2011CB808000), by Provincial Natural Science Foundation of Fujian (No. 2010J01329), by the National Science Foundation of China (No. 61170308), and by Technology Development Foundation of Fuzhou University(2011-XY-17).

References

1. Pan, M., Chu, C.: FastRoute 2.0: A High-quality and Efficient Global Routing. In: 12th Asia and South Pacific Design Automation Conference, pp. 250–255. IEEE Computer Society Press, Washington (2007)
2. Gao, J.-R., Wu, P.-C., Wang, T.-C.: A new global router for modern designs. In: 13th Asia and South Pacific Design Automation Conference, pp. 232–237. IEEE Computer Society Press, Los Alamitos (2008)
3. Hu, J., Roy, J., Markov, I.: Completing High-Quality Global Routes. In: 19th International Symposium on Physical Design, pp. 35–41. ACM Press, New York (2010)
4. Ayob, M.N., Yusof, Z.M., et al.: A Particle Swarm Optimization Approach for Routing in VLSI. In: 2nd International Conference on Computational Intelligence, Communication Systems and Networks, pp. 49–53. IEEE Press, Liverpool (2010)
5. Arora, T., Moses, M.E.: Ant Colony Optimization for power efficient routing in manhattan and non-manhattan VLSI architectures. In: 2009 Swarm Intelligence Symposium, pp. 137–144. IEEE Press, Nashville (2009)
6. Wu, P.-C., Gao, J.-R., Wang, T.-C.: A Fast and Stable Algorithm for Obstacle-Avoiding Rectilinear Steiner Minimal Tree Construction. In: 12th Asia and South Pacific Design Automation Conference, pp. 262–267. IEEE Computer Society Press, Washington (2007)
7. Karaboga, D.: An Idea Based On Honey Bee Swarm For Numerical Optimization, Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department (2005)
8. Karaboga, D., Basturk, B.: On the performance of artificial bee colony (abc) algorithm. Applied Soft Computing 8(1), 687–697 (2008)
9. Chang, Y.-J., Lee, Y.-T., Gao, J.-R., Wu, P.-C., Wang, T.-C.: NTHU-Route 2.0: A Robust Global Router for Modern Designs. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 29(12), 1931–1944 (2010)

A Differentiating Evolutionary Computation Approach for the Multidimensional Knapsack Problem

Meysam Mohagheghi Fard, Yoon-Teck Bau, and Chien-Le Goh

Faculty of Computing and Informatics, Multimedia University,
Persiaran Multimedia, 63100 Cyberjaya, Malaysia
meysam.mfard@gmail.com, {ytbau,clgoh}@mmu.edu.my

Abstract. In this paper, the DEC (Differentiating Evolutionary Computation) algorithm is presented for solving a zero-one multidimensional knapsack problem. It has three new improvements. They are the use of a chromosome bank for elitism, the use of the superior clan and the inferior clan to improve exploitation and exploration, and the use of genetic modification to enable faster convergence. The experimental results have shown that the DEC algorithm is better than a greedy algorithm and a generic genetic algorithm. It can find solutions very close to those found by the algorithm proposed by Chu & Beasley.

Keywords: multidimensional knapsack problem, evolutionary computation, genetic algorithm, DEC algorithm.

1 Introduction and Related Work

Optimization problems are encountered in many real-world situations in many fields. These problems should be solved within a reasonable amount of time and the solutions should be as precise and close to the optimum solution as possible in order to be practical. The Knapsack problem is a well-known combinatorial optimization problem with many real-life applications in areas such as business, finance, capital investments, and budget allocation. The problem of allocating funds to independent R&D projects in Motorola Inc. in 1966 [1] is one of the earliest knapsack problems.

The knapsack problem has different variants. Among them, the zero-one multidimensional knapsack problem (MDKP) is an NP-hard problem which has no known algorithms to solve it efficiently. Nevertheless, optimal solutions to it can greatly assist a decision making process to arrive at a good decision.

In general, the MDKP is a problem to select the most valuable items from a pool of candidate items to be put into a knapsack. The knapsack has multiple dimensions and a distinct capacity associated with each dimension. Each candidate item possesses a value and has a cost associated with each of the knapsack's dimensions. The goal is to choose a certain number of items to put into the knapsack in order to achieve the maximum total value, while respecting the capacity of the knapsack in each dimension.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 342–349, 2012.

If we have an *m*-dimensional knapsack with its *j*-th dimension having a capacity of c_j and there are *n* number of items to choose from, the MDKP can be formulated with equations (1) and (2). x_i is used to indicate whether the *i*-th item has been put into the knapsack. The *i*-th item possesses the value of v_i and a cost or weight of w_{ij} associated with the *j*-th dimension of the knapsack,

$$\text{maximize} \sum_{i=1}^{n} v_i x_i \,, \quad \text{where} \quad x_i = 0 \; or \; 1 \,; \quad i = 1, 2, \ldots, n \,; \tag{1}$$

$$\text{subject to} \sum_{i=1}^{n} w_{ij} x_i \leq c_j \quad \text{where } j = 1, 2, \ldots, m \,. \tag{2}$$

The focus of this research work is to find a better approach to solve the MDKP. Our proposed algorithm is called the DEC (Differentiating Evolutionary Computation) algorithm. It is a genetic algorithm with three new improvements. The improvements are (1) the use of a *superior clan* and an *inferior clan*, (2) the use of a *chromosome bank* and (3) the use of an operation named *genetic modification*.

Chu & Beasley [2] proposed in 1998 a genetic algorithm which used a *repairing operator* to solve the MDKP. The operator can convert infeasible solutions (those that have violated the knapsack capacities) to feasible ones instead of rejecting them outright. It works in two phases, a drop phase and an add phase. In the drop phase, chromosomes are examined in increasing order of a proposed measure called pseudo-utility and their genes are changed from 1 to 0 until feasibility is obtained. Pseudo-utility is a ratio which indicates how suitable an item *i* is to be put into the knapsack. In the add phase, chromosomes are examined in decreasing order of pseudo-utility and their genes are changed from 0 to 1 as long as feasibility is not violated. The operator contributes to the improvement of the quality of the solutions and can introduce very fit new chromosomes to a population. The genetic modification operator in the DEC algorithm is an improvement based on the repair operator.

An algorithm named SPEA2 (the Strength Pareto Evolutionary Algorithm) was introduced by Zitzler, Laumanns, and Thiele in 2001 [3]. Although this algorithm was developed for multi-objective optimizations (e.g. multi-objective knapsack problem), it has some good features that are adaptable to single objective problems. One of them is a feature called *environmental selection* which maintains an archive which contains the best solutions encountered in addition to the main population. In that way, the dominant solutions can survive for many generations. The same feature was also used in NSGA-II [4]. Since preserving the most valuable solutions have a great impact on the quality of the solutions and results in a steady convergence, a similar feature named *chromosome bank* has been adopted for our algorithm.

An improved *roulette wheel selection* strategy was used by Shao, Xu and Yin in 2009 [5] to select individuals to participate in crossover. This strategy not only tries to keep diversity but also reduces the required computation. Under this strategy, the individuals in a population are first sorted with respect to their fitness. Then 2/5 of the best individuals and, by random, 1/3 of the remaining 3/5 of the population are chosen to take part in mating. Crossover is more suitable for individuals of higher fitness as this encourages the propagation of their good characteristics to their children.

Mutation is more suitable for individuals of lower fitness as it can mutate the bad characteristics in them.

In [6], a procedure named *schema replacement* was proposed. It is a procedure used to improve individuals of low fitness faster in a population. In this procedure, the population is divided into two clans, the superior can and the inferior clan. An individual named *excellent schema* is produced based on the superior clan and each individual in the inferior clan are mated with the excellent schema to generate new individuals.

From the concepts presented in [5] and [6], a new concept of dividing the population into two clans and utilizing them in new ways to make crossover and mutation operations more effective has been developed for the DEC algorithm.

2 The DEC Algorithm

The DEC algorithm works with an initial randomly generated population of feasible chromosomes. The population size is 10 times the number of candidate items. A cycle of operations of the DEC algorithm is illustrated in Fig. 1. The improvements introduced in each operation are elaborated from section 2.1 to section 2.4.

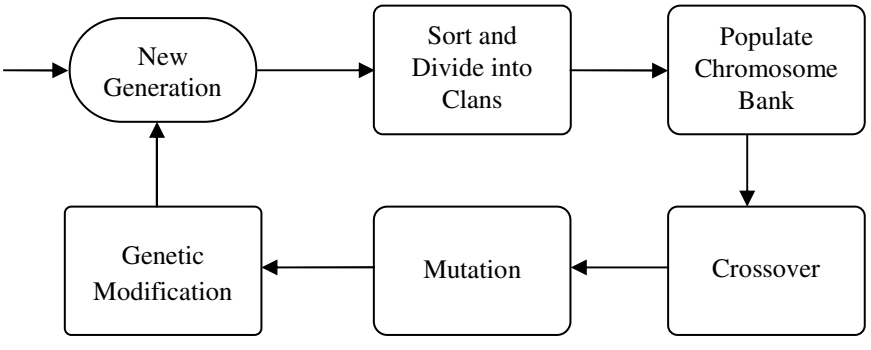


Fig. 1. A cycle of operations of the DEC Algorithm used for MDKP

2.1 Superior Clan and Inferior Clan

Before any operations, the population is sorted based on the individuals' fitness values and divided into two halves named the superior clan and the inferior clan.

Since the individuals in the superior clan have higher fitness values, they are more suitable to be selected as parents for crossover in order to promote exploitation. An individual with a lower fitness value has less number of genes set to 1. Considering that the goal is to maximize the number of 1s in an individual, the inferior clan is more suitable to be selected for mutation. This is because in a mutation operation on an individual from the inferior clan, it is likely that more 0's will be turned into 1's than 1's into 0's. In other words, mutation for the inferior clan promotes more exploration of the search space than mutation on the whole population without any discrimination.

2.2 Chromosome Bank

The DEC algorithm uses the chromosome bank to store a number of the fittest individuals found in different generations since the first generation and use them more often for crossover to promote *elitism*. Part of the chromosome bank also provides candidates for mutation as mutation on individuals with the highest fitness values can help the algorithm to escape from local maxima. The chromosome bank is kept alongside the main population and its size is set as 1/5 of the main population size.

2.3 Crossover and Mutation

Crossover operations are performed on a pool of parents selected using tournament selection from the superior clan and the chromosome bank. The pool size is half the size ($5n$) of the main population ($10n$). 2/5 of the pool's individuals are selected from the chromosome bank and the rest are selected from the superior clan. A uniform crossover with a ratio of 0.5 is used. In this operation, two parents produce two children. The children produced are stored as individuals in a population called *children population*.

The mutation operation (mutation ratio is 0.01) is performed on a pool of individuals where 2/3 of this pool's individuals are selected from *children population* while the remaining 1/3 are selected (using tournament selection) from both the whole inferior clan and 1/5 of the chromosome bank. Similar to the number of children produced from crossover, the total number of individuals produced from mutation is also half the size ($5n$) of the main population ($10n$). Together, they make up $10n$ new individuals in a new generation.

2.4 Genetic Modification

Genetic modification is an operation in the DEC algorithm which makes the infeasible chromosomes feasible and increases the fitness values of feasible chromosomes. It comprises two phases: the *drop* phase and the *add* phase.

In the *drop* phase, an infeasible chromosome is made feasible by dropping, from the knapsack, items that are imposing the greatest costs (weights) to the violated dimensions and at the same time have the least values. To identify a violated dimension, we first sum up the cost of a dimension from every chromosome. If the sum exceeds the capacity of the dimension, the dimension has been violated. Only the violated dimensions are considered for the dropping of items. After the *drop* phase, the *add* phase attempts to add items which have the least costs and the greatest values into the knapsack to make the chromosome fitter.

The main difference between the genetic modification and the repair operator from [2] is that there is no linear programming (LP) relaxation process involved in computing the pseudo-utility ratio. The following pseudo-code presents the genetic modification procedure.

```
// The genetic modification of a chromosome
BEGIN
//The DROP Phase
for every gene i of chromosome {
   effectiveWeight = 0
   for every violated dimension dim {
      effectiveWeight =
          effectiveWeight + (chromosome[i]× w_{i,dim} ÷ c_{dim})
   }
   pseudoUtility_i = effectiveWeight ÷ v_i
}
Set chromosome[i]= 0 in the descending order of
pseudoUtility_i until the chromosome becomes feasible

//The ADD Phase
for every gene i of chromosome {
   effectiveWeight = 0
   for every dimension j of knapsack {
       effectiveWeight = effectiveWeight + (w_{i,j} ÷ c_j)
   }
   pseudoUtility_i = effectiveWeight ÷ v_i
}
Set chromosome[i]= 1 in the ascending order of
pseudoUtility_i while the chromosome is feasible
END
```

3 Experiments and Results

For evaluating the effectiveness of the DEC algorithm, it has been compared with a generic genetic algorithm and a greedy algorithm called *primal effective capacity heuristic* (PECH) [8]. The generic genetic algorithm and the DEC algorithm uses the same parameters for crossover, mutation and tournament selection.

PECH, the generic genetic algorithm and the DEC algorithm have been tested using selected large datasets introduced in [2] which have 100, 250 and 500 items. Three datasets from each 100-item, 250-item, and 500-item datasets with different *tightness ratios* [2] have been chosen. Since the optimal solutions for the datasets are unknown, for each dataset, the gaps between the best solutions found by the three algorithms and the optimal value of LP relaxation presented in [9] have been used to measure the qualities of the solutions. The solution quality is expressed in equation (3). The results of this experiment are shown in Table 1. A problem named, for example, *mknapcb7-100-30-10*, is the 10th problem in *mknapcb7* from [9] with 100 items and 30 dimensions. The generic GA and the DEC algorithm have both been run 10 times on each dataset for 250 generations.

$$Best\ \%\ gap = 100 \times \frac{optimal\ LP\ value - best\ solution\ by\ algorithm}{optimal\ LP\ value} \quad (3)$$

From Table 1, it can be seen that the DEC algorithm is clearly better than PECH and the generic GA in terms of closeness to the optimal solution. In addition, we can see that it converges at a solution faster than the generic genetic algorithm if we compare the average best generation numbers between them.

Table 1. Computational results for Greedy, Generic Genetic Algorithm (250 generations) and the DEC Algorithm (250 generations). The smaller the Best % Gap the better the result.

Problem			Greedy Algorithm	Generic Genetic Algorithm		The DEC Algorithm	
Problem Set Name	Items #	Dim.	Best % Gap	Best % Gap	Average Best Generation Number	Best % Gap	Average Best Generation Number
mknapcb7-100-30-00	100	30	10.93	3.86	208	**2.80**	**81**
mknapcb7-100-30-10	100	30	5.70	2.67	200	**1.69**	**151**
mknapcb7-100-30-29	100	30	4.11	1.12	226	**0.93**	**149**
mknapcb8-250-30-00	250	30	7.64	7.52	191	**1.39**	**138**
mknapcb8-250-30-10	250	30	4.31	4.09	209	**0.66**	**157**
mknapcb8-250-30-29	250	30	1.59	2.17	221	**0.41**	**154**
mknapcb9-500-30-00	500	30	7.02	9.50	185	**0.96**	187
mknapcb9-500-30-10	500	30	2.12	5.79	228	**0.36**	**161**
mknapcb9-500-30-29	500	30	1.80	3.07	214	**0.23**	**210**

Table 2 shows the best solutions found by the DEC algorithm after 1,000 generations and the best solutions found by the algorithm proposed by Chu and Beasley [2] (written as C&B GA for simplicity). In terms of quality, the DEC algorithm has achieved very close results when compared to those found by C&B GA, despite using heuristics instead of the superior but more complex method based on mathematical programming foundation such as LP relaxations used by C&B GA.

In order to probe the effectiveness of the three new improvements introduced by the DEC algorithm, three experiments have been conducted where one of the operations is disabled in each experiment. The dataset used is mknapcb9-500-30-00 which has 500 items and a tightness ratio of 0.25. Fig. 2 shows the experiment results where genetic modification has been disabled. Results similar to Fig. 2 have been obtained when the superior clan and the inferior clan or the chromosome bank is disabled. All the results have shown that the values of the solutions decrease when any one of the improvements is disabled. Therefore, it can be concluded that together, they can integrate well to achieve good results.

Table 2. Computational results for the DEC algorithm after 1000 generations when compared with the results of Chu & Beasley. The higher the value the better the solution.

Problem			The DEC Algorithm	C&B GA
Problem Set Name	**Items #**	**Dim.**	**Value of the Best Solution**	**Value of the Best Solution**
mknapcb7-100-30-00	100	30	**21946**	**21946**
mknapcb7-100-30-10	100	30	40637	40767
mknapcb7-100-30-29	100	30	60574	60603
mknapcb8-250-30-00	250	30	**56693**	**56693**
mknapcb8-250-30-10	250	30	107638	107689
mknapcb8-250-30-29	250	30	149536	149572
mknapcb9-500-30-00	500	30	115771	115868
mknapcb9-500-30-10	500	30	217857	217995
mknapcb9-500-30-29	500	30	300341	300460

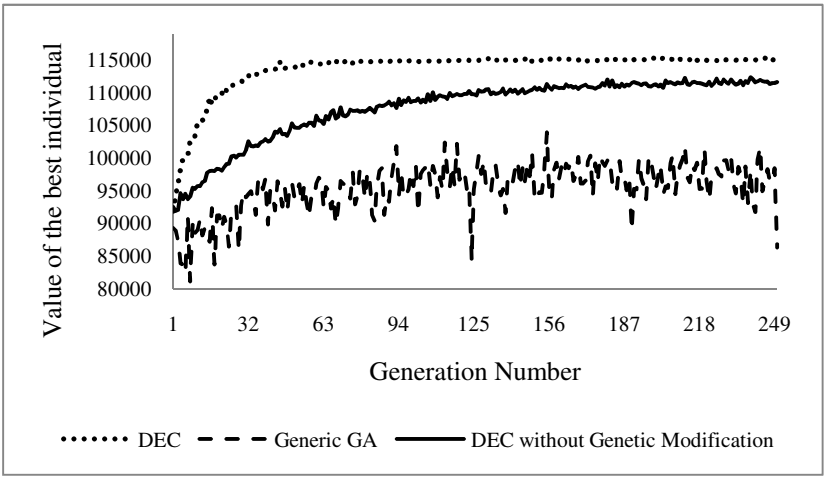


Fig. 2. Value of the best individual if genetic modification is disabled

4 Conclusion

The proposed DEC algorithm for solving a zero-one multidimensional knapsack problem has three new improvements. They are the use of a chromosome bank for elitism, the use of the superior clan and the inferior clan to improve exploitation and exploration, and the use of genetic modification to enable faster convergence.

The validity of the DEC algorithm has been verified through experiments based on established datasets. The results have shown that the DEC algorithm is better than a greedy algorithm such as PECH and the generic genetic algorithm in terms of the closeness to the optimal solution, and can search for a solution more effectively than the generic genetic algorithm. The DEC algorithm has also achieved very close results when compared with those from C&B GA, despite using heuristics instead of the superior but more complex method based on mathematical programming foundation such as LP relaxations.

The heuristics used in the DEC algorithm is still not a perfect mechanism to prevent the search from being trapped in local maxima. Therefore, to improve the search for the maximum value, more effort is needed to find ways to explore the search space more extensively with even better heuristics.

References

1. Petersen, C.C.: Computational Experience with Variants of the Balas Algorithm Applied to the Selection of R&D Projects. Management Science, 736 (1967)
2. Chu, P., Beasley, J.: A Genetic Algorithm for the Multidimensional Knapsack Problem. Journal of Heuristics 4(1), 63–86 (1998)
3. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, Swiss Federal Institute of Technology (ETH) Zurich, Computer Engineering and Networks Laboratory (TIK), Department of Electrical Engineering (2001)
4. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2), 182–197 (2002)
5. Shao, Y., Xu, H., Yin, W.: Solve Zero-One Knapsack Problem by Greedy Genetic Algorithm. In: IEEE 2009 International Workshop on Intelligent Systems and Applications (ISA), pp. 1–4 (2009)
6. Lin, C.: A Heuristic Genetic Algorithm Based on Schema Replacement for 0-1 Knapsack Problem. In: 2010 Fourth International Conference on Genetic and Evolutionary Computing (ICGEC), Shenzhen, China, pp. 301–304 (2010)
7. Fréville, A.: The multidimensional 0–1 knapsack problem: An overview. European Journal of Operational Research 155(1), 1–21 (2004)
8. Akçay, Y., Li, H., Xu, S.: Greedy algorithm for the general multidimensional knapsack problem. Annals of Operations Research 150(1), 17–29 (2007)
9. Beasley, J.: OR-Library, Multidimensional knapsack problem, `http://people.brunel.ac.uk/~mastjjb/jeb/orlib/mknapinfo.html`

Ensemble of Clearing Differential Evolution for Multi-modal Optimization

Boyang Qu[1,3], Jing Liang[2], Ponnuthurai Nagaratnam Suganthan[3], and Tiejun Chen[2]

[1] School of Electric and Information Engineering,
Zhongyuan University of Technology, China 450007,
E070088@e.ntu.edu.sg
[2] School of Electrical Engineering, Zhengzhou University, China, 450001
{Liangjing,tchen}@zzu.edu.cn
[3] School of Electrical and Electronic Engineering,
Nanyang Technological University, Singapore, 639798
{E070088,epnsugan}@e.ntu.edu.sg

Abstract. Multi-modal Optimization refers to finding multiple global and local optima of a function in one single run, so that the user can have a better knowledge about different optimal solutions. Multiple global/local peaks generate extra difficulties for the optimization algorithms. Many niching techniques have been developed in literature to tackle multi-modal optimization problems. Clearing is one of the simplest and most effective methods in solving multi-modal optimization problems. In this work, an Ensemble of Clearing Differential Evolution (ECLDE) algorithm is proposed to handle multi-modal problems. In this algorithm, the population is evenly divided into 3 subpopulations and each of the subpopulations is assigned a set of niching parameters (clearing radius). The algorithms is tested on 12 benchmark multi-modal optimization problems and compared with the Clearing Differential Evolution (CLDE) with single clearing radius as well as a number of commonly used niching algorithms. As shown in the experimental results, the proposed algorithm is able to generate satisfactory performance over the benchmark functions.

Keywords: Differential evolution, evolutionary computation, multi-modal optimization, niching.

1 Introduction

In recent decades, evolutionary algorithms (EAs) have been proven to be effective in solving difficult practical optimization problems. In practical optimization problems, it is often desirable to simultaneously locate multiple global and local peaks of a given objective function, such as classification problems in machine learning 1 and inversion of teleseismic waves 2. These problems are known as multi-modal optimization problems. The original forms of most EAs are designed for locating single global solution, which is not effective for multi-modal optimization. To overcome this problem, various techniques that commonly known as "niching" methods are proposed and incorporated in EAs to enhance the algorithm with the ability of maintaining multiple stable subpopulation which target on locating different peaks.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 350–357, 2012.

Niching is a generic term referred to as the technique of finding and preserving multiple stable niches, or favorable parts of the solution space possibly around multiple solutions, so as to prevent convergence to a single solution 3. A niching method generally modifies the behavior of a classical EA in order to maintain multiple groups within a single population in order to locate multiple optima. The earliest niching approach was proposed by Cavicchio 4. Some other representative methods are crowding 5-6, restricted tournament selection 7, clearing 8, fitness sharing 9 and speciation 10.

Differential evolution is a powerful global optimization technique. Niching techniques have also been incorporated into DE variants to enhance the ability of handling multimodal optimization 11-14. In this paper, an Ensemble of Clearing Differential Evolution (ECLDE) algorithm is proposed to overcome the problem of selecting suitable niching parameters.

The remainder of this paper is organized as follows. Section 2 gives a brief interlocution of differential evolution, and Clearing Differential Evolution (CLDE). In Section 3, the proposed ECLDE is introduced. The problem definition and results of the experiments are presented in sections 4. Finally, the paper is concluded in sections 6.

2 Differential Evolution and Clearing

2.1 Differential Evolution

The differential evolution (DE) algorithm was first proposed by Storn and Price 15. Although the idea of DE is simple, it is very effective in solving global optimization problem. Similar to other EAs, DE is also a population based searching algorithm. The individuals will compete with others inside the population. Four major steps are involved in DE known as, initialization, mutation, recombination and selection 16.

2.2 Clearing and Clearing DE

Clearing 8 is one of the most widely used niching methods. It removes bad solutions/individuals and keeps only the fitness individual in each niche. In this

Table 1. ECLDE algorithm

Step 1	Use standard DE to produce NP (population size) offspring.
Step 2	Combine parents with the newly generated offspring
Step 3	Sort the combined population in descending order.
Step 4	Apply clearing method on the combined population using the user predefined clearing radius Rs.

Step 5	If the remained population size > NP
	Remove the rest individuals and left with only top NP individuals.
	Else
	Contiue
	Endif

Step 6	Stop if the termination criteria are met otherwise go to step 1.

technique, the population is first sorted in descending order according to the objective value. Then it picks one individual at a time from the top and eliminate all the rest individuals that falling within the specified clearing radius. The process will be repeated until all individuals are processed. Clearing is able to maintain the diversity of the population by removing similar individuals. The niching parameter (clearing radius) is used as a dissimilarity threshold and the performance of clearing algorithm is highly depended on this user defined parameter. The complexity of clearing can be calculated as $O(cN)$, where c is the number of niches maintained during each generations and N is the number of individuals in the population.

Clearing DE (CLDE) incorporates DE with the clearing technique for handling multi-modal optimization problems. The main steps of CLDE are shown in Table 1.

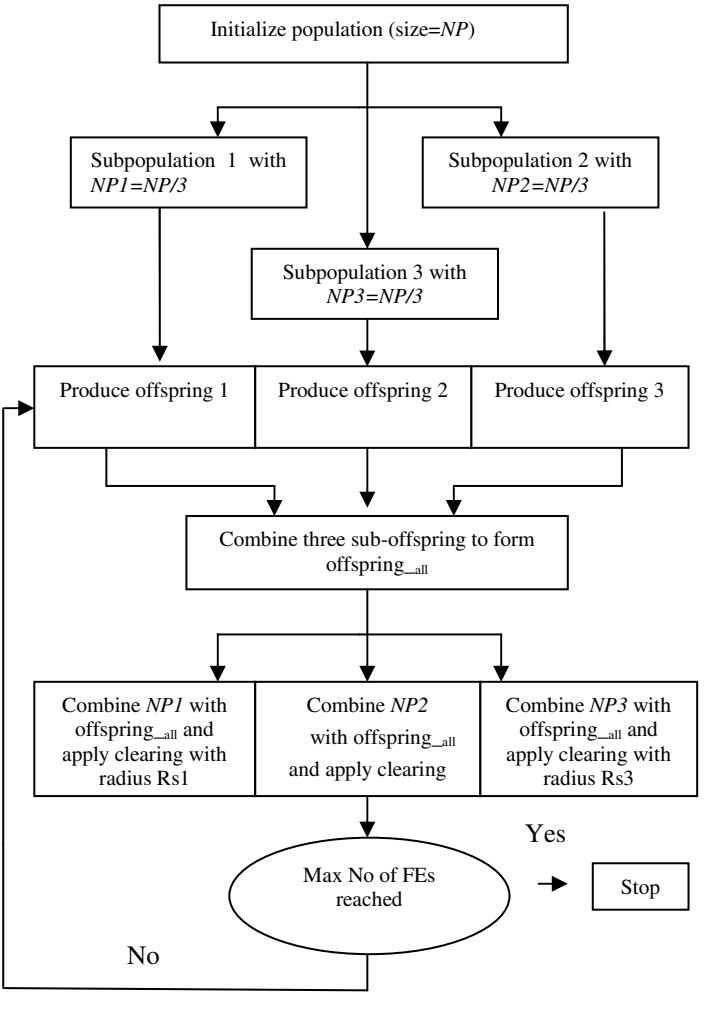


Fig. 1. The flowchart of ECLDE

3 Ensemble of CLDE

Although clearing is an effective niching method in solving multimodal optimization problems, selection of a suitable clearing radius generally is difficult if no prior information is available about the problem. The performance of CLDE is closely related to the parameter Rs (clearing radius). According to "No free lunch" theorem 17, it is unlikely to find one parameter that can outperform all other parameters, since different parameter is suitable for different problems. In recent years, ensemble ideas are commonly used to handle parameter/method selection problem 18-20. Motivated by these observations, an Ensemble of Clearing Differential Evolution (ECLDE) algorithm is proposed.

In ECLDE, the algorithm divides the initial population into three equal subpopulations. Each of the subpopulation is assigned a different clearing radius. In this way, the algorithm makes use of three different parameters and exchange information among different subpopulation during the selection process. The flowchart of ECLDE is shown in Fig. 1. Note that the three clearing radiuses are chosen as Rs_1 =0.005*the search range of the problem; Rs_2 =0.01*the search range of the problem; Rs_3 =0.05*the search range of the problem.

4 Experiment Preparation

4.1 Test Functions and Compared Algorithms

To assess the performance of the proposed algorithm, twelve commonly used multimodal optimization benchmark test functions with different characteristics are used. The details of these test functions are shown in Table 2.

Table 2. Test Functions

Test Function	Peaks Global/local	Test Function	Peaks Global/local
F1:Central Two-Peak Trap	1/1	F7: 1D Inverted Vincent function 21	6/0
F2:Five-Uneven-Peak Trap [9]	2/3	F8: 2D Inverted Vincent function 21	36/0
F3: Waves	1/9	F9: CF1 22	8/0
F4:2D Inverted Shubert function [9]	18/many	F10: CF2 22	6/0
F5:3D Inverted Shubert function [9]	81/many	F11: CF3 22	6/0
F6:4D Inverted Shubert function [9]	324/many	F12: CF4 22	6/0

4.2 Experimental Setup

In this experiment, Matlab 7.1 is used as the programming language. The configurations of the computer are Intel Pentium® 4 CPU, 4 Gb of memory. The population size,

maximum number of FEs and level of accuracy are listed in Table 3. 25 independent runs are conducted for each of the algorithms.

4.3 Performance Measures

A level of accuracy need to be specified in order to assess the performance of different algorithms. An optimum is considered to be found if there exists a solution in the population within the tolerated distance to that optimum. When doing the comparison, following to criteria are used:

1. Success Rate
2. Average number of optima found.

Table 3. Level of accuracy used in this experiment

Test Function	Population Size	Max No. of FEs	Level of accuracy	Test Function	Population Size	Max No. of FEs	Level of accuracy
F1	60	12000	0.0005	F7	60	12000	0.001
F2	150	30000	0.005	F8	600	200000	0.001
F3	240	36000	0.001	F9	600	300000	0.5
F4	180	72000	0.05	F10	600	300000	0.5
F5	600	200000	0.2	F11	600	300000	0.5
F6	900	400000	0.2	F12	600	300000	0.5

5 Experiment Results

5.1 Success Rate

ECLDE is first compared with the three original CLDE with single clearing radius and the success rate is presented in Table 4. Note that the success rates for function F9-F12 are all zero and they are not presented in Table 4. The rank of each algorithm is presented in the parentheses while total ranks (summation of all the individual ranks) are listed in the last row of the table. As can be seen from the table, the

Table 4. The success rate

Test Function	CLDE1	CLDE2	CLDE3	ECLDE
F1	0 (3)	0 (3)	0.04 (2)	**0.16** (1)
F2	0 (2)	0 (2)	0 (2)	**0.16** (1)
F3	0 (3)	0 (3)	**0.04 (1)**	**0.04 (1)**
F4	0 (3)	**1 (1)**	0 (3)	**1 (1)**
F5	0 (2)	0 (2)	0 (2)	**1 (1)**
F6	**0 (1)**	**0 (1)**	**0 (1)**	**0 (1)**
F7	**1 (1)**	0.84 (3)	0.76 (4)	**1 (1)**
F8	**0.8 (3)**	1 (1)	0 (4)	**1 (1)**
Total Rank	18	16	19	**8**

ECLDE always performs the best among all the tested algorithms. The superior performance is due to the exchanging information among different subpopulations which use different niching parameter. Note that the success rate obtained is highly related to the user defined parameter level of accuracy (demonstrated in following section).

5.2 Number of Optima Found

Besides success rate, number of optima found is another important criterion to access multi-modal optimization algorithms. The number of optima found by the four compared algorithms for each of the test functions are shown in Table 5. The mean value is highlight in bold face. In order to determine the statistical significance of the advantage of ECLDE, t-test is applied. The results are presented in the last row of each test functions. The numerical values 1, 0 represent that other methods are statistically inferior to, equal to ECLDE. From the t-test summary, we can see ECLDE performs either better or similar to other algorithms.

Table 5. The number of optima found

Test Function		CLDE1	CLDE2	CLDE3	ECLDE
F1	Mean	**1.00**	**1.00**	**1.04**	**1.16**
	Std	0.00	0.00	0.20	0.36
	t-test	1	1	0	-
F2	Mean	**2.84**	**2.60**	**2.68**	**3.16**
	Std	0.62	0.71	0.56	1.11
	t-test	0	1	1	-
F3	Mean	**4.60**	**4.72**	**6.36**	**6.96**
	Std	0.76	1.06	1.15	1.24
	t-test	1	1	0	-
F4	Mean	**13.16**	**18.00**	**9.12**	**18.00**
	Std	2.37	0.00	0.33	0.00
	t-test	1	0	1	-
F5	Mean	**52.10**	**51.08**	**19.93**	**81.00**
	Std	4.89	6.28	4.80	0.00
	t-test	1	1	1	-
F6	Mean	**226.40**	**195.70**	**0.00**	**301.40**
	Std	26.31	39.07	0.00	3.37
	t-test	1	1	1	-
F7	Mean	**6.00**	**5.84**	**5.76**	**6.00**
	Std	0.00	0.37	0.44	0.00
	t-test	0	1	1	-
F8	Mean	**35.80**	**36.00**	**20.40**	**36.00**
	Std	0.42	0.00	1.90	0.00
	t-test	1	0	1	-
F9	Mean	**1.00**	**0.00**	**0.00**	**1.50**
	Std	0.00	0.00	0.00	0.53
	t-test	1	1	1	-
F10	Mean	**1.00**	**1.00**	**0.00**	**1.00**
	Std	0.00	0.00	0.00	0.00
	t-test	0	0	1	-
F11	Mean	**1.00**	**1.00**	**0.00**	**1.20**
	Std	0.00	0.00	0.00	0.42
	t-test	1	1	1	-
F12	Mean	**1.00**	**0.10**	**0.00**	**1.00**
	Std	0.00	0.32	0.00	0.00
	t-test	0	1	1	-
t-test summary	Worse	0	0	0	**-**
	Similar	4	3	2	-
	Better	8	9	10	-

5.3 Comparison with Other Algorithms

In this section, ECLDE is compared with some popular multimodal optimization algorithms proposed in literature. The results (average number of optima found) are presented in Table 6. As can been seen from the results, ECLDE performs the best over the compared algorithms (total rank).

Table 6. Comparison with other algorithms (average number of optima found)

Test Function	CDE 13	R2PSO 21	FERPSO 21	SPSO 21	R2PSOLHC 21	SDE 23	ECLDE
F1	1.08 (2)	0.08 (7)	0.20 (6)	0.80 (5)	1.00 (4)	1.36 (1)	1.08 (2)
F2	2.52 (4)	0.80 (7)	1.64 (6)	2.08 (5)	3.08 (3)	4.00 (1)	3.16 (2)
F3	5.72 (2)	3.80 (4)	1.08 (7)	2.52 (6)	4.32 (3)	2.68 (5)	6.96 (1)
F4	17.80 (2)	12.60 (5)	15.84 (3)	9.64 (7)	13.60 (4)	10.80 (6)	18.00 (1)
F5	53.76 (2)	0.80 (6)	21.60 (3)	1.40 (5)	0.64 (7)	9.76 (4)	81.00 (1)
F6	1.12 (4)	0.00 (5)	7.40 (2)	0.00 (5)	0.00 (5)	3.72 (3)	301.40 (1)
F7	5.60 (5)	5.64 (4)	5.60 (5)	6.00 (1)	5.68 (3)	5.20 (7)	6.00 (1)
F8	33.84 (2)	21.76 (7)	23.60 (4)	25.68 (3)	23.08 (5)	22.84 (6)	36.00 (1)
F9	0.00 (4)	0.00 (4)	1.08 (3)	0.00 (4)	0.00 (4)	1.79 (1)	1.50 (2)
F10	1.20 (2)	0.00 (5)	2.00 (1)	0.00 (5)	0.00 (5)	1.20 (2)	1.00 (4)
F11	0.70 (4)	0.00 (5)	2.50 (1)	0.00 (5)	0.00 (5)	1.50 (2)	1.20 (3)
F12	0.00 (2)	0.00 (2)	0.00 (2)	0.00 (2)	0.00 (2)	0.00 (2)	1.00 (1)
Total Rank	35	63	43	53	50	40	**20**

6 Conclusion

In this paper, an ensemble of clearing differential evolution (ECLDE) algorithm for multi-modal optimization is introduced to overcome the difficulty of choosing niching parameter. The proposed algorithm is compared with the original clearing DE as well as some other commonly used multi-modal optimization algorithms. The experiments show that ECLDE is able to generate satisfactory performance over the tested benchmark functions.

Acknowledgments . This research is partially supported by National Natural Science Foundation of China (Grant NO. 60905039, 71001072) and Postdoctoral Science Foundation of China (Grants 20100480859).

References

1. Mahfoud, S.W.: Niching methods for genetic algorithms. Ph.D. dissertation, Urbana, IL, USA (1995), `http://citeseer.ist.psu.edu/mahfoud95niching.html`
2. Koper, K., Wysession, M.: Multimodal function optimization with a niching genetic algorithm: A seis-mological example. Bulletin of the Seismological Society of America 89, 978–988 (1999)

3. 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)
4. Cavicchio, D.J.: Adaptive search using simulated evolution, Ph.D. dissertation, University of Michigan, Ann Arbor (1970)
5. De Jong, K.A.: An analysis of the behavior of a class of genetic adaptive systems, Ph.D. dissertation, University of Michigan (1975)
6. Mahfoud, S.W.: Crowding and preselection revisited. In: Manner, R., Manderick, B. (eds.) Parallel Problem Solving From Nature 2, pp. 27–36
7. Harik, G.R.: Finding multimodal solutions using restricted tournament selection. In: Proceedings of the Sixth International Conference on Genetic Algorithms. Morgan Kaufmann
8. Pétrowski, A.: A clearing procedure as a niching method for genetic algorithms. In: Proc. of the IEEE Int. Conf. on Evolutionary Computation, New York, USA, pp. 798–803 (1996)
9. Goldberg, D.E., Richardson, J.: Genetic algorithms with sharing for multimodal function optimization. In: Grefenstette, J. (ed.) Proceedings of the Second International Conference on Genetic Algorithms, pp. 41–49 (1987)
10. Li, J.P., Balazs, M.E., Parks, G.T., Clarkson, P.J.: A species conserving genetic algorithm for multimodal function optimization. Evol. Comput. 10(3), 207–234 (2002)
11. Zaharie, D.: Extensions of differential evolution algorithms for multimodal optimization. In: Proceedings of SYNASC 2004, 6th International Symposium of Symbolic and Numeric Algorithms for Scientific Computing, pp. 523–534 (2004)
12. Hendershot, Z.: A differential evolution algorithm for automatically discovering multiple global optima in multidimensional, discontinues spaces. In: Proceedings of MAICS 2004, Fifteenth Midwest Artificial Intelligence and Cognitive Sciences Conference, pp. 92–97 (2004)
13. Thomsen, R.: Multi-modal optimization using crowding-based differential evolution. In: Proceedings of the 2004 Congress on Evolutionary Computation, vol. 2, pp. 1382–1389 (2004)
14. Qu, B.Y., Suganthan, P.N.: Modified species-based differential evolution with self – adaptive radius for multi-modal optimization. In: International Conference on Computational Problem Solving (ICCP), China, pp. 326–331 (2010)
15. Storn, R., Price, K.V.: Differential evolution-Asimple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11, 341–359 (1995)
16. Price, K.: An introduction to differential evolution. New Ideas in Optimization, 79–108 (1999)
17. Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation, 67–82 (1997)
18. Yu, E.L., Suganthan, P.N.: Ensemble of niching algorithms. Information Sciences 180(15), 2815–2833 (2010)
19. Qu, B.Y., Suganthan, P.N.: Constrained Multi-Objective Optimization Algorithm with Ensemble of Constraint Handling Methods. Engineering Optimization 43(4), 403 (2011)
20. Qu, B.Y., Gouthanan, P., Suganthan, P.N.: Dynamic Grouping Crowding Differential Evolution with Ensemble of Parameters for Multi-modal Optimization. In: Panigrahi, B.K., Das, S., Suganthan, P.N., Dash, S.S. (eds.) SEMCCO 2010. LNCS, vol. 6466, pp. 19–28. Springer, Heidelberg (2010)
21. Li, X.: Niching without niching parameters: particle swarm optimization using a ring topology. IEEE Transactions on Evolutionary Computation 14 (February 2010)
22. 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, pp. 3480–3486 (July 2010)
23. 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)

Memetic Differential Evolution for Vehicle Routing Problem with Time Windows

Wanfeng Liu, Xu Wang, and Xia Li*

Department of Electrical Engineering College of Information Engineering,
Shenzhen University Shenzhen, China
{liuwf,wangxu_626}@163.com, lixia@szu.edu.cn

Abstract. In this paper, an improved memetic differential evolution algorithm with generalized fitness (MDEGF) is proposed for vehicle routing problem with time windows (VRPTW). A generalized fitness strategy is designed to evaluate the quality of source-individuals, which incorporates three simple local search techniques and helps to improve the convergent performance. Experimental results show that the novel algorithm can solve the VRPTW and obtain better solution in short time.

Keywords: Vehicle Routing Problem, Optimization, Differential Evolution, Local Search.

1 Introduction

The vehicle routing problem (VRP) was proposed by Dantzig and Ramser[1] in 1959. As a well-known combinatorial optimization problem , the VRP has several variations. The most extensively studied is the vehicle routing problem with time windows (VRPTW). Actually, many real-life problems can be modeled as the VRPTW, such as the post office delivery, the route scheduling of trains and so on.

Many heuristic algorithms have been proposed to solve the VRPTW. Solomon [2] described several heuristics for the VRPTW and introduced 56 benchmark problems. Those heuristics were widely studied and improved by later scholars [3, 4]. In [3], a hybrid genetic algorithm was proposed with adaptive diversity management for a large class of VRPTW. The authors introduced new move evaluation techniques and developed geometric and structural problem decompositions to address efficiently large scale problems. In [5], a new genetic algorithm based hybrid algorithm was presented, which incorporates with the greedy randomized adaptive search procedure, the expanding neighborhood search strategy and particle swarm optimization.

Differential Evolution (DE) falls into the evolutionary algorithms family and is regarded as a stochastic global optimizers. It employs a real-value encoding scheme and makes use of the differentiation information among individuals to find the global optimum in the continuous search space. It has been applied with remarkable success

* Corresponding author.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 358–365, 2012.

on large quantities of numerical optimization problems outperforming other more popular meta-heuristics such as the genetic algorithms for its fast, robust, and efficient global search heuristics of current interest. Nevertheless, DE may occasionally suffer from the problem of stagnation and premature convergence. The combination of local search with DE which comes from the idea of memetic algorithms (MAs) is an appropriate strategy to improve the convergence performance. MAs are hybrid algorithms which combine a population-based global algorithm with local search[6]. Paper [7] presents a memetic algorithm based on DE to improve the performance of evolutionary algorithms for job shop scheduling.

In this paper, an improved memetic differential evolution algorithm with generalized fitness is presented for VRPTW. We define the source-space in VRPTW and design a generalized fitness strategy to evaluate the quality of source-individuals. The source-individuals are mapped to solution-individuals of VRPTW. The generalized fitness strategy is presented to convert the continuous values of individuals from DE to discrete ones in VRPTW and to reserve the potential solutions.

The remainder of this paper is organized as follows: Section 2 briefly describes the mathematical model for VRPTW. Section 3 presents the proposed memetic differential evolution with generalized fitness. The experimental results and analysis are reported in Section 4 follow by the conclusion in Section 5.

2 Mathematical Model for VRPTW

A typical VRPTW consists of a central depot with a fleet of M identical vehicles with capacity limit Q and K customers locating at different sites. Each customer $i(i=1,2,\ldots,K)$ has a varied demand d_i, a service time T_i and a pre-defined time window $[e_i, l_i]$, where e_i is the earliest arrival time and l_i the latest arrival time. Vehicles must service each customer within its time window. If the arriving time of a vehicle is earlier than e_i, a waiting time will be incurred. It is obvious that the arriving time can never be later than l_i. In addition, each route must start from and end at the central depot within the depot time window $[e_0, l_0]$. Each customer must be serviced once by one vehicle. The VRPTW aims to service all the customers using the minimum costs so that the following constraints are satisfied: 1) The time window constraints is observed; 2) The capacity limit is satisfied and 3) Each route satisfies the depot time window constraint. The mathematical formulation for the VRPTW is as follows and a detailed description of the VRPTW model can be found in[3].

$$min \sum_i \sum_j \sum_m c_{ijm} x_{ijm} \tag{1}$$

$$subject\ to \sum_i d_i y_{im} \le Q \quad m = 1, \ldots, M \tag{2}$$

$$\sum_m y_{im} = \begin{cases} M & i = 0 \\ 1 & i = 1, \ldots, K \end{cases} \tag{3}$$

$$\sum_i x_{ijm} = y_{jm} \qquad j = 0, \ldots, K \quad m = 1, \ldots, M \tag{4}$$

$$\sum_j x_{ijm} = y_{im} \qquad i = 0, \ldots, K \quad m = 1, \ldots, M \tag{5}$$

$$e_i \le b_i \le l_i \qquad i = 0, \ldots K \tag{6}$$

where c_{ijm} states the transportation cost of vehicle *m* from customer *i* to customer *j* and b_i is the time of vehicle arrived at customer *i*. If customer *i* is serviced by vehicle *m*, then y_{im} equals 1, otherwise it equals 0. If edge (*i*, *j*) is used by vehicle *m*, x_{ijm} is set to 1, or 0 otherwise.

The objective function in eqn (1) is devoted to minimize the total travel distance if the number of vehicles is determined. Eqn (2) satisfies the vehicle capacity constraint. Eqn (3) guarantees that each vehicle starts from and ends at the central depot, and each customer is serviced only once. Eqns (4) and (5) are flow constraints requiring that each customer must be assigned to the appointed vehicle exactly. Eqn (6) gives the time window constraints for each customer and the depot.

3 Differential Evolution

DE was introduced by Storn and Price in 1995[8]. It is one of the novel evolutionary algorithms for continuous optimization problems with simple theoretical framework and easy computation. Basically, differential evolution is a population-based evolution algorithm which starts with the random initialization of individuals and works on the cooperative and competitive behaviors of the individuals in the population. According to the difference of the individuals, DE searches the global optimum by employing the distance and direction information. Similar to the popular GA, it has three basic operations: selection, mutation and crossover.

Let $S_i^t = [s_{i1}, \ldots, s_{iK}](i = 1,2,..,NP)$ denotes the *i*th individual in the *K*-dimensional search space at generation *t*. The DE basic mutation scheme, which is denoted as DE/rand/1/bin [7], can be described as:

$$V_i^{t+1} = S_{r1}^t + F \cdot (S_{r2}^t - S_{r3}^t) \tag{7}$$

Where *NP* is the size of the population. $S_{r1}^t, S_{r2}^t, S_{r3}^t$ are randomly chosen from the current population which are mutually different and also different from the current individual S_i^t. $F \in (0,2)$ is the scaling factor and S_{r1}^t is the base vector to be perturbed.

The crossover operator is applied to each target individual after the mutation phase. Thus, a trial vector $U_i^{t+1} = [u_{i1}^{t+1}, \ldots, u_{iK}^{t+1}]$ is generated by the following equation:

$$u_{ij}^{t+1} = \begin{cases} v_{ij}^{t+1}, & if \quad (rand(j) \le CR) or \; j = randn(i) \\ s_{ij}^t, & otherwise \end{cases} \qquad j = 1, \ldots, K \tag{8}$$

Where *rand*(*j*) is the *j*th evaluation of a random number uniformly distributed in the range of [0,1], $CR \in [0, 1]$ is the crossover parameter which controls the diversity of the population, *randn*(*i*) is a randomly chosen index from the set {1,2,…,*K*}.

The selection operation simply chooses the better individual between the trial vector U_i^{t+1} and the current target individual S_i^t.

$$S_i^{t+1} = argminf(Y) \quad Y \in \{U_i^{t+1}, S_i^t\} \tag{9}$$

f is the fitness function and S_i^{t+1} is the individual of the new population.

4 Memetic Differential Evolution Algorithm

The basic DE algorithm cannot be directly used for VRPTW since it is based on the real valued operators. Though there are modified differential evolutions proposed for combinatorial optimization problems in [9, 10], for the sake of simplicity, this paper prefers to use the real-valued DE. Two spaces are defined, the real-valued source-space suitable for DE, and the feasible integer-valued solution-space for VRPTW.

4.1 Source-Space and Solution-Space

Definition 1. For the VRPTW with *K* customers, $S = \{s_1, \dots, s_i, \dots, s_K\}\ s_i \in [0,1]$ is defined as the source-individual of VRPTW, and the set of all source-individuals is called source-space.

Definition 2. For the VRPTW with *K* customers,

$$X = \{x_1, \dots x_i, \dots, x_N\} \begin{cases} x_i = 0 & if\ i = 1\ or\ i = N \\ x_i = 0,1, \dots, K & if\ i = 2,3, \dots, N-1 \end{cases} \tag{10}$$

is defined as the solution-individual, and the set of all solution-individuals constructs the solution space.

It is worth noting that in eqn(10) 0 stands for the central depot and may be repeated in the individual *X*, while the value of x_i denotes the sequence number of customers, each of which appears only once in a specific solution-individual *X*. The coding length of solution-individual *N* associated with the number of vehicles varies from time to time in the evolution of the optimization. For instance, 0 can be inserted in an appropriate position when a sub-path is overload or timeout to produce a feasible solution, or successive 0s can be replaced by a single 0.

4.2 Generalized Fitness Strategy

In this paper, differential evolution is iterated in real-valued source-space. However, VRPTW is a discrete combinatorial optimization problem. The key problem of using DE algorithm for VRPTW is how to evaluate the quality of source-individual. Therefore, the generalized fitness of source-individual is defined. In the generalized fitness strategy, for arbitrary source-individual S_i, there is a corresponding feasible

solution-individual X_i in solution space. The fitness of X_i is defined as the generalized fitness of S_i, denoted by $GF(S_i)$. To calculate the generalized fitness, we take the following steps:

For an arbitrary source-individual $S_i \in [0,1]^K$, sort the elements in ascending order. The ascending-order sequence numbers of S_i construct the discrete source-individual S_d. Then, convert S_d to solution-individual X_i by inserting appropriate zeros into S_d. The insertion of the zeros are implemented in such a way that the sub-path between two zeros is feasible (satisfies the capacity and time demands) with as many customers as possible. This feasible solution-individual X_i is optimized by three local search algorithms including Or-opt, 2-opt* and λ-interchange in exhaustive manner, and the resulting new local-optimum solution-individual Y_{local} is obtained. The fitness for Y_{local}, defined as eqn(10) is the resulting generalized fitness for S_i. Since the solution-individual has varied length, the local search algorithms employed may simultaneously minimize the total distance and the number of vehicles.

4.3 Memetic Differential Evolution

In the framework to solve VRPTW, a modified DE is designed with three popular local search techniques which are embedded in the generalized fitness strategy. The proposed memetic differential evolution algorithm based on generalized fitness (MDEGF) is described as follows:

```
Randomly generate NP source individuals S_i
Evaluate the generalized fitness GF(S_i) for all. repeat
for i=1 to NP do
    //Mutation
    select three individuals S_r1, S_r2 and S_r3
        compute S'_off = S_r1 + F · (S_r2 − S_r3)
    // Crossover with rate CR, obtain S_off
    //Selection
    if (GF(S_off) ≤ GF(S_i))
        save index for replacement  S_i = S_off
    end if
end for
perform replacement
until best solution not improved for Gmax iterations
```

The flowchart for MDEGF is the same as the classical DE except the generalized fitness strategy is employed. In the framework for MDEGF, F is a scale factor controlling the rate at which the population evolves, which typically takes values between 0 and 2. CR is the crossover rate which mentioned before complements the differential mutation search strategy. The mutation strategy is the popular DE/rand/1[7] . Other parameters used in the algorithm are $Gmax$ which represents the maximum allowed iterations the best solution without improved and NP the population size.

5 Experimental Results and Comparison

This paper presents the results for applying the MDEGF for VRPTW and compares the results with other recently proposed algorithm. The parameters used in MDEGF are obtained by repeated testing and finally are set as follows: *NP*=200, *Gmax*=300, *F* increases linearly from 0.4 to 1, and *CR* decreases linearly from 0.9 to 0.2. The VRPTW instances investigated are taken from the Solomon[2]. Each of these instances has 100 customers. The best known solutions comes from website [11] . The MDEGF algorithm is applied to solve Solomon's VRPTW instance C202 as an example. Figure1 depicts the best solution to problem C202.

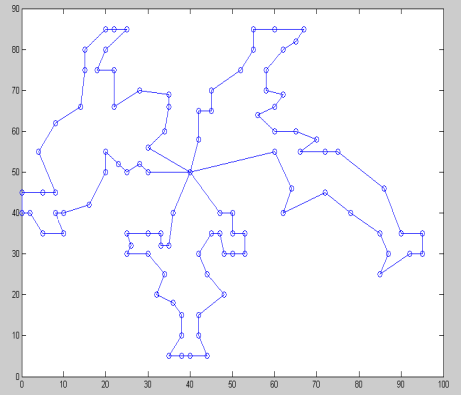


Fig. 1. Best solution for the problem of C202 (591.56/3)

As shown in the figure, there are 100 customers that are serviced by three vehicles. Each route must satisfy the capacity constraint and time windows constraints. The total distance of three routes is 591.56 which is the best known solution.

The proposed algorithm is used to solve problem set C of Solomon's VRPTW instances, in which the customers are clustered in groups. According to whether the time window is narrow or not, C is divided into two sets C1 and C2. Each of the problem instances has been experimented for 20 times independently.

Table 2 and table 3 respectively list the experimental results for problem sets C1 and C2. The solutions are with the form total distance/number of vehicles. The solutions obtained in this paper which are equal to the well-known best solutions[11], are identified by boldface solutions. Instances with solutions superior to those appeared in literature [5] are identified by underlined solutions. Table 2 and table 3 show that the cumulative total distance for problem set C is very close to the best solutions published. The relative error is no bigger than 0.4%. The number of vehicles obtained is the same as the best results of the entire problem set C. The average running time of our algorithm is 76 seconds which is obviously satisfactory to users. Compared with paper [5], our approach manages to apply simple and effective memetic framework to resolve VRPTW. Among those 17 instances of set C, 16 instances have obtained better solutions than literature [5], and 14 instance have obtained better average outcomes. For instances with narrow/wide time windows, the results are more convincing.

Table 1. Computational results on problem set C1

Problem	Best solutions published[11]	Literature [5]	MDEGF Algorithm	Average Length	Length Relative Error(%)	Average CPU(s)
C101	827.3/10	831.97/10	828.94/10	828.94	0.20	60.6633
C102	827.3/10	829.97/11	828.94/10	828.94	0.20	83.9063
C103	828.06/10	831.97/11	828.06/10	828.06	0.00	81.9977
C104	824.78/10	831.97/10	828.09/10	837.64	0.40	84.2132
C105	828.94/10	831.97/11	828.94/10	828.94	0.00	60.8821
C106	827.3/10	831.97/10	828.94/10	828.94	0.20	69.0328
C107	827.3/10	831.97/11	828.94/10	828.94	0.20	62.7977
C108	827.3/10	837.97/10	828.94/10	828.94	0.20	87.2367
C109	828.94/10	840.42/11	828.94/10	828.94	0.00	61.1805

Table 2. Computational results on problem set C2

Problem	Best solutions published[11]	Literature [5]	MDEGF Algorithm	Average Length	Length Relative Error(%)	Average CPU(s)
C201	591.56/3	604.37/3	591.56/3	591.56	0.00	65.2813
C202	591.56/3	591.78/3	591.56/3	591.56	0.00	69.5633
C203	591.17/3	591.34/4	591.17/3	593.09	0.00	86.7586
C204	590.60/3	591.11/3	591.17/3	604.21	0.09	73.0844
C205	588.88/3	589.4/3	588.88/3	588.88	0.00	80.9883
C206	588.49/3	589.13/3	588.49/3	588.49	0.00	88.2071
C207	588.29/3	589.21/4	588.29/3	588.29	0.00	78.3046
C208	588.32/3	595.32/4	588.32/3	588.32	0.00	65.0828

6 Conclusion

In this paper, an improved MDEGF is proposed for VRPTW. The generalized fitness strategy is employed to evaluate the quality of source-individuals and has greatly increased the universal performance of DE algorithm. This method has shown to be very efficient in solving the clustered data (problem set C) VRPTW. For customers up to 100, the proposed algorithm can find the nearly best results in short time.

Acknowledgments. This work is supported by the National Natural Science Foundation of China (No.61171124) and Shenzhen Key Research Funds.

References

1. Dantzig, G.B., Ramser, J.H.: The truck dispatching problem. Management Science 6, 80–91 (1959)
2. Solomon, M.M.: Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations Research 35, 254–265 (1987)
3. Vidal, T., Crainic, T.G., Gendreau, M., Prins, C.: A Hybrid Genetic Algorithm with Adaptive Diversity Management for a Large Class of Vehicle Routing Problems with Time Windows. Tech. Rep. 61, CIRRELT (2011)

4. Potvin, J.Y., Rousseau, J.M.: A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. European Journal of Operational Research 66, 331–340 (1993)
5. Mao, Y., Deng, Y.: Solving Vehicle Routing Problem with Time Windows with Hybrid Evolutionary Algorithm. In: 2010 Second WRI Global Congress on Intelligent Systems (GCIS), Wuhan, pp. 335–339. IEEE Press, New York (2010)
6. Bräysy, O., Gendreau, M.: Vehicle routing problem with time windows, Part I: Route construction and local search algorithms. Transportation Science 39, 104–118 (2005)
7. Qian, B., Wang, L., Huang, D.X., Wang, X.: Scheduling multi-objective job shops using a memetic algorithm based on differential evolution. The International Journal of Advanced Manufacturing Technology 35, 1014–1027 (2008)
8. Storn, R., Price, K.: Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces. International Computer Science Institute-Publications-TR (1995)
9. Sauer, J., dos Santos Coelho, L., Mariani, V., de Macedo Mourelle, L., Nedjah, N.: A Discrete Differential Evolution Approach with Local Search for Traveling Salesman Problems. Innovative Computing Methods and Their Applications to Engineering Problems 357, 1–12 (2011)
10. Pan, Q.K., Tasgetiren, M.F., Liang, Y.C.: A discrete differential evolution algorithm for the permutation flowshop scheduling problem. Computers & Industrial Engineering 55, 795–816 (2008)
11. http://neo.lcc.uma.es/radi-aeb/WebVRP/index.html

Advances in Differential Evolution for Solving Multiobjective Optimization Problems

Hongtao Ye[1], Meifang Zhou[2], and Yan Wu[1]

[1] Department of Electronic Information and Control Engineering, Guangxi University of Technology, Liuzhou 545006, China
[2] Office of the President, Guangxi University of Technology, Liuzhou 545006, China
yehongtao@126.com

Abstract. Differential evolution (DE) is a powerful evolutionary optimization algorithm with many successful scientific and engineering applications. This paper presents a survey of DE for solving multiobjective optimization problems (MOPs). It provides several prominent variants of the DE for solving MOPs. Then it presents an overview of the most significant engineering applications of DE. Finally, it points out the potential future research directions.

Keywords: Differential evolution, Multiobjective optimization problems, Evolutionary algorithms.

1 Introduction

There are two goals in multiobjective optimization problems (MOPs): (ⅰ) to discover solutions as close to the Pareto front as possible and (ⅱ) to find solutions as diverse as possible in the obtained nondominated front. The MOPs can be stated as follow.

$$\begin{aligned} &min \quad \mathrm{F(x)}=(f_1(\mathrm{x}), f_2(\mathrm{x}), \cdots, f_m(\mathrm{x})) \\ &s.t. \quad \mathrm{G(x)}=(g_1(\mathrm{x}), g_2(\mathrm{x}), \cdots, g_m(\mathrm{x})) \geq 0 \end{aligned} \tag{1}$$

where x is a decision vector ($x_1, \cdots, x_n$), F(x) is an objective vector, and G(x) represents constraints.

Differential evolution (DE) was designed to optimize problems over continuous domain by K. Price and R. Storn [1]. DE is a branch of evolutionary algorithms (EAs) for optimization problems over continuous domains. Like other EAs, DE is a population-based stochastic search algorithm. Therefore they can generate a number of Pareto solutions in a single run. DE algorithms have been proposed in the literature to overcome the drawbacks of traditional approaches to MOPs. Since DE algorithms deal with a group of candidate solutions, it seems natural to use them in MOPs to find a group of optimal solutions. Indeed, DE algorithms have proved very efficient in solving MOPs.

The rest of this paper is arranged as follows. The main steps of the DE algorithm are given in Section 2. Section 3 provides an overview of DE for solving MOPs.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 366–373, 2012.

Several prominent variants of DE are provided in Section 4. Section 5 provides an overview of the most significant engineering applications. Section 6 points out the potential future research directions. Section 7 concludes this paper.

2 Differential Evolution (DE)

DE algorithm creates new candidate solutions by combining the parent individual and several other individuals of the same population. A candidate replaces its parent only if it has better fitness. This is a rather greedy selection scheme that often outperforms traditional EAs [2]. Pseudocode for DE is described as follows.

Step 1. Initialize and evaluate population P
Step 2. While stopping criterion not met, do:
  Step 2.1. For each individual P_i from P, repeat:
    Step 2.1.1. Create candidate C from parent P_i
    Step 2.1.2. Evaluate the candidate
    Step 2.1.3. If the candidate is worse than the parent, the candidate is discarded. Otherwise the candidate replaces the parent.
  Step 2.2. Randomly enumerate the individual in P.

Fig. 1. The DE algorithm

3 DE for Solving MOPs

Many DE algorithms were formulated by the researchers to tackle MOPs in the past years. Abbass *et al*. [6] proposed a pareto-frontier DE (PDE) approach, and it was the first to apply DE to MOPs. The PDE employed DE to create new individuals and keep only the nondominated ones as the basis for the next generation. Compared to the strength Pareto evolutionary algorithm (SPEA) [7] on two test problems, PDE was found to outperform it. However the crossover rate of PDE was found to be very sensitive to the solutions. Xue *et al*. [8] proposed a multiobjective DE (MODE). The fitness of an individual was firstly calculated using Pareto-based ranking and then reduced with respect to the individual's crowding distance value in MODE. Yao *et al*. [9] presented a multi-objective DE algorithm, which takes the selection by the non-dominated sorting and crowding distance. The experimental results reported by Yao indicated that the algorithm was better than the nondominated sorting genetic algorithms II (NSGA-II) [10] both in convergence and in diversity. Li and Zheng [11] proposed an improved multiobjective DE algorithm, which incorporated non-dominated sorting and crowding distance to improve the convergence.

Some researchers proposed approaches that use non-Pareto based multiobjective concepts like combination of functions, problem transformation, and so on. Li *et al*. [12, 13] proposed a multiobjective DE algorithm based on decomposition (MOEA/D-DE) for continuous MOPs with variable linkages. The DE/rand/1/bin scheme is used for generating new trial solutions, and a neighborhood relationship among all the sub-problems generated is defined, such that they all have similar optimal solutions. In [12],

they introduced a general class of continuous MOPs with complicated Pareto set shapes and reported the superiority of MOEA/D-DE over NSGA-II with DE type reproduction operators. Summation of normalized objective values with diversified selection approach was used in [14] without the need for performing non-dominated sorting.

4 Important Variants of DE for Solving MOPs

In this section, we shall undertake an in-depth discussion of the most important DE-variants for MOPs.

4.1 DE with Adaptive Parameter Control

DE algorithms have been successfully applied to solve MOPs. However, it is need to choose the suitable parameters to ensure the success of the algorithms. It may lead to demanding computational costs because of the time-consuming trial-and-error parameter and operator tuning process. There are three crucial control parameters in DE algorithms: (i) the population size *NP*, (ii) the mutation scale factor *F*, and (iii) the crossover rate *Cr*.

Self-adaptation allows an evolution strategy to adapt itself without any user interaction [15]. Adaptive parameter control can enhance the robustness of the algorithm by dynamically adapting the parameters to the characteristic of different fitness landscapes [16]. Some researchers developed DE algorithms with adaptation strategy. Abbass [18] proposed a self-adaptive Pareto DE (SPDE) algorithm for multi-objective optimization. The SPDE algorithm self-adapted the crossover rate *Cr* for MOPs. Zaharie and Petcu [17] proposed an adaptive Pareto DE (APDE) algorithm for multiobjective optimization and analyzed its parallel implementation. The numerical tests suggest the APDE algorithm is competitive in solving MOPs on continuous domains when is compared with SPEA and SPDE.

The concept of self-adaptive DE has been extended to handle MOPs recently. Wu *et al*. [19] proposed a multiobjective self-adaptive DE (MOSADE) algorithm for the simultaneous optimization of component sizing and control strategy in parallel hybrid electric vehicles. The MOSADE adopted an external elitist archive to retain nondominated solutions that were found during the evolutionary process. And the MOSADE employed a progressive comparison truncation operator based on the normalized nearest neighbor distance to preserve the diversity of Pareto optimal solutions. Huang *et al*. [20, 21] proposed a multiobjective self-adaptive DE with objective-wise learning strategies to solve numerical optimization problems with multiple conflicting objectives. Zamuda *et al*. [22] proposed a DE for multiobjective optimization with self-adaptation (DEMOwSA) algorithm. Xue *et al*. [23] used a fuzzy logic controller to adjust the parameters of the multiobjective DE algorithm dynamically. The fuzzy logic controlled multiobjective DE (FLC-MODE) was applied to a suite of benchmark functions proposed in [24]. Compared with those results obtained by using MODE with constant parameter settings, the results that the

FLC-MODE obtained were better in 80% of the testing examples. Qian and Li [5] proposed a new adaptive DE algorithm (ADEA) for MOPs.

4.2 DE Based on Opposite Operation

The concept of opposition-based learning (OBL) was original introduced by Tizhoosh [3]. The idea of OBL is the simultaneous consideration of an estimate and its corresponding opposite estimate in order to achieve a better approximation for the current candidate solution.

Dong *et al.* [4] proposed a multiobjective DE algorithm based on opposite operation. The proposed algorithm incorporated opposite operation in two procedures: population initialization and generation operating with opposition. Before discussing the algorithm, we give the definition of the opposite number.

4.3 Hybrid DE Algorithms

Hybridisation primarily refers to the process of combining the best features of more algorithms together, to form a new algorithm that is expected to outperform its ancestors over application-specific or general benchmark problems [25].

Deb *et al.* [26] proposed a hybrid methodology evolutionary and local search approaches. Local search approaches primarily explore a small neighborhood of a candidate solution in the search space until a locally optimal point is found. Niu *et al.* [27] proposed a chaotic DE for multiobjective optimization (CDEMO). In the CDEMO, chaotic sequences are used in the initialization of the evolutionary population and chaotic population candidate is created with chaotic variables to be used in substitution operation. Wang *et al.* [28] proposed a multi-objective chaotic DE algorithm with grading second mutation. Grading second mutation and chaotic theory are combined into standard DE. By testing benchmarks functions, the algorithm is superior to standard DE in keeping balance between diversity and convergence.

Chang and Wu [29] investigated the optimal multiobjective planning of large-scale passive harmonic filters using the hybrid DE (HDE) method. Simulation results show that the HDE offers a good method for multiobjective optimal filter planning of multibus systems. Gujarathi and Babu [30] proposed a hybrid strategy of multiobjective DE (hybrid-MODE) algorithm for the multiobjective optimization of an industrial adiabatic styrene reactor. The hybrid-MODE is consisted of an evolutionary algorithm for global search and a deterministic algorithm for local search.

4.4 DE Based on Multi-populations

Santana-Quintero and Coello Coello [31] presented the ε- MyDE algorithm. This approach keeps two populations: the main population which is used to select the parents and a secondary population, in which the concept of ε- dominance is adopted to retain the nondominated solutions found and to distribute them in a uniform way. Meng *et al.* [32] presented the DE based on double populations for Constrained MOPs. One population is for the feasible solutions found during the evolution, and

the other is for infeasible solutions with better performance, which are allowed to participate in the evolution with the advantage of avoiding difficulties such as constructing penalty function and deleting infeasible solutions directly. Wu *et al*. [33] presented a pseudo parallel DE algorithm with dual subpopulations (DSPPDE). The DSPPDE employs the ideal of isolated evolution and information exchanging in parallel DE algorithm by serial program structure.

5 Engineering Applications of DE for Solving MOPs

Due to the rapidly growing popularity of DE as a simple and robust optimizer, researchers from several domains of science and engineering have been applying DE to solve MOPs arising in their own fields. For the sake of space economy, we summarize only the major applications in Table 1.

Table 1. Engineering applications of DE for solving MOPs

	Sub areas and details	Types of DE applied and references
Signal processing	Digital filter design	Hybrid DE [29]
	Microwave filter design	Generalized DE [34]
	Micro-Array Data Analysis	Multiobjective DE [35]
Chemical engineering	Optimization of adiabatic styrene reactor	Hybrid -MODE [30]
	Optimization of chemical process	Improved DE [36]
Control system	PID regulator design	DE based on double populations[37]
	Multi-objective robust PID controller	Multi-objective DE [38]
Electrical power system	Reactive power optimization considering voltage stability	Self-adaptive MODE [39]
Economics	Economic environmental dispatch	Multiobjective DE [40]
	Portfolio optimization	DEMPO [41]

6 Future Work with DE for Solving MOPs

Like all other metaheuristics to solve MOPs such as particle swarm optimization, DE also has some disadvantages. DE has a high convergence rate, but it has difficulties to reach the true Pareto front. This seems to indicate that multiobjective DE approaches require additional mechanisms to maintain diversity such as crowded-based operators or good mutation operators [42].

The theoretical studies about DE for solving MOPs are still scarce. Not much research has so far been devoted to theoretically analyze the search mechanism. And

the timing complexity analysis of DE for solving MOPs has been reported scarcely. Convergence properties analysis is still a challenging field of future search.

Parameter adaptation is a promising path for future research. Online adaptation attempts are still scarce in multi-objective DE. Novel schemes to adapt the key parameters like "*F*", "*Cr*" or even the number of differences for the mutation operator are promising topics for future research.

7 Conclusions

This paper provides an overall picture of the state-of-the-art research on and with DE for solving MOPs. This paper provides several prominent variants of the DE for solving MOPs. And it provides an overview of the most significant engineering applications. Finally, it points out the potential future research directions. This paper indicates the fact that DE for solving MOPs will continue to remain an active and challenging field in the years to come.

Acknowledgments. This work was supported by the Doctoral Initiating Project of Guangxi University of Technology (No.11Z09), the Key Project of Chinese Ministry of Education (No.212135), Guangxi Natural Science Foundation (No. 2012GXNSFBA053165), and the Project of Education Department of Guangxi Autonomous Region (No. 201010LX242).

References

1. Stron, R., Price, K.: Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11, 341–359 (1997)
2. Robič, T., Filipič, B.: DEMO: Differential Evolution for Multiobjective Optimization. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) EMO 2005. LNCS, vol. 3410, pp. 520–533. Springer, Heidelberg (2005)
3. Tizhoosh, H.R.: Opposition-based learning: A new scheme for machine intelligence. In: Processing of Computational Intelligence for Modelling, Control and Automation, Vienna, Austria, pp. 695–701 (2005)
4. Dong, N., Wang, Y.P.: Multiobjective differential evolution based on opposite operation. In: International Conference on Computational Intelligence and Security, Beijing, China, pp. 123–127 (2009)
5. Qian, W.Y., Li, A.J.: Adaptive differential evolution algorithm for multiobjective optimization problems. Applied Mathematics and Computation 201, 431–440 (2008)
6. Abbass, H.A., Sarker, R., Newton, C.: PDE: A pareto-frontier differential evolution approach for multi-objective optimization problems. In: Proceedings of the Congress on Evolutionary Computation, pp. 831–836. IEEE Service Center Piscataway, New Jersey (2002)
7. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. IEEE Transactions on Evolutionary Computation 3, 257–271 (1999)

8. Xue, F., Sanderson, A.C., Graves, R.J.: Pareto-based multi-objective differential evolution. In: Proceedings of the 2003 Congress on Evolutionary Computation, pp. 862–869. IEEE Press, Canberra (2003)
9. Yao, F., Yang, W.D., Zhang, M.: Multi-objective differential evolution used for load distribution of hot strip mills. Control Theory & Application 27, 897–902 (2010)
10. Deb, K., Pratap, A., Agarwal, S., et al.: A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6, 182–197 (2002)
11. Li, K., Zheng, J.H.: Improved multi-objective evolutionary algorithm based on differential evolution. Computer Engineering and Applications 44, 51–56 (2008)
12. Li, H., Zhang, Q.F.: Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. IEEE Transactions on Evolutionary Computation 13, 284–302 (2009)
13. Li, H., Zhang, Q.: A Multiobjective Differential Evolution Based on Decomposition for Multiobjective Optimization with Variable Linkages. In: Runarsson, T.P., Beyer, H.-G., Burke, E.K., Merelo-Guervós, J.J., Whitley, L.D., Yao, X. (eds.) PPSN 2006. LNCS, vol. 4193, pp. 583–592. Springer, Heidelberg (2006)
14. Qu, B.Y., Suganthan, P.N.: Multiobjective evolutionary algorithms based on the summation of normalized objectives and diversified selection. Information Science 80, 3170–3181 (2010)
15. Eiben, A.E., Smith, J.E.: Introduction to evolutionary computing (natural computing series). Springer, Berlin (2003)
16. Zhang, J.Q., Sanderson, A.C.: JADE: Adaptive differential evolution with optional external archive. IEEE Transactions on Evolutionary Computation 13, 945–958 (2009)
17. Zaharie, D., Petcu, D.: Adaptive pareto differential evolution algorithm and its parallelization. In: Proc. 5th. Conf. Parallel Process. Appl. Math., Czestochowa, Poland, pp. 261–268 (2003)
18. Abbass, H.A.: The self-adaptive pareto differential evolution algorithm. In: Proc. IEEE Congr. Evol. Comput., Honolulu, HI, pp. 831–836 (2002)
19. Wu, L.H., Wang, Y.N., Yuan, X.F., et al.: Multiobjective Optimization of HEV Fuel Economy and Emissions Using the Self-Adaptive Differential Evolution Algorithm. IEEE Transactions on Vehicular Technology 60, 2458–2470 (2011)
20. Huang, V.L., Qin, A.K., Suganthan, P.N., et al.: Multi-objective optimization based on self-adaptive differential evolution algorithm. In: Proceedings of the Evolutionary Computation, pp. 3601–3608 (2007)
21. Huang, V.L., Zhao, S.Z., Mallipeddi, R., et al.: Multi-objective optimization using self-adaptive differential evolution algorithm. In: Proceedings of the Eleventh Conference on Congress on Evolutionary Computation, pp. 190–194. IEEE Press (2009)
22. Zamuda, A., Brest, J., Boškovic, B., et al.: Differential Evolution for Multiobjective Optimization with Self Adaptation. In: IEEE Congress on Evolutionary Computation, pp. 3617–3624 (2007)
23. Xue, F., Sanderson, A.C., Bonissone, P.P., et al.: Fuzzy logic controlled multiobjective differential evolution. In: Proc. IEEE Int. Conf. Fuzzy Syst., Reno, NV, pp. 720–725 (2005)
24. Zitzler, E., Deb, K., Thiele, L.: Comparison of multi-objective evolutionary algorithms: empirical results. Evolutionary Computation 8, 173–195 (2000)
25. Das, S., Suganthan, P.N.: Differential evolution: A survey of the state-of-the-art. IEEE Transactions on Evolutionary Computation 15, 4–31 (2011)

26. Deb, K., Miettinen, K., Chaudhuri, S.: Toward an estimation of nadir objective vector using a hybrid of evolutionary and local search approaches. IEEE Transactions on Evolutionary Computation 14, 821–841 (2010)
27. Niu, D.P., Wang, F.L., He, D.K., et al.: Chaotic differential evolution for multiobjective optimization. Control and Decision 24, 361–364 (2009)
28. Wang, X.Z., Li, P., Yu, G.Y.: Multi-objective chaotic differential evolution algorithm with grading second mutation. Control and Decision 26, 457–463 (2011)
29. Chang, Y.P., Wu, C.J.: Optimal multiobjective planning of large-scale passive harmonic filters using hybrid differential evolution method considering parameter and loading uncertainty. IEEE Transactions on Power Delivery 20, 408–416 (2005)
30. Gujarathi, A.M., Babu, B.V.: Optimization of adiabatic styrene reactor: a hybrid multiobjective differential evolution (H-MODE) approach. Ind. Eng. Chem. Res. 48, 11115–11132 (2009)
31. Santana-Quintero, L.V., Coello Coello, C.A.: An algorithm based on differential evolution for multiobjective problems. International Journal of Computational Intelligence Research 1, 151–169 (2005)
32. Meng, H.Y., Zhang, X.H., Liu, S.Y.: A differential evolution based on double populations for constrained multi-objective optimization problem. Chinese Journal of Computer 31, 228–235 (2008)
33. Wu, L.H., Wang, Y.N., Zhou, S.W., et al.: Research and application of pseudo parallel differential evolution algorithm with dual subpopulations. Control Theory & Applications 24, 453–458 (2007)
34. Goudos, S.K., Sahalos, J.N.: Pareto Optimal Microwave Filter Design Using Multiobjective Differential Evolution. IEEE Transactions on Antennas and Propagation 58, 132–144 (2010)
35. Suresh, K., Kundu, D., Ghosh, S., et al.: Multi-Objective Differential Evolution for Automatic Clustering with Application to Micro-Array Data Analysis. Sensors 9, 3981–4004 (2009)
36. Niu, D.P., Wang, F.L., He, D.K., et al.: Optimization of nosiheptide fermentation process based on the improved differential evolution algorithm for multi-objective optimization. Control Theory & Applications 27, 505–508 (2010)
37. Chen, Y., Bo, Y.M., Zou, W.J., et al.: Satisfactory Optimization for PID Regulator with Constraints on Exceeding Tolerance Characteristic Indices. Information and Control 39, 581–587 (2010)
38. Zhao, S.Z., Qu, B.Y., Suganthan, P.N., et al.: Multi-objective robust PID controller tuning using multi-objective differential evolution. In: International Conference on Control, Automation, Robotics and Vision, pp. 2398–2403 (2011)
39. Qiu, W., Zhang, J.H., Liu, N.: A Self-Adaptive Multi-0bjective Differential Evolution Algorithm for Reactive Power Optimization Considering Voltage Stability. Power System Technology 35, 81–87 (2011)
40. Basu, M.: Economic environmental dispatch using multi-objective differential evolution. Applied Soft Computing 11, 2845–2853 (2011)
41. Krink, T., Paterlini, S.: Multiobjective optimization using differential evolution for real-world portfolio optimization. Computational Management Science 8, 157–179 (2011)
42. Coello Coello, C.A., Lamont, G.B., Van Veldhuizen, D.A.: Evolutionary Algorithms for Solving Multi-Objective Problems. Springer, Berlin (2007)

Fast Mixed Strategy Differential Evolution Using Effective Mutant Vector Pool

Hao Liu[1], Han Huang[1,2], Yingjun Wu[1], and Zhenhua Huang[1]

[1] School of Software Engineering South China University of Technology Guangzhou, P.R. China
[2] Department of Management Sciences, College of Business City University of Hong Kong, Hong Kong
liuhaoscut@gmail.com,hhan@scut.edu.com

Abstract. The mutant vector has significant influence on the performance of Differential Evolution (DE). Different mutant vector always generates different result, one outstanding mutant vector for a specify problem perhaps achieve unbearable bad result for another question. There still no one perfect mutant vector can solve all problems excellently. In this situation, mixed strategy method is proposed to improve the performance of DE by combining multi-effective mutant vectors together. This paper proposes a fast mixed strategy DE (FMDE). The new method uses two best mutant vectors selected from the mutant vector pool and applies a fast mixed method to generate better result without increase computing expense. The FMDE is evaluated by 27 benchmarks selected from Congress on Evolutionary Computation (CEC) competition. The experiment result shows FMDE is competitive, stable and comprehensive. *abstract* environment.

Keywords: Differential Evolution (DE), mutant vector, mixed strategy, fast method.

1 Introduction

DE is proposed by Storn and Price [1] at 1995. At first it is proposed as a new kind of mutant vector that perturbs the population by calculating the difference between two individuals. But with further study of DE, it is accepted widely because of the simple and effective scheme. So even if DE is a case of evolutionary algorithm, it is already distinguished as an independently algorithm and attracted many scientists devoting to it. Ferrante Neri and Ville Tirronen[2] has summed the development of DE and concluded the optimization of DE into two classes: integrating extra component and modify structure of DE. Swagatam Das and P.N. Suganthan [3] conducts a survey of the state-of-art of DE, in which introduces DE's difference to normal EA and the performance of DE in previous CEC competition. It also discussed the applications of DE on discrete, constrained, multi-objective and dynamic problems. This paper concludes the development trend of DE accurately.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 374–381, 2012.

So far, many mutant vectors have been proposed and currently some of them are widely used because of their excellent performance on specific problem. Even the mutant vectors make up a big family, none of them can solve all problems perfectly at the same time. Therefore the mixed method or composite method is proposed. A. K. Qin, V. L. Huang, and P. N. Suganthan proposed a mixed method and proposed an adaptive mixed method with a mutant vector pool at [4].Yong Wang, Zixing Cai and Qingfu Zhang proposed a composite method CoDE[5] with a random method.

This paper proposed a fast mixed method. A mixed strategy would be applied for every several generations. In the mix operation each mutant vector candidate would be used and finally the best one will be used in the succeeding generation. This strategy is effective and need not apply the mix operation in each generation, so the computing expense is smaller than other mixed strategy and the performance is better than other predict methods.

This paper is organized as follows, 2.1 gives a brief description of classic DE, 2.2 introduces the fast mixed method, and 2.3 proposes the mutant vector used in the FMDE. Part 3.1 provides detail of experiment benchmarks and 3.2 presents the setting of the algorithm, and then the result of experiments and analysis consist in 3.3. Lastly conclusion is given in part 4.

2 Fast Mixed DE

2.1 Classic DE

Classic DE is used to find the minimum value of objective function f(x). To symbolize individuals as $X = [x_1, x_2, \ldots, x_{NP}]$, and randomly initial the population in search space S. We would use the benchmark test suite to find $x_{min} \in S$.

The framework of DE:

a) Initialization
Generate NP individuals with D dimension at G generation $x_i = \{x_{i1}^G, x_{i2}^G, \ldots, x_{iD}^G\}$, The individuals are randomly distributed in the prescribed minimum and maximum parameter bound

$$x_{max} = \{x_{1max}^G, x_{2max}^G, \ldots, x_{Dmax}^G\},$$

$$x_{min} = \{x_{1min}^G, x_{2min}^G, \ldots, x_{Dmin}^G\}.$$

b) Mutation operator
The evolution process begins with application of the mutant vector. The mutant vector perturbs individuals in the search space randomly. The classical vector could be expressed as

$$V_i^G = x_{r1}^G + F(x_{r2}^G - x_{r3}^G) \tag{1}$$

r1, r2, r3 are integers randomly selected in the range [1, NP].
c) Crossover
The crossover operator is used to increase the diversity of the population again. It is useful to optimize the performance of DE on multimodal problems [6]. Currently two kinds of crossover operators are widely used: exponential and binomial. This paper uses binomial operator. The binomial operator can be described as:

$$u_{i,j}^{G} = \begin{cases} v_{i,j}^{G}, if(rand_j[0,1) \leq CR) or (j = j_{rand}) \\ x_{i,j}^{G}, otherwise \qquad\qquad j = 1, 2, \ldots, D. \end{cases} \tag{2}$$

The operation would generate a trial vector $U_i^G = [u_{1i}^G, u_{2i}^G, \ldots\ u_{Di}^G]$,.

d) Select
In this step the select operation would determine the trail vector otherwise the target would be used to generate offspring. It can be defined as

$$x_i^{G+1} = \begin{cases} U_i^G, iff(U_i^G) \leq f(x_i^G) \\ x_i^G, otherwise. \end{cases} \tag{3}$$

b)c)d)would be repeated until certain criterion is met.

2.2 Fast Mixed DE Algorithm

In the first part various mixed method is mentioned. All these mixed method achieved better performance by calculating the performance of various mutant vectors. The mix method can be concluded into two classes: 1) calculate the result of each mutant vector and select the best result. If one of the vectors can achieve a good result, it will be inherited and used in following generation.2) Construct a selection model to predict which mutant vector in use in each generation. In general the best mutant vector would be selected in most generation.

While, the performance of the second method highly rely on the accuracy of the modal used to predict the mutant vector, obviously. The predefined model perhaps performs poorly on new problem. Although employing the first method could find the best method from the mutant vector pool, the computing expense rises. When adding one more vector to the mutant pool, the computing expense will stably increase by fixed amount. To avoid the error of the prediction and save the computing expense, this paper proposed the fast mixed method (FMDE).

The FMDE method is based on the first mix method that combines various mutant vectors together. In the mixed process all the mutant vectors will be calculated and all the result will be stored to get into the crossover operation. In the crossover process, all the results of these mutant vectors would be compared with the parent individuals, and the best one would get into next generation. This method also optimized the mutant vector pool, less but more effective mutant vectors are selected into the pool. To save the computing expense, FMDE decides to decrease the frequency of the mix operation. It means the algorithm need not mix these vectors in each generation. In this paper we set the mix

operation applies on every 5 generation. Since after the mix operation only the best mutant vector would in use in next 4 generations, the computing expense would only slightly more than a single mutant vector algorithm.

Although the new method decreased the mix frequency, this fast method would not decrease the performance of DE. In the process of the mixed method, mostly it is one of the mutant vectors achieves best result, so the mix method need not be applied in each generation. In some rare situation if two or more mutant vectors achieve best result at the same time, the mixed method would not influence the result, too, because the mixed operation would select the current best mutant vector and applied in next 4 generations. If the method was wrong to choose other mutant vector, the selection would be corrected in next mix operation -just several generations dose error apply. In the long iteration process the influence of the second best mutant vector is small.

2.3 Selection of the Mutant Vector

The DE mutant vector family is huge. Paper [4] has concluded many DE mutant vectors. [3] giving the naming notation of the mutant vector, while this paper expend these mutant vectors and have a test of the performance of these mutant vectors independently. With the experiment of the mutant vector family benchmarks, it is easy to find the most effective mutant vector. This paper select two mutant vector:

1.DE/rand/1: $V_i^G = x_{r1} + F(x_{r2}^G - x_{r3}^G)$
2.DE/best/2: $V_i^G = x_{best} + F(x_{r2}^G - x_{r3}^G) + F(x_{r4}^G - x_{r5}^G)$

The first one performs best on unimodal problems and rotated problems. The second performs best on multimodal problems. To current test suite, this two mutant vectors have achieved mostly best result, so the FMDE vector pool just select this two mutant vector. When the new method encounters some other new functions or problems, the mutant pool can add other outstanding mutant vector into the pool or delete bad mutant vector. Above all the FMDE has a good expansibility and flexibility.

3 Experiment Studies

3.1 Benchmark Functions for Global Optimization

Test suite of CEC2005, CEC2008 would be carried out to compare the performance of the new DE algorithm with other DE mutant algorithms. The unimodal functions, multimodal functions could be seen at paper [6], rotated functions could be found at paper [7] and noise functions could be found at paper [8]and [9]. The performance of the modified method would be measured by mean value and standard deviation over 50 independent runs.

This paper uses 23 benchmarks to compare these algorithms. f1-f6 are unimodal functions, f7-f13 are multimodal functions with many local minima, f14-f20 are multimodal functions with a few local minima, f18-f20 are multimodal

functions with deceiving, above functions are seen in [6]. f21-f23 are rotated functions. These rotated functions are f7,f10,f11 multiply an orthogonal matrix. They can be found at [7]. Restricted by the length of the paper, partly experiment result is listed.

3.2 Adaptive Method and the Parameter Setting

The DE algorithm is highly simplified than other Evolution Algorithm (EA). It has only three control parameters F, CR and NP. We NP left for the user to define, this adaptive method focus on parameters F and CR. Parameter F is the mutation scale factor which can be found at (1). F is highly related to the convergence speed. CR is a constant value in the crossover operator. CR is sensitive to different problems. The effects of them are well studied at [6] and [10], respectively.

Table 1. Selected Benchmark Functions

Test Functions	D	S	f_{min}
$f_1(x) = \sum_{i=1}^{D} x_i^2$	30	$[-100, 100]^D$	0
$f_2(x) = \sum_{i=1}^{D} \|x_i\| + \Pi_{i=1}^{D} \|x_i\|$	30	$[-100, 100]^D$	0
$f_3(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	30	$[-100, 100]^D$	0
$f_4(x) = \max_i \{\|x_i\|, 1 \geq i \leq D\}$	30	$[-100, 100]^D$	0
$f_5(x) = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-100, 100]^D$	0
$f_6(x) = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$	30	$[-100, 100]^D$	0
$f_8(x) = \sum_{i=1}^{D} -x_i sin(\sqrt{\|x_i\|})$	30	$[-100, 100]^D$	0
$f_9(x) = \sum_{i=1}^{D} [x_i^2 - 10cos(2\pi x_i) + 10]$	30	$[-100, 100]^D$	0
$f_{10}(x) = -20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - exp(\frac{1}{D}\sum_{i=1}^{D} cos2\pi x_i) + 20 + e$	30	$[-100, 100]^D$	0
$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \Pi_{i=1}^{D} cos(\frac{x_i}{\sqrt{i}}) + 1$	30	$[-100, 100]^D$	0
$f_{14}(x) = [\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij}^6)}]^{-1}$	30	$[-100, 100]^D$	0
$f_{21}(x) = -\sum_{i=1}^{5} (\sum_{j=1}^{4} (x_j - a_{ij})^2 + c_i)^{-1}$	30	$[-100, 100]^D$	0
$f_{22}(x) = -\sum_{i=1}^{7} (\sum_{j=1}^{4} (x_j - a_{ij})^2 + c_i)^{-1}$	30	$[-100, 100]^D$	0
$f_{23}(x) = -\sum_{i=1}^{10} (\sum_{j=1}^{4} (x_j - a_{ij})^2 + c_i)^{-1}$	30	$[-100, 100]^D$	0
$f_{26}(x) = \frac{1}{4000}\sum_{i=1}^{D} y_i^2 - \Pi_{i=1}^{D} cos(\frac{y_i}{\sqrt{i}}) + 1, y = M * x$	30	$[-100, 100]^D$	0
$f_{27}(x) = \sum_{i=1}^{D} (y_i^2 - 10cos(2\pi y_i) + 10), y = M * x$	30	$[-100, 100]^D$	0

To ensure all DE performance are in the same conditions, this paper set NP=100, dimension=30 as constant value, F=0.5 and CR=0.9. The parameter F in FMDE is nearly normal distribution. Iteration number of each function is given in TABLE1. Previous researches find that if the parameter F maintains in small value at the exploitation and large value at exploration state, the algorithm would achieve better performance. Various state of the search process can be found at paper [10]. This paper uses the primary adaptive method proposed by [6] to assure this setting. The parameter would increase or decrease based on the evolution state, and the size of the various is decided by multiply a random number.

3.3 Analysis of the Experiment Result

This part we compare the result between FMDE with other excellent DE. Restricted by the length of the paper, we select part of the experiment from the test suite.

Here we conclude the performance of these algorithms of various benchmarks:

Table 2. Experiment Result of compared algorithms over 50 Independent Runs

benchmark	generation		FMDE	ADE	DE	ADE(Rand/1)	ADE(Best/2)
f1	1500	Mean	3.63e-065	2.51E-28	7.62E-10	2.71e-19	5.56e-49
		Std.Dev	9.56e-065	1.91E-28	4.42E-014	1.79e-19	9.11e-49
f2	2000	Mean	2.28e-044	1.50E-24	2.70E-10	4.18e-013	3.53e-34
		Std.Dev	2.25e-044	1.03E-21	2.56E-010	2.59e-013	4.11e-34
f3	5000	Mean	3.81e-065	1.45e-048	6.80E-11	1.74e-016	8.72e-49
		Std.Dev	8.94e-065	1.35E-07	3.82E-011	1.25e-016	1.52e-48
f4	5000	Mean	8.51e-005	3.15E-03	1.80E-03	5.74e-001	1.58e-07
		Std.Dev	4.29e-004	5.31E-4	0.91	1.245e+00	2.69e-07
f6	1500	Mean	0	0	0	0	0
		Std.Dev	0	0	0	0	0
f10	1500	Mean	4.141e-15	6.90E-15	5.61E-08	1.24e-010	6.07e-01
		Std.Dev	0	79.94E-16	2.96E-08	4.91e-011	7.59e-01
f11	2000	Mean	0	0	0	0	9.22e-03
		Std.Dev	0	0	1.38E-3	0	1.13e-02
f14	100	Mean	9.98e-001	0.99892	0.998004	9.98e-001	1.15e-001
		Std.Dev	0	1.71E-3	3.60e-003	0	0
f21	100	Mean	-5.13e+00	-1.000	-1.015	-2.73e-01	-4.19e+0
		Std.Dev	3.46e+00	1.04	1.19E-007	7.84e-002	3.29e-001
f22	1500	Mean	-5.378e+0	-1.040	-1.040	-2.69e-01	-4.11e+0
		Std.Dev	3.52e+000	2.92E-7	3.9E-7	6.07e-002	2.25e-001
f23	3000	Mean	-5.143e+0	-1.054	-1.040	-2.74e-01	-4.51e+0
		Std.Dev	3.47e+000	3.15E-07	1.19E-007	6.91e-002	2.70e+00
f26	1500	Mean	0	3.61e-03	2.09e+001	0	1.13e-02
		Std.Dev	0	5.853-03	1.88e-002	0	1.09e-02
f27	2000	Mean	1.49e+01	1.044e+01	1.536e+02	1.55e+002	3.72e+01
		Std.Dev	5.44e+001	4.0476	9.982	1.063e+01	1.28e+01

To f1-f6 unimodal functions, FMDE achieves best result, and the second best DE is DE with Best/2 mutant vector. This means the mixed algorithm can optimize the performance than single mutant vector.

To f7-f13 multimodal functions with many local minima, FMDE also achieves best result, but it does not outperform other algorithms from a distance.

To f13-f23 multimodal functions with few local minima, FMDE achieves best mean value, but the standard deviation FMDE does not beaten other De algorithms.

To the rotated benchmarks, FMDE is the best algorithm, however it does not mean it solves rotated problem well. In general all the algorithms perform weak on rotated problem.

Conclusion and analysis of the experiment result: FMDE outperforms other DE algorithms used to compare at most situation. The experiment result shows

the mix method can optimize the performance of DE. The mean value and standard deviation of FMDE is smaller than any other DE with only one mutant vector. It means the mix operation help the algorithm optimized the search ability. This optimization could be seen at the mean value of FMDE at each kind of problems.

However, the standard deviation of FMDE at multimodal function with few local optima is large than other DE with only one mutant vector. The reason is the mix strategy lack operation which helps individuals out of local optima. This problem awaits further study. In a word, FMDE is a simple but effective method to solve various problems.

4 Conclusion

The FMDE modifies classic DE framework, adds a mix operation at the crossover operation for every several generation. It also solves the shortcoming of traditional mix strategies which increased the computing expense and lost accuracy in the prediction process. The fast mix strategy also optimizes the performance better than single mutant vector does. On the other side, the fast method lost excellent stability in multimodal functions with few local optima, and this shortcoming worth further study. Above all the FMDE method is more competitive than other DE algorithms and easy to modify and expand.

Acknowledgment. This work was supported by Innovative Practice Project of Guangdong Province (S1056111113), National Students Innovative Pilot Scheme Project (111056173), National Natural Science Foundation of China (61170193, 61070033), Doctoral Program of the Ministry of Education (20090172120035), Guangdong Natural Science Foundation(9251009001000005,10151601501000015), Science and Technology Planning Project of Guangdong Province (2010B050400011,2010B080701070,2008B080701005), the Fundamental Research Funds for the Central Universities, SCUT (2012ZM0083) and the Pearl River Science & Technology Start Project (2012-07).

References

1. 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)
2. Neri, F., Tirronen, V.: Recent Advances in Differential Evolution: A Survey and Experimental Analysis. Artif. Intell. Rev. 33(1), 61–106 (2010)
3. Das, S., Suganthan, P.N.: Differential evolution: A survey of the state-of-the-art. IEEE Trans. on Evolutionary Computation (2011), doi:10.1109/TEVC.2010.2059031
4. 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)

5. Wang, Y., Cai, Z., Zhang, Q.: Differential Evolution with Composite Trial Vector Generation Strategies and Control Parameters. IEEE Trans. Evol. Comput. 15(1), 55–66 (2011)
6. Brest, J., Greiner, S., Boskovic, B., Mernik, M., Zumer, V.: Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. IEEE Trans. Evolut. Comput. 10(6), 646–657 (2006)
7. Mezura-Montes, E., Velazquez-Reyes, J., Coello, C.A.: A comparative study of differential evolution variants for global optimization. Proc. Genet. Evol. Comput., 485–492 (2006)
8. Caponio, A., Neri, F.: Differential Evolution with Noise Analyzer. 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. 715–724. Springer, Heidelberg (2009)
9. Liu, B., Zhang, X., Ma, H.: Hybrid differential evolution for noisy optimization. In: Proc. IEEE Congr. Evol. Comput., vol. 2, pp. 1691–1698 (2005, 2009)
10. Zhan, Z., Zhang, J., Li, Y., Chung, H.S.: Adaptive Particle Swarm Optimization. IEEE Trans. On Systems, Man, and Cybernetics 39(6), 1362–1381 (2009)

Differential Annealing for Global Optimization

Yongwei Zhang, Lei Wang, and Qidi Wu

College of Electronics and Information Engineering
Tongji University, Cao'an Road 4800, Shanghai 201804, P.R. China
yongwzhang@gmail.com

Abstract. This paper propose a hybrid stochastic approach called differential annealing algorithm. The algorithm integrated the advantages of differential evolution and simulated annealing. It can be considered as a swarm-based simulated annealing with differential operator or differential evolution with the Boltzmann-type selection operator. The proposed algorithm is tested on benchmark functions, along with simulated annealing and differential evolution. Results show that differential annealing outperforms the comparative group under the same amount of function evaluations.

Keywords: Swarm Intelligence, Differential Evolution, Simulated Annealing, Stochastic Search; Selection Operator, Global Optimization.

1 Introduction

Among all swarm-based algorithms, differential evolution (DE) [1,2] is one of the most succinct algorithms which integrated neighborhood generation and information exchange into to one operation. But the population diversity of DE drops fast with the decreasing of individual distance, resulting in premature convergence.

Compared with swarm-based optimization, simulated annealing (SA) [3] is a representative stochastic algorithm based on single state or point. But more importantly, it introduced a stochastic acceptance criterion. Unlike the greedy criterion that rejects all inferior states; stochastic criterion enables some inferior state to act as a bridge to the betters.

To exploit the advantages of differential operator and stochastic acceptance criterion, many efforts have been made [4,5,6,7]. However, most of these methods either simply add Metropolis acceptance criterion into DE framework or introduce a stochastic local search operator. This block-type integration does not fully develop the potential of DE and SA because 1) the inferior solutions introduced by stochastic acceptance may affect the memory of populations of search space; 2) inferior states may drift far away from best history record. For overcoming these drawbacks and closely integrating DE and SA, we developed a two stage algorithm with trace mechanism called Differential Annealing (DA). The method is shown to be better on an eight-function test bed.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 382–389, 2012.

2 Simulated Annealing and Differential Evolution

A D-dimension unconstrained minimization problem is defined as follow.

$$\min f_{\text{obj}}(\boldsymbol{x}), \boldsymbol{x} = [x_1, \cdots, x_D] \tag{1}$$

where D is the dimension of the problem or the number of parameters to be optimized, $\boldsymbol{x}$ is a solution of the problem, $x_{i\,\min} \leq x_i \leq x_{i\,\max}$.

2.1 Simulated Annealing

The framework of SA is consisted by three parts: 1) neighborhood generation; 2) annealing schedule and 3) acceptance probability. Despite many variants of these parts, we chose the most common form to define a simple SA algorithm.

Neighborhood Generation. SA uses Metropolis algorithm to draw samples form the neighborhood of current state $\boldsymbol{x}_t$. In most cases, the samples are draw from probability distribution $P(\boldsymbol{x}_t)$, which can be symmetric or not. Here a Gaussian distribution

$$\boldsymbol{x}' \sim N(\boldsymbol{x}_t, \sigma^2 \boldsymbol{I}) \tag{2}$$

is used to give samples $\boldsymbol{x}'$ centered on the current state $\boldsymbol{x}_t$ with variance $\sigma^2 \boldsymbol{I}$.

Annealing Schedule. The annealing schedule is decided by initial temperature T_{ini}, final temperature T_{final} and temperature function $T_k = f(T_{\text{ini}}, T_{\text{final}}, k)$, where k is the index of the schedule and T_k is the current temperature. Note that $k \neq t$ as the current state may change serval times at one temperature.

Acceptance Probability. In SA, the new state $\boldsymbol{x}'$ will be accepted by probability $P[C(\boldsymbol{x}'), C(\boldsymbol{x}_t), T_k]$, where $C(\cdot)$ is the cost function or the energy of state. In our work, the evaluation of object function f_{obj} is used as cost function. The formal definition of SA's acceptance criterion is as follow:

$$\boldsymbol{x}_{t+1} = \begin{cases} \boldsymbol{x}' & \text{if } C(\boldsymbol{x}') < C(\boldsymbol{x}_t) \vee \exp(\frac{-\Delta C}{T_k}) > \text{rand} \\ \boldsymbol{x}_t & \text{else} \end{cases} \tag{3}$$

where ΔC equals to $C(\boldsymbol{x}') - C(\boldsymbol{x}_t)$. Eq. (3) states that the acceptance probability of $\boldsymbol{x}'$ will be 1 if it is better and be $\exp(-\Delta C/T_k)$ if worse, otherwise the state will not change.

2.2 Differential Evolution

Storn defined many variants of DE[1], but practically only one form is broadly used:

$$x'^d_i = \begin{cases} x^d_i + F \cdot (x^d_{r1} - x^d_{r2}) & \text{if rand} < CR \\ x^d_i & \text{else} \end{cases}, \; r1, r2 \in [1, \cdots N] \vee r1 \neq r2 \neq i \tag{4}$$

where x_i^d is d-th element located in i-th individual of the population, N is the number of population, CR is crossover probability and F is a real and constant factor which controls the amplification of the differential variation $(x_{r1}^d - x_{r2}^d)$. The standard DE uses a greedy acceptance criterion.

3 Differential Annealing Algorithm

SA draws samples from a stochastic distribution centered on the current state, which restrains most trials in a small area. This feature facilitates local search of SA, but weakened the global convergence speed.

Since our goal is to design a hybrid algorithm that integrates the advantages of SA and DE, three issues need to discuss first: 1) neighborhood diameter, 2) annealing schedule, 3) integration with DE.

3.1 Neighborhood Diameter

As we stated in section 2, the new state can be draw from a Gaussian distribution centered on current state, here we define the variance of the Gaussian distribution as the diameter of neighborhood, which can be a measure of the coverage of local search. Clearly the neighborhood diameter needs to be adjusted according to the searching space. Here the neighborhood diameter σ is defined as: $\max(X_\mathrm{u} - X_\mathrm{l})/L$, where X_u and X_l is the upper and lower feasible bounds of state vector $\boldsymbol{x}$. L is the granularity of local search, usually within [10,1000], which controls the precision of local search.

3.2 Annealing Schedule

Once the initial and final temperatures are decided, the annealing schedule becomes a function of change index of temperature k. Normally in evolution algorithm, k is the index of generation. This function describes how the temperature drops from the initial to the final. Among various annealing schedules, the following is the most common one: $T_{k+1} = T_k \cdot \alpha$, where $\alpha \in (0, 1]$ controls the dropping rate of temperature. Since initial and final temperatures are decided beforehand, α can be decided by $\alpha = (T_\mathrm{final}/T_\mathrm{ini})^{1/N_\mathrm{G}}$, where N_G is the total number of generations, and the k-th temperature can be decided by $T_k = T_\mathrm{ini} \cdot \alpha^k$.

3.3 Integration with DE

If we build a population upon the scheme of SA, the annealing will have multiple starts and parallel processes. To communicate with individuals and share information of searching space is the goal of introducing differential operator. In the original DE, the differential is used directly on individuals. Because of the existence of greedy criterion, individual itself is the best record. But in our approach, some inferior states can be accepted due to (3), we have to introduce

a best record mechanism and make the differential operator works on the best records. The simplified differential operator in our approach is as follow:

$$x'^{d}_{i,\text{best}} = \begin{cases} x^{d}_{i,\text{best}} + x^{d}_{r1,\text{best}} - x^{d}_{r2,\text{best}} & \text{if rand} < \text{CR} \\ x^{d}_{i,\text{best}} & \text{else} \end{cases} \\ d \in [1, D] \wedge i, r2, r3 \in [1, N] \wedge r1 \neq r2 \neq i \tag{5}$$

where $x^{d}_{i,\text{best}}$ denotes the d-th parameter in the best record of $\boldsymbol{x}_i$. The crossover is conducted on every state vector and other two randomly selected vectors. The greedy criterion is used to ensure the best state is recorded.

3.4 Trace Mechanism

As discussed in section 3.3, a record of the best individuals is maintained and the differential operator works directly on it. This scheme does not influence the true states of the system, which leaves a potential drawback that a state may continually drift to higher energy state. To avoid this, a trace mechanism is introduced as follow:

$$\boldsymbol{x}_i \leftarrow \boldsymbol{x}_{i,\text{best}} \text{ if } C(\boldsymbol{x}_i) > E \cdot C(\boldsymbol{x}_{i,\text{best}}) \tag{6}$$

where $E \geq 1$ is overflow coefficient of energy. Eq. (6) states that $\boldsymbol{x}_i$ will be restored to its history best if its cost function (or energy) is larger than its history best multiples E, i.e., a state will be drag back if it is too far away from the best energy state. This mechanism ensures that the local search is conducted around $\boldsymbol{x}_{i,\text{best}}$. When $E = 1$ the search rejects all states that have higher energy than $\boldsymbol{x}_{i,\text{best}}$. An example of controlling state through trace mechanism is showed at Fig. 1. The fourth step (S4) has higher energy state than history best, if the trace mechanism is not work, the higher energy state will probably been accepted as the left subfigure. With trace mechanism, state will be draw again centered on its history best. The pseudo code of proposed approach is showed at Algorithm 1, here we use maximum number of function evaluation $FE_{\max}$ as the terminal criterion.

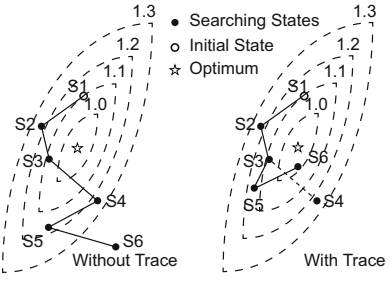


Fig. 1. Trace Mechanism

Algorithm 1. Differential Annealing

```
Cost function C(x) = f_obj(x), x = (x_1, ··· , x_D)
Initialize N start points x_i(i = 1, 2, ··· N), evaluate points and record x_{i,best}
T_0 ← T_ini; k = 0; FE ← N
while FE < FE_max do
  for each state x_i do
    Search neighborhood according to (2); Accept new position according to (3)
    Restore x_i according to (6); FE ← FE + 1
  end for
  for each record x_{i,best} do
    for each element x^d_{i,best} do
      Crossover by (5)
    end for
    Update x_{i,best}; FE ← FE + 1
  end for
  T_{k+1} ← T_k · α; k ← k + 1
end while
Post process results and visualization
```

4 Validation Experiments

To verify the proposed algorithm, numerical experiments are conducted. The performance of DA is compared with SA and DE. The parameter are set as follow: $L = 100$, $F = 1$, $T_{\text{ini}} = 10e + 300$, $T_{\text{final}} = 10e - 10$, $FE_{\max} = 2e5$, $CR = 0.15$, $N = 30$, $\alpha = 0.8670$.

For SA, detecting the equilibrium of system at each temperature will cause additional computing costs, sometime the costs will exceed the cost of evaluating problem. To avoid this dilemma, we change temperature in a same pace in SA and DA, i.e., the temperature of DA will change in every generation (every $2N$ evaluations) and the temperature of SA will change after every $2N$ evaluations. This means that SA has the Markov chain of length $2N$ in every temperature.

4.1 Benchmark Functions

The benchmark functions are usually used as standard test bed for optimization algorithms, which are listed at Table 1. To avoid the unintentional attraction of zeros, the benchmark functions are shifted with a displacement $\boldsymbol{d} = (d_1, d_2, \cdots d_D)$ according to following formula:

$$\begin{aligned} \boldsymbol{z} &= \boldsymbol{x} - \boldsymbol{d} \\ C(\boldsymbol{x}) &= f_{\text{obj}}(\boldsymbol{z}) \end{aligned} \tag{7}$$

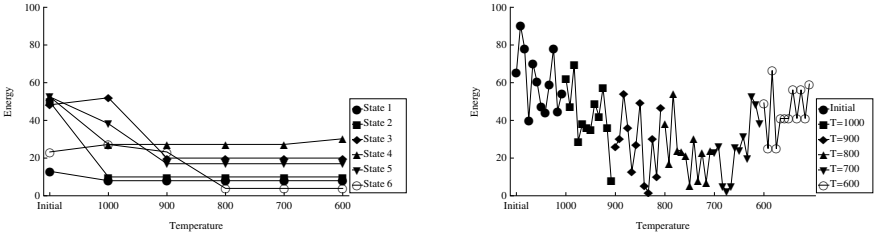
To test the overall performance of proposed algorithm, we try to cover all kinds of test functions. As the characteristic showed at Table 1, three functions are unimodal and five are multimodal. Meanwhile, five functions are separable and three are non-separable. The parameters of Shekel' problem can be find in [8].

Table 1. Benchmark Functions. D: Dimension, F: Feasible Bounds, C: Characteristic, U: Unimodal, M: Multimodal, S: Separable, N: Non-Separable, R: Regular, I: Irregular.

Name	Equation	D	F	C	**d**
Sphere	$f_1 = \sum_{i=1}^{D} x_i^2$	30	[-100, 100]	USR	25^D
Sum of Different Power	$f_2 = \sum_{i=1}^{D} \lvert x_i \rvert^{i+1}$	30	[-1, 1]	USI	0.25^D
Schwefel 1.2	$f_3 = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	30	[-65.536, 65.536]	UNR	16.384^D
Michalewicz	$f_4 = -\sum_{i=1}^{D} \sin(x_i) \left[\sin(ix_i^2/\pi)\right]^{20}$	30	$[0, \pi]$	MSR	0^D
Rastrigin	$f_5 = \sum_{i=1}^{D} \left[x_i^D - 10\cos(2\pi x_i) + 10\right]$	30	[-5.12, 5.12]	MSR	1.28^D
Non-continuous Rastrigin	$f_6 = \sum_{i=1}^{D} \left[y_i^D - 10\cos(2\pi y_i) + 10\right]$ $y_i = \begin{cases} x_i & \lvert x_i \rvert < \frac{1}{2} \\ \frac{\text{round}(2x_i)}{2} & \lvert x_i \rvert \geq \frac{1}{2} \end{cases}$	30	[-5.12, 5.12]	MSI	1.28^D
Griewank	$f_7 = \frac{1}{4000} \sum_{i=1}^{D} x_i^D - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1$	30	[-600, 600]	MNR	150^D
Shekel	$f_8 = -\sum_{i=1}^{10} (\sum_{j=1}^{D} (x_j - a_{ij})^2 + c_i)^{-1}$	4	[0, 10]	MNR	0^D

4.2 Simulation Results

Convergence Test. To show the benefit that information sharing brings to individual states, we tested SA and DA under same conditions of temperature and amount of function evaluations. The states of two algorithms are showed at Fig. 2. The test is carried on with 2-D Non-continuous Rastrigin function. The population size of DA is 6, and the temperature is changed from 1000 to



(a) DA States (b) SA States

Fig. 2. States of DA and SA

600. The first point of curve is the initial energy of every state, the second point is the energy of temperature 1000, then 900 and so forth. The curve of SA is the change of one state in six stages: initial, temperature 1000 to 600. As we can see, in SA the energy level of state go up and down repeatedly, although it has accessed the lower energy state in the initial level, it still end up with a relatively high energy when temperature drops to 600. But in DA, six parallel states present a decrease tendency of almost monotonous.

Table 2. Comparison of Three Algorithms

	Function	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
	ε	1.00e-8	1.00e-8	1.00e-8	1.00e-4	1.00e-7	1.00e-6	1.00e-6	1.00e-8
DA	Best	0.00	0.00	0.00	-9.66	3.19e-8	3.74e-7	5.60e-8	-10.54
	Median	0.00	0.00	1.14e-8	-9.66	6.81e-8	1.00	2.30e-7	-10.54
	Worst	0.00	0.00	2.40e-8	-9.66	2.19e-7	3.00	1.53e-6	-10.54
	Mean	0.00	0.00	1.18e-8	-9.66	7.85e-8	0.73	3.35e-7	-10.54
	Std	0.00	0.00	0.00	0.00	4.16e-8	0.81	3.12e-7	4.34e-8
	Scr	100.0%	100.0%	38.0%	100.0%	76.0%	14.0%	96.0%	94.0%
SA	Best	39.97	1.86e-8	250.16	-7.99	141.23	143.11	1.39	-10.53
	Median	56.86	5.94e-7	343.78	-6.30	228.26	229.15	1.51	-4.41
	Worst	69.39	1.52e-6	540.73	-5.11	313.38	322.07	1.65	-2.63
	Mean	55.40	6.07e-7	348.49	-6.37	227.85	227.79	1.51	-5.64
	Std	6.45	3.58e-7	61.91	0.70	40.20	42.70	0.06	2.82
	Scr	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
DE	Best	0.00	0.00	0.00	-9.66	15.69	32.67	0.0	-10.54
	Median	0.00	0.00	0.00	-9.45	83.48	81.43	0.0	-10.54
	Worst	0.00	0.00	0.00	-7.89	133.97	118.62	7.40e-3	-2.42
	Mean	0.00	0.00	0.00	-9.36	84.25	81.25	1.48e-4	-8.37
	Std	0.00	0.00	0.00	0.32	27.56	22.97	0.001	2.95
	Scr	100.0%	100.0%	100.0%	0.0%	0.0%	0.0%	98.0%	64.0%

Benchmark Test. To further demonstrate the effectiveness of proposed algorithm, 50 runs are conducted for each algorithm, and the results are summarized at Table 2. If the cost function value C_{best} satisfy $|C_{\text{best}} - f_*| < \varepsilon$, where f_* is the global minimum of object function, a success run is recorded. For each benchmark function, the best, median, worst, mean, standard deviate (Std) and success (Scr) rate of 50 runs are listed. We can conclude from Table 3 that SA failed all test under corresponding accuracy, and did not find an even close solution in six functions. This proves that SA lacks of efficiency in high dimension problems. Compared with SA, DA has steady performance in all test problems. In the first two unimodel functions, DA and DE have same results. In the third unimodel function, Schwefel 1.2 problem, DA has lower success rate. But note that solutions obtained by DA are very close to the limit of tolerance, the result is in fact satisfactory. In the rest five multi-model functions, DA outperforms DE in four except function 7. In function 7, Griewank's problem, the success rate of DA is slightly lower than DE (2%), but the mean value and standard deviate of DA are lower for three magnitude level, indicating that DA has more stable performance. Especially for function 5 and 6, DE stuck in local minima, but DA still find optimum with high accuracy.

Overall, the proposed differential annealing algorithm has stable performance for either unimodel or multi-model problems. For high dimension multi-model problems, results showed that DA outperforms SA and DE.

5 Discussion and Conclusion

The heuristics of classic optimization algorithms continually inspires researchers to develop high-efficiency hybrid algorithms. In order to successfully integrate SA into proposed algorithm, we managed to solve an important issue concerning to information sharing between annealing states. By using the set of best records of searching history, the searching progress of single annealing state won't be interrupted until it exceeds the upper limit of energy. By introducing the trace mechanism, the Gaussian distribution and acceptance criterion act as a stochastic local search operator. Results show that the DA approach has achieved follow goals: 1) control the search progress of states, 2) synthesize the information acquired in search and 3) increase the diversity of population.

Acknowledgements. The research is supported by Ph.D. Programs Foundation of Ministry of Education of China (No. 20100072110038), National Natural Science Foundation of China (NSFC, No. 70871091, 61075064, 61034004, 61005090), Program for New Century Excellent Talents in University of Ministry of Education of China.

References

1. Storn, R., Price, K.: Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J. Global. Optim. 11(4), 341–359 (1997)
2. Storn, R., Price, K.: Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical report, International Computer Science Institute, Berkley (1995)
3. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science 220(4598), 671–680 (1983)
4. Das, S., Konar, A., Chakraborty, U.K.: Annealed differential evolution. In: IEEE Congress on Evolutionary Computation, CEC 2007, pp. 1926–1933 (2007)
5. Liu, K., Du, X., Kang, L.: Differential Evolution Algorithm Based on Simulated Annealing. In: Kang, L., Liu, Y., Zeng, S. (eds.) ISICA 2007. LNCS, vol. 4683, pp. 120–126. Springer, Heidelberg (2007)
6. Peichong, W., Xu, Q., Yu, Z., Ning, L.: A novel differential evolution algorithm based on simulated annealing. In: Control and Decision Conference (CCDC), 2010, Chinese, pp. 7–10 (2010)
7. Olenšek, J., Tuma, T., Puhan, J., Bűrmen, A.: A new asynchronous parallel global optimization method based on simulated annealing and differential evolution. Appl. Soft Comput. 11(1), 1481–1489 (2011)
8. Karaboga, D., Akay, B.: A comparative study of artificial bee colony algorithm. Appl. Math. Comput. 214(1), 108–132 (2009)

The Application of Genetic Algorithm to Intrusion Detection in MP2P Network

Lu Li[1,⋆], Guoyin Zhang[1], Jinyuan Nie[2], Yingjiao Niu[1], and Aihong Yao[1]

[1] Harbin Engineering University, Harbin, 150001, China
lilu_1231@163.com, {zhangguoyin,yaoaihong}@hrbeu.edu.cn,
niuyingjiao@yeah.net
[2] Defence Industry Secrecy Examination and Certification Center
kevin_psguy@163.com

Abstract. With the rapid development of MP2P network, high network security is required. However, the existing intrusion detection scheme for network security does not perform well enough. Aiming at the characteristics of MP2P, this paper proposes an intrusion detection method based on the genetic algorithm, which selects the initial population from the known attack data, extracts data attack characteristics by reproducing, crossover, and mutating themselves, and thus changes from passive defense to active detection. Research results verify that, the application of genetic algorithm could enhance intrusion detection in terms of the dynamic monitoring of internal and external network attacks, and thus gains real-time protection for MP2P networks.

Keywords: MP2P, genetic algorithm, intrusion detection, eigenvector, network security.

1 Introduction

With the development of wireless network communication technology, MP2P (Mobile Peer-to-Peer)network as the actual application of the relevant technology production gets extensive research and application with its characteristic gradually[1].

MP2P network is consisted by several communication nodes which with computing power, wireless communication ability, and limited power consumption. The advantages of MP2P are mobility, high signal noise ratio, and dynamic, therefore, it's a self-organizing network with the scalable framework structure[2]. However, these characteristics of MP2P network also resulted in its face two following problems:

(1) MP2P network is prone to be attacked by Wormhole, False Routing Information, Sybil, Sinkhole, hello Flooding Attacks and Selective Forwarding Attacks, etc., which is due to the characteristics of the wireless network communication.

⋆ Project supported by the National Natural Science Foundation of China (No. 61073042) and the special funds of basic scientific research business expenses for central university (No. 100606, HEUCF).

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 390–397, 2012.

(2) The volume of mobility nodes are restricted, therefore, the performance of power, computing power and storage space are limited. In this case, intrusion detection algorithm should spend less overhead and get more effect, therefore, efficiency is considered when choosing detection algorithm.

According to these defects, an intrusion detection mechanism is presented in this paper, which based on genetic algorithm, the purpose is to improve the detection accuracy, reduce the storage space, limit the power consumption, and take on the role of the protection measures, and thus the robustness of MP2P network could be increased.

2 Related Technology

Invasion is defined as a series of act which is unauthorized access or harmful to resource's integrity, usability and privacy. The intention of the invasion, which attempts to destroy the network security protocol, is to capture information, or make normal communication operation of the system in disorder[3].

It is important to establish secure channel in MP2P. MP2P as a self-organization network need to establish secure communication channel due to the characteristics of working environment. But the wireless network, which is exposed to the mobile environment, is more vulnerable to be attacked and give out important information. The purpose of Intrusion Detection (ID) is to monitor if there are unusual events in the network traffic and make analysis, and then to take effective action against the invasion[4].

The intrusion detection is classified into two categories: misuse detection and anomaly detection[5]. The information transferred in the network is taken to match with the intrusion pattern, and then judge whether intrusion in the misuse detection. The anomaly detection can statistics the intrusion characteristic based on the known models.

3 Frame-Based Intrusion Detection Model

The cluster management idea is adopted in this paper and the MP2P network is divided into a number of regions. The mobile agent technology is also brought in to enhance the defense of intrusion detection system, by which the defects of static components are overcome.

The system framework includes the following sections: agent management module, intrusion detection system module and information capture module. The framework of intrusion detection is shown in Fig. 1.

The bottom layer of the intrusion detection framework in the MP2P network is information capture module. It is composed of the sniffer and 'honey pot'. The 'honey pot' is designed as a trap to track the invaders, thus, the invaders can be located and cracked down by defensive strike. The components of information capture module is show as fllows:

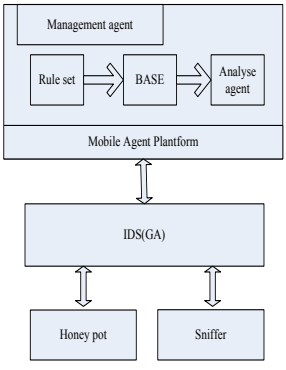


Fig. 1. The framework of intrusion detection

(1)Some nodes are used as 'honey pot' agents, who capture the attack information real-time, and the database model of network intrusion detection can be updated. Then according to the known regular pattern of the intrusion model, some suspicious behaviors of unknown intrusion can be detected by statistic method.

(2)The sniffers on the rest of agent nodes are used to capture the abnormal packets.

In generally, the intrusion detection module is realized in two steps: the first step is the misuse detection, and the second is the anomaly detection. Some of the nodes in every area are selected as cluster heads which can detect the intrusion in the global perspective.

When an abnormal event is monitored by the capture module, such as timing abnormal in communication, the abnormal data would be submitted to detect. The intrusion detection is executed through misuse detection and anomaly detection, and then it is confirmed that whether an intrusion event has happened. If the detection can detect the abnormal events, it would deal with them, or it transfers the relevant data to the agent management module to make further analysis and updates the database.

Agent management module as the kernel part, are responsible for the data analysis, database update and cluster domain partition. The database analysis module selects population from the database, predicts the attack characteristic by genetic algorithm and then the attack data would be transferred to the intrusion detection module through agent management module. The agent management module is mobile and can transfer data when the management module suffers devastating attacks, and the irreversible results can be avoided.

4 Intrusion Detection Based on Genetic Algorithm

The MP2P network should not only improve in the network structure, but also bring in new excellent intrusion detection algorithm[6] in order to deal with the

unknown intrusion better. The misuse detection just only matches the data in the limited intrusion database. However the efficient algorithm should have the ability of self-study and active defense. Therefore, firstly, the database should be updated by widely collecting suspect data. Secondly, the possible intrusion pattern can be detected based on the known information by the best search algorithm which can simulate the nature evolution, such as genetic algorithms.

4.1 The Choice of Intrusion Detection Algorithm

Game playing algorithms, Markov models and Genetic algorithms are the common methods in intrusion detection. After comparison, the genetic algorithm is superior to others in the aspect of getting the global optimal solution, and improving the detecting precision. The compared results is shown in table 1.

Table 1. Comparison of three common algorithms in intrusion detection

Intrusion detection algorithm	Detection mode	Search content	Conclusion
Game playing Algorithms	defense	source address, destination address, connection time, bytes sent num and status	The self-study and self-control is poor. In the case of artificial interference, it can make a good response to the intrusion. Resources occupation is reasonable
Markov Model	anomaly	source address, destination address, connection time, bytes sent num and status	Because of the limits of the algorithm, the false rate is very high. The self-study ability is low. But because the algorithm is simple, the resource overhead is low.
Improved genetic algorithm	anomaly, misuse	source address, destination address, connection time, bytes sent num and status	With high self-study ability and self-adaptability. When the detection accuracy is improved, the resources consumption would cost more.

4.2 Improved Genetic Algorithm

GA (Genetic Algorithm)[7] establishes the calculation model based on the principles of evolution and natural selection, which transforms the specific areas' problems to the similar data structure type of chromosome, thus the selection-reproduction, crossover, and mutation are used to calculate. The genetic algorithm is shown in Fig. 2.

In the flow diagram, i: stands for the number of generations; ai: stands for the individual fitness, A: stands for the threshold of fitness.

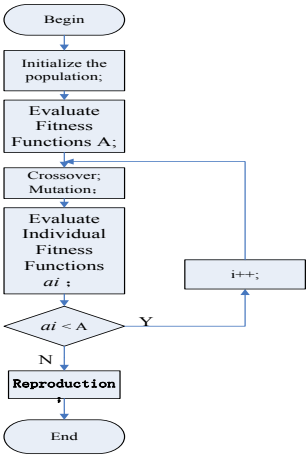


Fig. 2. The flow diagram of Genetic algorithm

Genes Coding. The concrete content have been transformed into genes in the paper, which can be network address of information flow, the network port, the length and number of packets, transmission time, protocol type, the connection attribution and so on. The selected attribution (genes) number maybe increased temporarily in order to improve the accuracy. In consideration of the content diversity and space occupied, binary floating-point coding is chosen to represent content. The attributions that are liable to detect the intrusion are chosen as genes, such as source address with high risk, the destination address, the connection time, transmit bytes, connection status and so on.

Population Initialization. In the initial population, the chromosomes, which are composed of the population, should be typical and cover the entire solution space, so it's easy to choose the optimal solution, and avoid the situation of local convergence. In this paper, the MP2P network is divided into several cluster domains for easily managing. Therefore, the corresponding chromosome should be analyzed synthetically. All the chromosomes in different zones form the population matrix, and the initial population can be selected from the original population matrix. The genes that constitute the chromosome can be showed by equation (1) :

$$M(t) = [m_1(t), m_2(t), \cdots, m_n(t)] \tag{1}$$

The eigenvector $m_i(t)$ is the gene attribution, which is at the position i of the population t, equation (1) is a row vector in the matrix. The chromosomes selected in the same region form a population matrix. In this matrix, the similarity of each pair chromosomes is contrasted by the hamming distance analysis method:

$$d_q = |m_{p\times q} - m_{l\times q}| \tag{2}$$

In the equation (2), the $p \times q$ and $l \times q$ is the p-th row, the q-th colum and the l-th row, the q-th colum, that is to compare the genes in the same column which corresponds different rows. When $d_q < D_q$, it means the genetic diversity degree a is 0, otherwise a is 1(D_q is similarity threshold which is set according to different genes). The final value is accumulated by counter A automatically:$A = \sum(a)$.

After the two vectors compared, if $A > N \times 90\%$ (N is the number of gene in the chromosome encoding), then the two chromosomes is consider very similar.

Crossover, Mutation and Reproduction. When the initial population has been finalized, the three operators of crossover, mutation, and reproduction should be optimized continually, in order to achieve the global optimal solution.

(1) Crossover algorithm: Multi- chromosome multi-point cross Crossover is an exchange process from the parent chromosomes, in order to re-form a new generation for improving the search capability. However, the traditional restructuring of crossover algorithm is executed in two chromosomes; the evolution of subspace is limited by just simple reorganization. Therefore, Multi-chromosome multi-point cross as a new improving method is adopted to expand the searching space. In consideration of the increased diversity and the computing overhead brought in by multi-point loop cross, this paper chooses three chromosomes as the parent generation ultimately, and two gene fragments are chose to cross randomly. The operation is shown in Fig. 3.

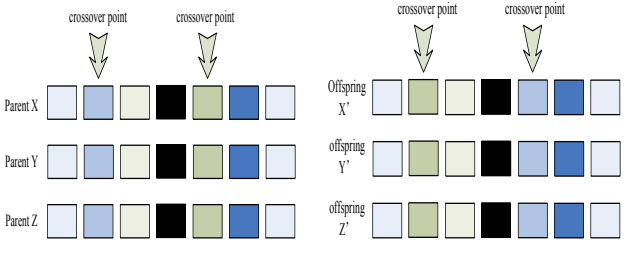


Fig. 3. Crossover algorithm

Thus, the probability of getting optimal solution could be increased with the extending of searching space.

(2) mutation algorithm: Poisson distribution

Mutation operator references to the phenomenon of gene mutation which is generated in the evolution process of the biology, in order to generate some new individuals to expand the search space.

Poisson distribution algorithm is adopted in this paper to determine the polymorphic, the genes with more frequently abnormal could be found by Poisson distribution which appear in a certain period: $P(x = k) = \frac{e^(-\lambda) \cdot \lambda \cdot k}{k!}$.

In which, K is the abnormal times in a certain period and in the same cluster.

Then sort the result in descending order according to the probability. The gene with the highest abnormal rate is selected to mutate. Improved mutation operator changes the polymorphic according to different time periods, which is different from the traditional random mutation. The search space is defined more clearly and the unnecessary waste of resources is controlled. The probability of mutation is controlled between $0.0001 - 0.1$.

(3) Copy-preferred protection
In each new generation of chromosomes, it should calculate the individual fitness of the chromosomes. When the fitness of individual reaches the certain threshold, the individual could be directly copied to the next generation without the operation of mutation. It is a protection to the chromosomes which have good genes, and the calculation of time and space also could be reduced, so as to fully use of the resources.

(4) Fitness function Each new generation of chromosomes should be filtered, the excellent individual will be incorporated into the next populations continue to do optimization calculation. And this screening scale is decided by fitness. The individual with higher fitness is chosen as the alternative, and the lower is directly eliminated. Therefore, the choice of fitness directly affects the accuracy of the algorithm and the global convergence of the situation. Fitness function: $F = \frac{a}{A} - \frac{b}{B}$.

In the above function, F represents the value of fitness function,; a is the number of detected attacks, A is the total number of attacks; b is the wrong number of attack detection, B is the total number of errors detected[8].

5 Experiment

In this paper, VC++ language is adopted to achieve the above proposal, and the experiment is based on the network connection data set of KDDCUP99[9]. Because of KDDCUP99 contains a lot of simulated attack records, part of the intrusion records are adopted by this paper as the basic experimental data. Meanwhile, the improved genetic algorithm is adopted by the intrusion detection of MP2P network, the results from real-time intrusion detection is shown in Fig.4, in which the horizontal axis represents the iterations number of genetic algorithm, vertical axis indicates the percentage, the blue line represents the detection rate, and the false alarm rate is expressed by the red line. Research results verify that, the detection rate could be improved and false alarm rate could be reduced as the iterations number of genetic algorithm increases when the structure of the MP2P network adopts cluster management and the improved genetic algorithm is chosen as a means of intrusion detection.

5.1 Results and Analysis

The improved genetic algorithm is adopted by this paper in the MP2P network, in order to improve the searching capabilities of intrusion detection. For the

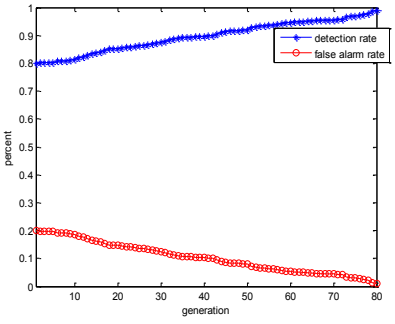


Fig. 4. Detection rate and false alarm rate

problem of poor self-updating, low accuracy rate, and other issues in intrusion detection, this paper re-plans the architecture of MP2P network. Firstly, improve the information management and searching. Secondly, the three operators of genetic algorithm are optimized to do better in searching abnormal of MP2P network, thus the searching space and the global optimal solution is improved, the limited computing and storage resources are not excessively wasted. Research results verify that, the propose of intrusion detection scheme in this paper could reduce the false alarm rate, and is effective in strengthening the capability of intrusion detection for the MP2P network, thereby the robustness of MP2P network is enhanced.

References

1. Ou, Z., Song, M., et al.: Key techniques for Mobile Peer-to-Peer Networks. Journal of Software 19(2), 404–418 (2008)
2. Cheng, J.J., Li, Y.H., Cheng, S.D., et al.: The architecture on the mobile P2P system and the study for the key technology. Journal of Beijing University of Posts and Telecommunications 29(4), 86–89 (2006)
3. Zhu, X., Chen, D., et al.: Pareto Multi objective Genetic Algorithm with Multiple-Chromosomes Crossover. Acta Electronica Sinica 29(1), 106–109 (2001)
4. Qing, H., Jiang, J., et al.: Research on intrusion detection techniques. Journal of China Institute of Communications 25(7), 19–29 (2004)
5. Cabrera, J.B.D., Gutierrez, C., Mehra, R.K.: Ensemble method for anomaly detection and distributed intrusion detection in mobile. Ad Hoc Networks 9(1), 96–119 (2008)
6. Tang, Z., et al.: The design and implementation of network intrusion detection, pp. 2–10. Electronic Industry Press (2002)
7. Holland, J.H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence, pp. 20–35. MIT Press, Cambridge (1992)
8. Pillai, M.M., et al.: An approach to implement a network intrusion detection system using genetic algorithms. In: Proceedings of the Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists (2004)
9. KDD Cup Data (2006), http://kdd.ics.uci.edu/

Mining the Role-Oriented Process Models Based on Genetic Algorithm

Weidong Zhao, Qinhe Lin, Yue Shi, and Xiaochun Fang

Software School, Fudan University, Shanghai,
200433 China
{wdzhao,10212010014,10212010024,052053033}@fudan.edu.cn

Abstract. Traditional role-oriented process modeling seems to be subjective in identifying roles. To solve the problem, the similarity of activities is used in this paper. Sub-processes with high similarity are recognized as the process undertaken by a certain role. In this way, a relatively objective role identification approach is proposed, which determines the interaction between roles and establishes the role-activity diagram. Furthermore, by analyzing the interaction between roles, genetic algorithm is used to introduce multiple factors to optimize the identification. Therefore, an optimized role-oriented process modeling approach is established and an example is presented to show the feasibility of this approach.

Keywords: Process Mining, Role-Activity Diagram, Role Identification, Genetic Algorithm.

1 Introduction

Business process management has continued to attract attentions of both academics and industry. Process modeling is fundamental to BPM. Generally, there are mainly two kinds of modeling methods: activity-based and role-oriented process modeling. In the activity-based process model, a business process consists of activities and their dependence [1,2]. But there are some problems when describing the relation between process roles and other elements. Thus, some researchers proposed the role-oriented process modeling, which highlights process participants and their interactions.

Role-oriented workflow models are supposed to be extended from activity-based workflow models. The social network analysis proposed in [3] has drawn more attention in the domain of process mining. As participants are the basic elements of organization structure, some researchers use the participant interactions to evaluate their social relations and identify the social network among them [4,5]. In role engineering, roles are identified from permission assignments because participants have the corresponding privilege when they play certain roles [7]. Jurgen performs clustering analysis of permission assignments to build a hierarchy of permission clusters and finally the role hierarchy [8]. RolEnact describes processes using roles, states, activities and events [9]. Role activity diagram (RAD) is another typical role-oriented process modeling approach, which

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 398–405, 2012.

uses sub-processes to describe process roles' responsibility and highlights the interaction between them. In role engineering, the bottom-up aggregation method identifies role hierarchy by activity similarity among participants. But as each clustering step randomly selects sub-roles, the result is uncertain.

The approaches discussed above analyze the relation between roles from different aspects, but lack of objective role identification. In this paper, a more objective role identification approach is proposed. It identifies roles by regarding the activities with high similarity as a sub-process corresponding to a certain role. Then we determine the interaction between roles by participants' interaction and generate the role-activity diagram. Furthermore, as participants who play the same role should have similar interaction with other roles, we use genetic algorithm to combine these two factors and finally develop a role-oriented process modeling approach based on process mining.

The remainder of this paper is organized as follows: section 2 describes the concepts of activity dependence. Section 3 presents two steps of the basic modeling. Section 4 shows how genetic algorithm can be used to optimize role identification. Section 5 draws conclusions.

2 Basic Concepts

Dependence between activities can be determined by analyzing process logs. In this paper, we use definitions in [10] to describe the dependence between activities. The set of all activities in process logs is denoted as $A = \{a_1, a_2, \ldots, a_m\}$, where $m = |A|$ is the number of activities contained in the log. The set of all participants in workflow logs is denoted as $P = \{p_1, p_2, \ldots, p_n\}$, where n is the number of participants. I is the set of all process instances, and for any instance $i \in I, i = (a_1, a_2, \ldots, a_k), a_i \in A$, where i records the execution ordering of activities in the instance. The activity set of instance i is denoted as $AS\,(i)$.

Definition 1. *Dependence* $a > b \Leftrightarrow \exists i = (a_1, a_2, ..., a_k), i \in I, u \in \{1, 2, ..., k-1\}, a, b \in A$, *where* $a_u = a, a_{u+1} = b$.

Definition 2. *Direct dependence* $a \rightarrow b \Leftrightarrow a > b$, *and* $\forall i \in I, \neg\exists b > a$.

Definition 3. *Activity set with direct dependence*

$$DEP = \{(a, b) | a, b \in A, a \rightarrow b\}$$

Definition 4. *Adjacent activity set*

$$adjAS(A_v, S) = \{a | a, b \in A_v, a \notin S, b \in S, a > b \text{ or } b > a, S \subseteq A_v\}$$

3 Role-Activity Diagram Modeling

Process logs contain a lot of information about participants and their activities. In this paper, we establish a more objective and reliable role identification approach in terms of the similarity of activities undertaken by participants as the measurements for role identification using process mining.

3.1 Diversity Degree of Activity Set

Definitions in [10] are used here to illustrate the diversity degree of sub-processes. Higher diversity degree means more difference between participants in responsibilities, while lower diversity degree implies more similarity between participants.

Definition 5. *Activity execution frequency of a participant* p, $R(p,a) = \frac{times(p,a)}{times(p)}$, *where* $times(p,a)$ *is the times* p *executes* a *and* $times(p)$ *is the times* p *executes all activities.*

Definition 6. *Activity execution vector*
$S(p, A_u) = (R(p,a_1), R(p,a_2), ..., R(p,a_k)), a_j \in A_u, 1 \leq j \leq k$, *where* k *is the total number of activities in* A_u.

Definition 7. *Average execution vector of activity set*

$$AG(A_u) = \frac{S(p_1, A_u) + S(p_2, A_u) + ... + S(p_m, A_u)}{|PS(A_u)|}, p_i \in PS(A_u), 1 \leq i \leq m.$$

$m = |PS(A_u)|$, *where* $PS(A_u)$ *is the set of all participants who execute activities in* A_u.

Definition 8. *Diversity degree of activity set*

$$D(A_u) = \frac{\sum_{i=1}^{m} |S(p_i, A_u) - AG(A_u)|}{|PS(A_u)|}, p_i \in PS(A_u), m = |PS(A_u)|$$

3.2 Generation of Role-Oriented Process Model

There are two steps in this phase. Firstly, identify roles based on the concept of diversity degree of activity set. Then, determine the interaction between roles according to the dependence between participants.

(1) Role Identification
In this paper, the sub-process whose diversity degree is less than a threshold is considered as the process corresponding to a certain role. We use the similar algorithm as "Mining activity set taken by each role" in [10] to discover the activity sets in which activities are executed by participants playing the same role. Given a threshold TH, the algorithm first finds a remainder activity which has dependence relationship with the current activity set and then combines them into a new activity set until the diversity degree is larger than TH. The process designer may set a reasonable TH value according to their experience: The larger TH is set, the fewer roles will be generated. Note that TH should be higher than the minimum diversity degree of single-activity sets. Therefore, a new role corresponding to the combined activity set is generated.

(2) Role Interaction mining
Sub-processes can be determined by identifying roles and their corresponding activity sets, while interactions between roles can be determined by the relationship

between participants. In this paper, we determine interactions between roles by establishing a social network of participants using similar approach proposed in [7]. Here we use definitions in [10] to identify interactions between participants.

Definition 9. *Dependence activity pairs between participants*
$rel(p_1, p_2) = \{(a, b) | i \in I, i = (a_1, a_2, ..., a, b, ..., a_k), p_1, p_2 \in P, p_1$ executes a, p_2 executes b and $a \rightarrow b\}$

Then we can generate the activity set that connected related roles, which demonstrates the interaction between roles.

4 Role Identification Optimization Based on Genetic Algorithm

In section 3, the role is identified according to participants with similar activities. But this approach performs unsatisfactorily in determining the role interaction because it does not take the participant interaction into account when roles are identified. Based on process mining, through the analysis of the interaction between participants, we consider both the activity diversity and the interaction diversity degree. This idea comes from the fact that the participants playing the same role should have similar interaction. By using genetic algorithm to combine these two factors, an optimized role identification approach is proposed.

4.1 Activity Interaction between Participants

Definition 10. *The activity dependence occurrence times*

$$Exi(p_1, p_2, a) = \sum_{u=1}^{t} Exi(p_1, p_2, a, i_u), where$$

(1) $p_1, p_2 \in P$;
(2) $i_u \in I, i_u = (a_1, a_2, ..., a_s), \exists k \in \{1, 2, ..., s-1\}, a_k = a, (a_k, a_{k+1}) \in rel(p_1, p_2) \cup rel(p_2, p_1)$;
(3) t is the number of instance i_u in I which satisfies (2);
(4) $Exi(p_1, p_2, a, i_u)$ is the occurrence of a dependence activity pair between p_1 and p_2 in i_u.

In (2), $rel(p_1, p_2) \cup rel(p_2, p_1)$ means that when a modeler considering the dependence between two participants, both the two direction dependence should be considered, i.e. p_1 depends on p_2, or p_2 depends on p_1. $Exi(p_1, p_2, a, i_u)$ describes the activity dependence occurrence times. For example, in the instance $i_1 = (a, b, c, b, c, e)$, b executed by p_1 appears twice, c is executed by p_2 and $b \rightarrow c$, then $Exi(p_1, p_2, b, i_1)$ equals 2.

Definition 11. *Activity interaction vector between participants*

$$R_1(p_1, p_2) = (Exi(p_1, p_2, a_1), Exi(p_1, p_2, a_2), ..., Exi(p_1, p_2, a_n)), n = |A|$$

The activity interaction vector describes the activity interaction between p_1 and p_2. Assume the activity set is $\{a_1, a_2, a_3\}$ and $R_1(p_1, p_2) = (2, 3, 0)$. In this case, a_1 appears twice, a_2 appears three times in the interaction between p_1 and p_2.

Definition 12. *Activity interaction vector between participants and roles*

$$R_2(p, r) = \sum_{i=1}^{k} R_1(p_i, p)$$

where p_i plays r, and k is the total number of participants who play r.

Similarly, the activity interaction vector between roles is defined as the sum of all interaction vectors between one role and participants who play other roles. Thus, the average activity interaction vector between r_2 and r_1 is denoted as $R_3(r_1, r_2) = \frac{\sum_{i=1}^{l} R_2(p_i, r_2)}{l}$, where p_i plays r_1 and l is the total number of participants who play r_1.

Definition 13. *Activity interaction diversity degree of role*

$$R_4(r) = \frac{\sum_{i=1}^{k} \sum_{j=1}^{m} \left| \frac{R_2(p_i, r_j)}{|R_2(p_i, r_j)|} - \frac{R_3(r, r_j)}{|R_3(r, r_j)|} \right|}{k \times m}$$

where $r_j \neq r$, k is the total number of participants who play r, and m is the number of all roles except r itself.

Lower activity interaction diversity degree means more similarity between participants, while larger degree indicates more difference.

4.2 Role Identification Optimization

In this section, the main steps of genetic algorithm to optimize role identification are analyzed.

A typical genetic algorithm requires a suitable representation of solution domain. Herein, the upper triangular matrix represents the following: If p_i and p_j play the same role, then M_{ij} equals 1, otherwise 0; If $i = j$ and $M_{ij} = 1$, then p_i plays the role. A representation is generated by combining all the row bit strings in sequence.

The fitness function is defined as $f(R) = (1 - \lambda) \sum_{i=1}^{q} R_4(r_i) + \lambda \sum_{i=1}^{q} D(AC(r_i))$ where R is the set of roles, q is the number of roles, and $AC(r_i)$ is the activity set which role r_i is in charge of. The function $f(R)$ consists of two parts: the first part describes the diversity degree of participant activity execution and the second part means the diversity degree of interactions between participants. λ is a weight parameter of $f(R)$. The activity execution similarity of participants who play the same role and the interaction similarity of them with other roles are both considered, thus $f(R)$ is a more comprehensive role identification measure.

(1)Selection: The population is sorted by descending fitness values and each generation selects the constant number of best individuals.

(2)Crossover: a genetic operator is used to vary the coding of a chromosome or chromosomes from one generation to the next. The probability of two parents mixing is called crossover probability. As the initial solution set is large in the paperthe crossover probability p_c is defined as $p_c = \frac{C_1(f_{\max} - f_m)}{f_{\max} - f_{avg}} + C_2$,where C_1, C_2 are constants, $f_{\max}$ is the maximum fitness value of current solutions set while f_{avg} is the average fitness value and f_m is a large fitness value in the current solution. The solution with lower fitness value has a higher crossover probability while the solution with higher fitness value has a lower crossover probability. Thus, it can speed up genetic algorithm.

(3)Mutation: the mutation probability of each solution is defined as $1 - (1 - p_m)^L$ where p_m is the crossover probability and L is the size of the solution set. When the mutation conditions are satisfied, one bit is changed from the previous solution, making an NOT operation at a specified position. The mutation probability p_m is defined as $p_m = \frac{C_3(f_{\max} - f_m)}{f_{\max} - f_{avg}} + C_4$, where C_3 and C_4 are constants. The purpose of mutation is to preserve and introduce the diversity. With the crossover probability defined in (2), this mutation probability can accelerate the search.

5 Case Study

The matrix R generated from a clothing production process log subsumes times of the activities performed by the participants and denotes the percentage of the activities executed by each participant.

$$R = \begin{array}{c} \\ p_{101} \\ p_{105} \\ p_{106} \\ p_{107} \\ p_{114} \\ p_{115} \\ p_{120} \end{array} \begin{array}{c} \begin{array}{cccccccc} a & b & c & d & e & f & g & h \end{array} \\ \begin{pmatrix} 0.5 & 0.3 & 0 & 0 & 0 & 0 & 0 & 0.2 \\ 0 & 0 & 0.154 & 0.385 & 0.461 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0.4 & 0.35 & 0 & 0 & 0 & 0 & 0 & 0.25 \\ 0 & 0 & 0.1 & 0.4 & 0.3 & 0.1 & 0.1 & 0 \\ 0 & 0.185 & 0.111 & 0.185 & 0.148 & 0 & 0.037 & 0.334 \\ 0 & 0 & 0.385 & 0 & 0 & 0.615 & 0 & 0 \end{pmatrix} \end{array}$$

(1) Role identification

To identify roles, assume $TH = 0.25$ and the identification result is shown here. p_{101}, p_{107} and p_{115} are planning dept. managers, p_{101}, p_{107} and p_{115} are deliverymen, p_{105}, p_{114}, p_{115} and p_{120} are technicians, p_{106}, p_{114} and p_{115} are workers.

(2)Role-oriented process modeling

Fig. 1 shows the mined role-activity diagram and the text between roles describes the interaction between roles.

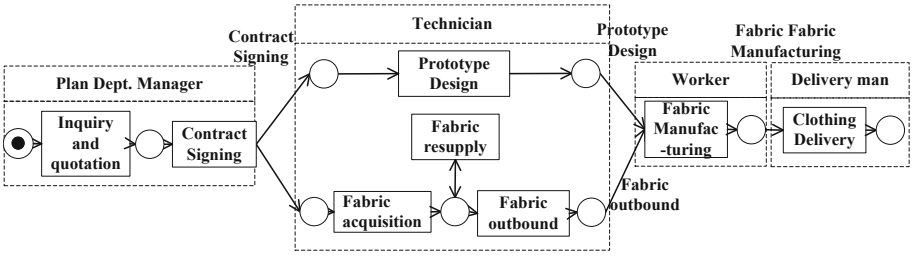


Fig. 1. The role-activity diagram of clothing production process

(3) Role identification optimization using genetic algorithm
The matrix M_r shows the role identification results using genetic algorithm, in which '-' implies no value. The representation is generated by combining all the row bit strings in sequence, i.e. 00010100001011000000010001000.

$$M_r = \begin{array}{c} \\ p_{101} \\ p_{105} \\ p_{106} \\ p_{107} \\ p_{114} \\ p_{115} \\ p_{120} \end{array} \begin{array}{c} \begin{array}{ccccccc} p_{101} & p_{105} & p_{106} & p_{107} & p_{114} & p_{115} & p_{120} \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ - & 0 & 0 & 0 & 1 & 0 & 1 \\ - & - & 1 & 0 & 0 & 0 & 0 \\ - & - & - & 0 & 0 & 1 & 0 \\ - & - & - & - & 0 & 0 & 1 \\ - & - & - & - & - & 0 & 0 \\ - & - & - & - & - & - & 0 \end{pmatrix} \end{array}$$

Table 1. Genetic Algorithm - Participants and their Roles

Participant	Role
p_{101}	Panning Dept. Manager
p_{105}	Technician
p_{106}	worker
p_{107}	Panning Dept. Manager
p_{114}	Technician
p_{115}	Panning Dept. Manager
p_{120}	Technician

Table 1 expresses the participants and their corresponding roles. Each participant is related to only one role and the interaction between participants are simpler. For example, p_{115} plays three roles in the first result, but only one role in Table 1. Furthermore, the activities which one role is in charge of can be disconnected. For example, p_{101} , p_{107} and p_{115} are Planning Dept. Managers and the corresponding activity set is $\{a, b, h\}$ where activity a, b and h are not connected. In contrast to this, the general role identification approach will regard the activity h as a single role. By comparison, it seems that the genetic algorithm can get a better and more objective result.

6 Conclusions

In this paper, we present a role identification approach by analyzing the similarity of activities undertaken by participants based on process logs. On this basis, we use the interaction between participants to further determine role interactions, which allows us to mine the role activity diagram. Furthermore, we use information contained in process logs to determine the interaction consistency between participants and other roles. Also, genetic algorithm is used to achieve a more objective role identification approach. However, how to determine a reasonable threshold TH, how to achieve more reasonable role identification and how to deal with the complexity of process structures need further researches.

Acknowledgement. This research is supported by the Natural Science Foundation of China (No. 71071038).

References

1. Johnathan, E.C., Alexander, L.W.: Discovering Models of Software Processes from Event-Based Data. ACM Transactions on Software Engineering and Methodology 7, 215–249 (1998)
2. van der Aalst, W.M.P., Weijters, A.J.M.M.: Process mining a research agenda. Computers in Industry 53, 231–244 (2003)
3. Wasserman, S., Faust, K.: Social Network Analysis: Methods and Applications. Cambridge University Press, Cambridge (1994)
4. van der Aalst, W.M.P., Song, M.: Mining Social Networks: Uncovering Interaction Patterns in Business Processes. In: Desel, J., Pernici, B., Weske, M. (eds.) BPM 2004. LNCS, vol. 3080, pp. 244–260. Springer, Heidelberg (2004)
5. van der Aalst, W.M.P., Reijers, H.A., Song, M.: Discovering Social Networks from Event Logs. Computer Supported Cooperative Work 14, 549–593 (2005)
6. Kuhlmann, M., Shohat, D., Schimpf, G.: Role mining - revealing business roles for security administration using data mining technology. In: Proceedings of the 8th ACM Symposium on Access Control Models and Technologies, pp. 179–186. IEEE Press, New York (2003)
7. Schlegelmilch, J., Steffens, U.: Role mining with ORCA. In: Proceedings of the Tenth ACM Symposium on Access Control Models and Technologies, pp. 168–176. IEEE Press, New York (2005)
8. Phalp, K.T., Henderson, P., Walters, R.J., Abeysinghe, G.: RolEnact: role-based enactable models of business processes. Information and Software Technology 40(3), 123–133 (1998)
9. Zhao, W., Dai, W., Wang, A., Fang, X.: Role-activity Diagrams Modeling Based on Workflow Mining. In: 2009 World Congress on Computer Science and Information Engineering, pp. 301–305. IEEE Press, New York (2009)
10. van der Aalst, W.M.P., Weijters, T., Maruster, L.: Workflow Mining: Discovering Process Models from Event Logs. IEEE Transactions on Knowledge and Data Engineering 16(9), 1128–1142 (2004)

Image Retrieval Based on GA Integrated Color Vector Quantization and Curvelet Transform

Yungang Zhang[1,2], Tianwei Xu[1], and Wei Gao[3]

[1] School of Information Science, Yunnan Normal University, Kunming 650092, China
[2] Department of Computer Science & Software Engineering, Xi'anJiaoTong-Liverpool University, Suzhou 215123, China
[3] Department of Mathematics, Soochow Univeristy, Suzhou 215006, China
yungang.zhang@liv.ac.uk

Abstract. Color and shape information have been two important image descriptors in Content Based Image Retrieval (CBIR) systems. The focus of this research is to find a method representing images with color and shape information in the way of human visual perception. The image retrieval approach proposed here depends on the color and shape features extracted by color Vector Quantization (VQ) and the Digital Curvelet Transform (DCT), respectively. The extracted color and shape features were combined and weighted by Genetic Algorithm (GA), then used for image similarity measurement. Experimental results show that the GA combined features can bring about improved image retrieval performance.

Keywords: Image retrieval, color vector quantization, curvelet transform, genetic algorithm.

1 Introduction

As one of the most important applications of image analysis and understanding, CBIR (Content-Based Image Retrieval) has received more and more attention. The tremendous growth of the quantities and sizes of digital image and video require powerful tools for searching in image and video databases. Since a lot of image databases are poorly indexed or annotated, there is a great need for developing automated, content-based systems to help users to get images they want.

There have been a large number of CBIR systems developed in the recent years such as IBM's QBIC project [5], VisualSeek [15], PicSOM [13], PicToSeek [7] and a lot more. When facing with a query, the system extracts features from the query, compares them to that of the images stored in the database, the distance between the query image and each image in database is evaluated according to the similarity of features. Sometimes the searching result can be quite different from user's expectation because of the 'semantic gap', the main reason

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 406–413, 2012.

of semantic gap is the extracted visual features mismatch human's judgements on similarity. The focus of our research[1] is mining image features which can represent images in the way of human visual perception.

In human visual perception system, humans use a combination of features (color and shape) to recognize objects and do not rely on any one individual feature [10]. In our research, the color vector quantization is selected for image color feature representation; digital curvelet transform is used to extract shape information in images. Genetic Algorithm was then used to optimize weights for all the curvelet and color features of each image category. The combined and weighted features were used for similarity measurement. Experimental results show that the combined and weighted features can bring about good retrieval performance in terms of precision and recall.

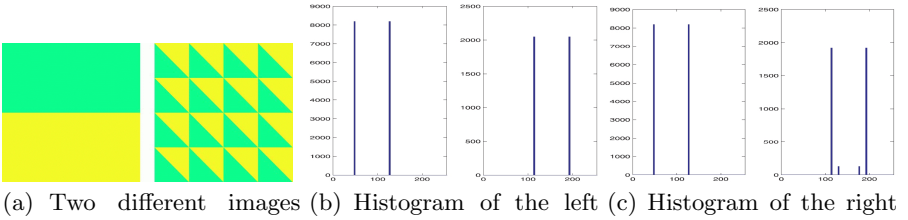
2 Image Feature Extraction by Vector Quantization

2.1 Color Vector Quantization

When processing color data in images, one is always faced with the problem that color information on the one hand needs to be quantized as compactly as possible and on the other hand must be represented with sufficient accuracy.

Vector Quantization can take into account the actual color distributions for quantization by exploiting training images from the database. That is, a set of representative color components from training images can be determined and each representative color component serves as a color histogram bin [17,11].

Although there are various color scalar quantization (SQ) methods, they have apparent drawbacks when using them in image retrieval work. Since they do not consider the spatial relationship between pixels, as we can see from Fig.1 , VQ can provide a way of better exploiting the spatial information to generate different histograms in such case.



(a) Two different images have same numbers of colors

(b) Histogram of the left image. Left: SQ. Right: VQ

(c) Histogram of the right image. Left: SQ. Right: VQ

Fig. 1. Comparison of histograms generated by SQ and VQ

[1] This work is supported by the Key Laboratory of Ethnologic Education Informatization and Chunhui Planning Grant Z2009-1-65001 of Ministry of Education, China.

2.2 Color Spatial Pattern Codebook Generation and Training

G.Qiu has developed a Colored Pattern Appearance Model ($CPAM$)[8], which has two channels capturing the characteristics of the chromatic and achromatic spatial patterns. This model is working in YC_bC_r color space. In $CPAM$, the visual appearance of a small image block is modeled by three components: the stimulus strength (S), the spatial pattern (P) and the color pattern (C).

The P channel of $CPAM$ captures the achromatic spatial pattern of the input colored image pattern. The C channel captures the chromatic spatial pattern. The P and C channels are called Achromatic Spatial Pattern (ASP) and Chromatic Spatial Pattern (CSP), respectively.

A specific neural network training algorithm, the Frequency Sensitive Competitive Learning(FSCL) algorithm[1] is used here to design our codebook. According to [9], FSCL is insensitive to the initial choice of codewords, and the codewords designed by FSCL are more efficiently utilized than those designed by methods such as the LGB algorithm. The FSCL VQ design algorithm is:

1. Initialize the codewords, $C_i(0), i = 1, 2, \ldots, I$, to random numbers and set counters associated with each codeword to 1, i.e., $n_i(0) = 1$.
2. Present the training sample, $X(t)$, where t is the sequence index, and calculate the distance between $X(t)$ and the codewords, $D_i(t) = D(X(t), C_i(t)$, and modify the distance according to $\hat{D}_i(t) = n_i(t)D_i(t)$.
3. Find j, such that $\hat{D}_j(t) \leq \hat{D}_i(t)$ for all i, update the codeword and counter
   $C_j(t+1) = C_j(t) + a\big[X(t) - C_j(t)\big]$
   $n_j(t+1) = n_j(t) + 1$
   where $0 < a < 1$ is the training rate.
4. Repeat by going to 2.

3 Curvelet Transform and Image Spatial Feature Extraction

3.1 The Discrete Curvelet Transform(DCT)

Curvelet Transform [6] is one of the latest developments in non-adaptive transforms. Compared to the wavelet transform, the curvelet transform provides a more sparse representation of an image, with improved directional elements and better ability to represent edges and other singularities along curves. Sparse representation usually offers better performance with its capacity for efficient signal modeling.

In the curvelet transform, fine-scale basis functions are long ridges; the shape of the basis functions at scale j is 2^{-j} by $2^{-j/2}$ so the fine-scale bases are skinny ridges with a precisely determined orientation. The curvelet coefficients can be expressed by

$$c(j,l,k) := \langle f, \varphi_{j,l,k} \rangle = \int_{R^2} f(x)\varphi_{j,l,k}(x)dx \tag{1}$$

where $\varphi_{j,l,k}$ denotes the curvelet function, and j, l and k are the variables of scale, orientation, and position, respectively.

In the last few years, several discrete curvelet transforms have been proposed. The most influential approach is based on the Fast Fourier Transform (FFT) [3]. The DCT decomposes the frequency space into dyadic rectangular coronae, each of which is divided into wedges, the number of wedges doubles with every second level.

3.2 Spatial Feature Extraction through Curvelet Transform

Once the curvelet coefficients have been obtained from DCT, the standard deviation of the curvelet sub-bands is computed as the shape features for the curvelet, since this feature has shown good capability in description of wavelet and curvelet sub-bands [2,16].

In consideration of computational complexity, not all levels of curvelet coefficients are used. Only level 2 and level 5 sub-bands coefficients are selected for feature extraction in a 6 levels decomposition of images. The coefficients at level 5 are in 'finer' details of an image, they can be used as the texture description of the image, and coefficients at level 2, on the contrary, in a 'coarser' level, they are suitable for describing the edges in an image. Some statistics can be calculated from each of these curvelet sub-bands. The mean and standard deviation of the first half of the total subbands for level 2 and level 5 were calculated since the curvelet at angle θ produces the same coefficients as the curvelet at angle $(\theta + \pi)$ in the frequency domain. These subbands are symmetric in nature.

The total features for each image I then can be described as:

$$F(I) = \left\{\mu_1^I, \sigma_1^I, \mu_2^I, \sigma_2^I, C^I\right\}, \tag{2}$$

where μ_1 and μ_2 are mean values of level-2 and level-5 curvelet coefficients; σ_1 and σ_2 are standard deviation of level-2 and level-5 curvelet coefficients; C^I is the color histogram. According to curvelet transform, there are 16 sub-bands and 32 sub-bands in level-2 and level-5 respectively, thus, the dimensionalitie of μ_1 and σ_1 are 8, the dimensionalitie for μ_2 and σ_2 are 16, and the dimensionality of C^I is 512 as we have explained in Section 2.

4 Genetic Algorithm Based Weights Assignment for Image Categories

4.1 In-class Distance and Between-class Distance

For an image category K in the image database T, the *in-class* distance ICD is defined as:

$$ICD_K = \{D_{i,j} | D_{i,j} = EuDis(K_i, Kj), i \neq j\} \tag{3}$$

where $EuDis(\cdot, \cdot)$ is the weighted Euclidean distance between the five features(see equation 2) of two images K_i and K_j, which are belong to the same image category K. the $EuDis$ of K_i and K_j is calculated as

$$EuDis(K_i, Kj) = \sqrt{\sum_{l=1}^{5}(w_l \times F_l(K_i) - w_l \times F_l(K_j))^2} \tag{4}$$

where $F_l(\cdot)$ is the lth feature of an image, and w_l is the weight assigned to $F_l(\cdot)$.

Thus, the in-class distance of an image category K is a set of distances between any two images K_i and K_j in the same category. Note that $D_{i,j} = D_{j,i}$.

For an image category K in the image database T, the *between-class* distance BCD is defined as:

$$BCD_K = \{D(K_i, R_i^c) | D(K_i, R_i^c) = EuDis(K_i, R_i^c)\} \tag{5}$$

where R_i^c represents an image R_i which belongs to any image category $c, c \neq K$ in the image database T. If we totally have C image categories and each category has M images, then, the number of elements in a BCD set is $(C-1) \times M$. For example, if we have 10 image categories and each category has 60 images, then the number of elements in a BCD set is 540.

4.2 GA-Based Weights Assignment

A chromosome in our GA is defined as:

$$c = (w_1, w_2, \ldots, w_i, \ldots, w_n), \tag{6}$$

where w_i is the weight assigned to feature vector i and n is the dimensionality of the image feature vector set, which is 5 in our current implementation. A population P is defined as:

$$P = \{c_1, c_2, \ldots, ci, \ldots, c_{PopSize}\} \tag{7}$$

where $PopSize$ is the number of individuals in the population and c_i is a chromosome. In our work, the $PopSize = 100$. The selection method, Roulette wheel, proposed by Holland[12] was used. The simple crossover and uniform mutation were employed in our implementation[4]. The crossover and mutation probabilities are 0.75 and 0.05, respectively.

Given an image category K with a set of weights w, if the in-class distance set ICD_K has M elements and the between-class distance set BCD_K has N elements, then the fitness function sorts all the distance elements from both ICD_K and BCD_K, counting the smallest N_1 elements which come from the in-class set ICD_K, we wish to let $N_1 \to N$ on the selection of proper w.

After GA weights training, each image category $K, K = 1, 2, \ldots, 10$ in the image database can obtain a proper weight vector $\left\{w_i^K\right\}, i = 1, \ldots, 5$ as listed in Table 1.

4.3 Automatic Weight Selection

The mean values of features for training images in each image category were firstly defined as:

$$\overline{F}_K = \left\{\overline{f}_i\right\} \tag{8}$$

where $i = 1, \ldots, 5$ and $\overline{f}_i$ is the mean value of the ith feature of all the training images belong to category K. When a query image Q comes in, the features of Q were extracted first, form a feature vector $F(Q) = \{fq_i\}, i = 1, \ldots, 5$, then, use the following strategy to decide which weight should Q use.

Table 1. Weights obtained by GA

Image Category	w_1	w_2	w_3	w_4	w_5
1	0.0777	0.06967	0.0891	0.0551	0.7514
2	0.8895	0.2420	0.4849	0.8722	0.4156
3	0.1323	0.0277	0.9768	0.8184	0.4156
4	0.2578	0.3026	0.8575	0.4278	0.6938
5	0.4518	0.7859	0.9949	0.1289	0.9108
6	0.9172	0.2591	0.9363	0.0369	0.5882
7	0.1174	0.7958	0.9783	0.8305	0.1062
8	0.4067	0.1181	0.9989	0.3282	0.5810
9	0.5104	0.5978	0.6298	0.1858	0.4478
10	0.5508	0.3120	0.1137	0.2070	0.5160

1. For each weight vector, $W_K(K = 1, 2, \ldots, 10)$(Table. 1), calculate $F_{QK} = \left\{ fq_i \times w_i^K \right\}$, $i = 1, \ldots, 5$.
2. For the mean feature vector of category $K, K = 1, 2, \ldots, 10, \overline{F}_K$(Equation 8), calculate its Euclidean Distance with F_{QK}, find out the minimum distance and the corresponding category R.
3. Use a uniform weight vector like $\{1, 1, 1, 1, 1\}$ to do things in step 1 and 2 again, finding another minimum distance and its corresponding category $R1$.
4. If $R = R1$, the weights assigned to the query image Q is W_R. Otherwise, a uniform weight $\{1, 1, 1, 1, 1\}$ vector will be assign to Q.

4.4 Experiments and Results

We used two metrics to measure retrieval performance of the image retrieval system: *recall* and *precision* [14]. In general, precision and recall are used together in a graph so that they can show the change of precision values with respect to the recall values. Every image in the image database has been used for testing the precision and recall.

The methods we compared were the proposed VQ and curvelet combined method; sole VQ retrieval method in [9]; uniform weighting image retrieval; and Gauss Mixture VQ retrieval method in [11]. The retrieval result can be seen in Fig. 2. Traditional quantization methods such as SQ were not included in the comparison because the research works in [9] and [11] have already done the comparison, their results have proved the VQ methods have much better performances in image retrieval than SQ-based methods.

As we can see from the result curves, our proposed method performs the best in the compared methods. The combination of curvelet features and VQ has improved the *precision*. The GMVQ has nearly the same results with our method when $recall \leq 40$, but the precision of GMVQ retrieval method drops quickly after the $recall > 40$, this is the reason why we did not chose GMVQ as our VQ method, the proposed retrieval method is able to provide more correlated images for users.

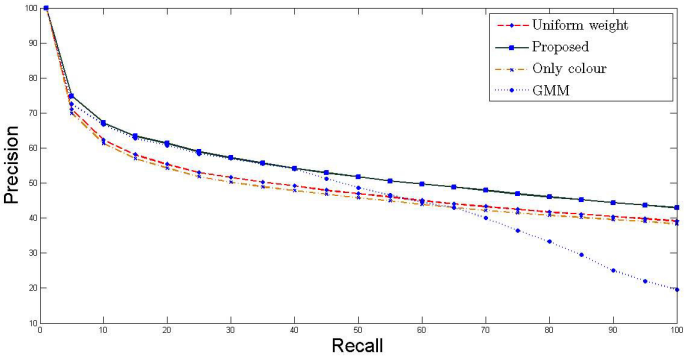


Fig. 2. Comparison of different image retrieval methods(1000 images' average)

5 Conclusion

In this paper, we have introduced a GA-based method to combine curvelet transform and color vector quantization for image retrieval. The Genetic Algorithm was employed for training the weights of features for each image category, an automatic weights selection strategy is also explained. Through the experiments, we demonstrated that the proposed scheme can achieve better retrieval performance than some state-of-art image retrieval techniques. The future work will include studying the proposed method in huge amount image database . The other optimization tools such as particle swarm optimization (PSO) will also be considered to be used in the future research.

References

1. Ahalt, S.C., Krishnamurthy, A.K., Chen, P., Melton, D.E.: Competitive learning algorithms for vector quantization. Neural Networks 3, 277–290 (1990)
2. Arivazhagan, S., Ganesan, L.: Texture classification using wavelet transform. Pattern Recognition Letters 24, 1513–1521 (2003)
3. Candès, E.J., Demanet, L., Donoho, D.L., Ying, L.: Fast discrete curvelet transform. Multiscale Modeling and Simulation 5(3), 861–899 (2006)
4. Goldberg, D.E.: Genetic algorithms: In search, optimization and machine learning. Addison-Wesley (1987)
5. Flickner, M., Sawhney, H., Niblack, W., Ashley, J., Huang, Q., Dom, B., Gorkani, M., Hafner, J., Lee, D., Pettovic, D., Steele, D., Anker, P.: Querying by image and video content: the QBIC system. IEEE Transactions on Computers 25, 23–32 (1995)
6. Gebäck, T., Koumoutsakos, P.: Edge detection in microscopy images using curvelets. BMC Bioinformatics 10(75), 1–14 (2009)
7. Gevers, T., Smeulders, A.: PicToSeek: Combining color and shape invariant features for image retrieval. IEEE Transactions on Image Processing 9, 102–119 (2000)
8. Qiu, G.: Image coding using a coloured pattern appearance model. In: Visual Communication and Image Processing 2001 (2001)

9. Qiu, G.: Indexing chromatic and achromatic patterns for content-based colour image retrieval. Pattern Recognition 35, 1675–1686 (2002)
10. Jain, A.K., Vailaya, A.: Image retrieval using color and shape. Pattern Recognition 29(8), 1233–1244 (1996)
11. Jeong, S., Won, C.S., Gray, R.M.: Image retrieval using color histogram generated by gauss mixture vector quantization. Computer Vision and Image Understanding 94, 44–66 (2004)
12. Holland, J.H.: Adaption in Natural and Artifical Systems. The University of Michigan Press, Ann Arbor (1975)
13. Laaksonen, J., Koskela, M., Oja, E.: PicSOM: self-organizing image retrieval with mpeg-7 content descriptors. IEEE Transactions on Neural Networks 13(4), 841–852 (2002)
14. Smith, J.R., Chang, S.F.: Tools and techniques for color image retrieval. In: Proc. SPIE Storage and Retrieval for Image and Video Database IV, vol. 2670, pp. 426–437 (1996)
15. Smith, J.R., Chang, S.F.: VisualSeek: a fully automated content-based image query system. In: Proceedings of the ACM Multimedia, pp. 87–98 (1996)
16. Sumana, I.J., Islam, M.M., Zhang, D., Lu, G.: Content based image retrieval using curvelet transform. In: IEEE 10th Workshop on Multimedia Signal Processing, pp. 11–16 (2008)
17. Tsai, C.W., Lee, C.Y., Chiang, M.C., Yang, C.S.: A fast vq codebook generation algorithm via pattern reduction. Pattern Recognition Letters 30, 653–660 (2009)

Self-configuring Genetic Algorithm with Modified Uniform Crossover Operator

Eugene Semenkin and Maria Semenkina

Siberian State Aerospace University, Department of System Analysis and Operation Research, Krasnoyarskiy Rabochiy Avenue 31,
660014 Krasnoyarsk, Russian Federation
eugenesemenkin@yandex.ru, semenkina88@mail.ru

Abstract. For genetic algorithms, new variants of the uniform crossover operator that introduce selective pressure on the recombination stage are proposed. Operator probabilistic rates based approach to genetic algorithms self-configuration is suggested. The usefulness of the proposed modifications is demonstrated on benchmark tests and real world problems.

Keywords: genetic algorithms, uniform crossover, selective pressure recombination, self-configuration, performance comparison.

1 Introduction

Evolutionary algorithms (EA), the best known representatives of which are genetic algorithms (GA), are well known optimization techniques based on the principles of natural evolution. Although GAs have been successful in solving many of real world optimization problems, their performance depends on the selection of the GA settings and tuning their parameters. GAs usually use a bit-string solution representation, but other decisions have to be made before the algorithms execution. The design of a GA consists of choosing of variation operators (e.g. recombination and mutation) that will be used to generate new solutions from the current population and the parent selection operator (to decide which members of the population are to be used as inputs to the variation operators), as well as a survival scheme (to decide how the next generation is to be created from the current one and outputs of the variation operators). Additionally, real valued parameters of the chosen settings (the probability of recombination, the level of mutation, etc.) have to be tuned.

The process of setting choice and parameter tuning is known as a time-consuming and complicated task. Much research has tried to deal with this problem. Some approaches attempted to determine appropriated setting by experimenting over a set of well-defined functions or through theoretical analysis. Another approach, usually applying terms such as "self-adaptation" or "self-tuning", tries to eliminate the setting process by adapting settings through the algorithm execution.

There exist much research devoted to "self-adapted" or "self-tuned" GAs and authors of corresponding papers determine similar ideas in very different ways, all of them aimed at reducing the human expert role in algorithms designing.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 414–421, 2012.

In this paper, we follow the definitions given by Gabriela Ochoa and Marc Schoenauer, organizers of the workshop "Self-tuning, self-configuring and self-generating evolutionary algorithms" (Self* EAs) within PPSN XI [1]. According to this definition, "… *The following 3 general paths toward automated heuristic design will be distinguished: 1) tuning: the process of adjusting the control parameters of the algorithm; 2) configuring: the process of selecting and using existing algorithmic components (such as variation operators); 3) generating: the process of creating new heuristics from existing sub-components borrowed from other search methods*…". The main idea of the approach proposed here relies on automated "selecting and using existing algorithmic components". That is why these algorithms might be called self-configuring ones. Within this, some parameters, namely the probabilities of the genetic operator use, are subject to automated tuning, this allows us to say that the algorithms are partially self-tuning ones.

In order to specify our algorithms more precisely, one can say that according to the classification in [2], we use dynamic adaptation on the population level ([3]). The probabilities of applying genetic operators are changed "on the fly" through the algorithm execution. According to the classification given in [4], we use centralized control techniques (central learning rule) for parameter settings with some differences from the usual approaches. Operator rates (the probability to be chosen for generating off-spring) are adapted according to the relative success of the operator during the last generation independently of the previous results. This is how our algorithm avoids the problem of high memory consumption typical for centralized control techniques [4]. Operator rates are not included in individual chromosomes and they are not subject to the evolutionary process. All operators can be used during one generation for producing off-springs one by one.

Having conducted numerical experiments, we found the proposed approach positively impacts an algorithms' performance and deserves special attention and further investigation.

The rest of the paper is organized as follows. Section 2 explains the idea of selective pressure during the stage of individual crossover in GA and describes the results of the algorithm performance investigation. Section 3 describes the proposed method of GA self-configuring and its testing results. Section 4 describes the results of the numerical experiments comparing the performance of the proposed approach in solving real-world problems, and in the Conclusion section we discuss the results.

2 Uniform Crossover Operators with Selective Pressure

The uniform crossover operator is known as one of the most effective crossover operators in conventional genetic algorithm [5, 6]. Moreover, from nearly the beginning, it was suggested to use parameterized uniform crossover operators and it was shown that by tuning this parameter (the probability for a parental gene to be included in an off-spring chromosome), one can essentially improve "The Virtues" of this operator [6] such that it can emulate other crossover operators. Nevertheless, in the majority of cases using uniform crossover the mentioned possibility is not adopted and the probability for parental gene to be included in an off-spring chromosome is given probability equal to 0.5 [7, 8].

This is why it seemed interesting to modify the uniform crossover operator with the purpose of improving its performance. Having a desire to avoid real number parameter tuning, we suggest introducing selective pressure during recombination [9] making the probability of parental gene to be taken for off-spring dependent on parent fitness values. Like the usual GA selection operators, fitness proportional, rank-based and tournament-based uniform crossover operators have been added to the conventional operator which we now call the equiprobable uniform crossover.

The performance of the conventional GA with three additional uniform crossover operators has been evaluated on the usual GA test problems [10]. Results are given in the first six rows of Table 1 below.

Table 1 contains the average, minimal and maximal reliability of the algorithms averaged over 14 test problems from [10], each solved 1000 times. The reliability is the portion of the algorithm runs that gives satisfactorily precise solutions. In Table 1, row headers "1-point, 2-point, UE, UT, UP, UR" indicate the type of crossover, respectively, 1-point, 2-point, uniform equiprobable, uniform tournament-based, uniform fitness proportional and uniform rank-based crossovers.

"Average" means the reliability of an algorithm averaged over all settings of the other operators (selection, mutation, population control). "Min" ("Max") means the reliability of the worst (best) setting variant for the corresponding crossover. The numbers in brackets demonstrate the variance in these indicators. The first number in brackets is the minimal value among the 14 tests, the second number in brackets is the maximal value among the 14 tests. The last number is the corresponding indicator averaged over the 14 test functions.

Table 1. Results of GAs performance evaluation

Crossover	Average	Min	Max
1-Point	[0.507,0.915] / 0.760	[0.411,0.856] / 0.696	[0.591,0.978] / 0.822
2-Point	[0.132,0.821] / 0.479	[0.000,0.754] / 0.413	[0.167,0.871] / 0.534
UE	[0.442,0.953] / 0.819	[0.589,0.887] / 0.780	[0.669,0.999] / 0.878
UT	[0.354,0.967] / 0.627	[0.299,0.917] / 0.587	[0.380,1.000] / 0.697
UP	[0.276,0.935] / 0.647	[0.232,0.888] / 0.622	[0.300,0.976] / 0.718
UR	[0.598,0.974] / 0.833	[0.578,0.935] / 0.771	[0.635,0.999] / 0.888
SelfCGA-1	[0.678,0.998] / 0.880		
SelfCGA-2	[0.830,1.000] / 0.928		

After multiple runs and statistical processing of the numerical results, the following observations were found (in terms of algorithm reliability). The best variants are new rank-based and conventional (equiprobable) uniform operators. Tournament-based crossover seems to be weak, but it is because the number of parents that is set equal to 2 in these experiments. In this case tournament-based crossover just copies the better parent and results to population diversity loss. The last observation suggests examining multi-parent recombination with the new uniform crossover operators.

The performance of GAs with additional uniform crossover operators and multi-parent recombination has been evaluated on the same test optimization problems. We observed that with one exception the best number of parents is 7 and the second best

is the conventional 2. In the case of tournament-based crossover we observed that "the more parents the better". This operator again seems to be weak on average, but it is the only operator having maximum reliability of 100% on some test problems where other operators fail. In our further investigations we used 3 and 7 parents for tournament-based uniform crossover and 2 and 7 for all others. One can also establish that we shouldn't exclude any crossover operator type from consideration because all of them can be useful.

Although the proposed new operators give higher performance than conventional operators, at the same time it increases the number of algorithm setting variants and complicates algorithm adjusting for the end user. That is why we have to suggest a way to avoid this extra effort for adjustment.

3 Self-configuring Genetic Algorithm Based on Operators' Rates

As mentioned in the Introduction, we apply dynamic adaptation on the level of population with centralized control techniques to the operator probabilistic rates. In order to avoid real parameter precise tuning, we used setting variants, namely types of selection, crossover, population control and level of mutation (medium, low, high). Each of these has its own probability distribution, e.g., there are 5 settings of selection – fitness proportional, rank-based, and tournament-based with three tournament sizes. During the initialization phase all probabilities are equal to 0.2 and they will be changed according to a special rule through the algorithm's execution in such a way that the sum of the probabilities should be equal to 1 and no probability could be less than a predetermined minimum balance. The list of crossover operators includes 11 items, i.e., 1-point, 2-point and four uniform crossovers all with two numbers of parents (2 and 7). The "idle crossover" is included in the list of crossover operators to make a crossover probability of less than 1 that is used in conventional algorithms to model a "childless couple".

When the algorithm has to create the next off-spring from the current population, it firstly must configure settings, i.e. form the list of operators with the use of probability operator distributions. Then the algorithm selects parents with the chosen selection operator, produces an off-spring with the chosen crossover operator, mutates this off-spring with the chosen mutation probability and puts it into an intermediate population. When the intermediate population is complete, the fitness evaluation is executed and the operator rates (probabilities to be chosen) are updated according to the operator's productivity. Then the next parents' population is formed with the chosen survival selection operator. The algorithm stops after a given number of generations or if termination criterion (e.g., the given error minimum) is met.

Productivity of an operator is the ratio of the average off-spring's fitness obtained with this operator and the off-spring population average fitness. A successful operator having maximal productivity increases its rate obtaining portions from other operators. There is no necessity in extra computer memory to remember past events and the reaction of updates are more dynamic (that can be both a plus and a minus).

Below in Table 2 one can find the result comparing the proposed self-configuring GA (SelfCGA) with the best single algorithms having the best performance on the corresponding problem. Saying "single" algorithm, we mean the group of algorithms with the same crossover operator but with all variants of other settings. The average reliability of this "single" algorithm is averaged over all possible settings.

Table 2. Comparison results of SelfCGA and problem best single algorithms

No	Crossover	Average	Min	Max	No	Crossover	Average	Min	Max
1	UE	0.818	0.787	0.894	8	UR	0.741	0.667	0.800
	SelfCGA	*0.886*				SelfCGA	**0.830**		
2	UE	0.841	0.808	0.903	9	UT	0.967	0.917	0.983
	SelfCGA	*0.866*				SelfCGA	**0.987**		
3	UE	0.901	0.887	0.921	10	UE	0.853	0.803	0.891
	SelfCGA	*0.901*				SelfCGA	*0.884*		
4	UR	0.925	0.877	0.959	11	UR	0.821	0.734	0.888
	SelfCGA	**0.976**				SelfCGA	**0.892**		
5	UT	0.950	0.901	1.00	12	UR	0.833	0.765	0.881
	SelfCGA	**1.000**				SelfCGA	**0.897**		
6	UE	0.953	0.861	0.999	13	UR	0.956	0.902	0.998
	SelfCGA	**1.00**				SelfCGA	**1.000**		
7	UT	0.897	0.832	0.927	14	UR	0.974	0.935	0.999
	SelfCGA	0.878				SelfCGA	**1.000**		

Analyzing Table 2, we can see that in four cases (1, 2, 3, 10 – numbers are given in italics) SelfCGA demonstrates a reliability better than the average reliability of the single best algorithm for the corresponding problem but worse than the maximal one. In one case (7^{th} problem), the best single algorithm (with tournament-based uniform crossover) gives better average performance than SelfCGA. In the 9 remaining cases (numbers are given in bold) SelfCGA outperforms even single algorithm with maximal reliability.

Having described our results, one can conclude that proposed GA self-configuration not only allows one to avoid the time consuming efforts for determining the best settings but also can give a performance improvement even in comparison with the best known settings of conventional GAs.

The described approach can be used in many EA techniques, in particular with conventional GA. That is why we compared GA self-configuration based on three conventional crossovers (1-point, 2-points, uniform) and GA self-configuration based on all 6 crossover operators. We call the first algorithm SelfCGA-1 and the second SelfCGA-2.

The results of the self-configuring GAs performance evaluation averaged over the same 14 test optimization problems are given in the last two rows in Table 1 above.

This provides the evidence that SelfCGA-1 works better than all three single algorithms combined within it. It permits us to say that GA self-configuring itself increases the algorithms' performance. Also one can see that SelfCGA-2 outperforms SelfCGA-1 in all indicators, i.e. it has greater average performance and better variance edges. This implies that the proposed types of uniform crossover with selective pressure play a positive role in the interaction of the GAs with different settings, giving a hope for better results in real world problem solving.

4 Numerical Experiments with Real World Problems

Having seen positive result when applying the algorithms on benchmark test optimization problems, there is a need to test them on real world optimization problems. The developed algorithms were applied to two classification machine learning credit scoring problems often used to compare the accuracy with various classification models from UCI repository [11]:

- Credit (Australia-1) (14 attributes, 2 classes, 307 examples of the creditworthy customers and 383 examples for the non-creditworthy customers);
- Credit (Germany) (20 attributes, 2 classes, 700 records of the creditworthy customers and 300 records for the non-creditworthy customers).

Both classification problems were solved with artificial neural network (ANN) based classifiers having fixed structure (one hidden layer perceptron with 3 or alternatively 5 hidden neurons, one output neuron and 14 or 20 input neurons). These ANNs have to be trained on 70% of the data base instances and validated on the remaining 30% of the examples. The results of validations (portion of correctly classified instances from validation sets) are given in table below.

From the optimization view point, training ANNs with 3 or 5 hidden neurons requires tuning from some tens till hundred real value weight coefficients. With precision to 0.01 and interval [-5; 5] this gives from 550 till 1050 bits in chromosome. This seems to be a challenge for optimization techniques.

We first conducted a comparison with alternative optimization techniques, namely standard error back propagation, conventional GA, modified GA with our new uniform crossover operators with selective pressure, and SelfCGA-2. As we have observed, both algorithms proposed in this paper demonstrate high workability on both classification tasks.

We then conducted the comparison of our ANN-based classifier with alternative classification techniques. Results for the alternative approaches have been taken from the scientific literature. In [12] the performance evaluation results for these two data sets are given for the authors' two-stage genetic programming algorithm (2SGP) as well as for the following approaches taken from other papers: conventional genetic programming (GP), multilayered perceptron (MLP), classification and regression tree (CART), C4.5 decision trees, k nearest neighbors (k-NN), linear regression (LR). We have taken additional material for comparison from [13] which includes evaluation data for the authors' automatically designed fuzzy rule based classifier as well as for other approaches found in the literature: Bayesian approach, boosting, bagging, random subspace method (RSM), cooperative coevolution ensemble learning (CCEL). The results obtained are given in Table 3. As one can see from Table 3, both proposed algorithms demonstrate competitive performance, taking 2nd and 3rd places on the easier Australian credit problem and 7th and 8th of 15 places on the harder German credit task.

We understand the limitations of this comparison, e.g. there is no information on indicators' variation or on computational efforts for compared methods. Some results are the best for the given method, others are averaged. Moreover, it is clear that 2SGP and fuzzy classifier are much more useful for decision makers as they give not only a computational expression but also human expert understandable production rules.

Table 3. The comparison of classification algorithms

Classifier	Australian credit	German credit	Classifier	Australian credit	German credit
SelfCGA-2	0.9013	0.7617	C4.5	0.8986	0.7773
MGA	0.8997	0.7604	LR	0.8696	0.7837
2SGP	**0.9027**	**0.8015**	CCEL	0,8660	0,7460
GP	0.8889	0.7834	k-NN	0.7150	0.7151
Fuzzy classifier	0,8910	0,7940	CART	0.8744	0.7565
Bayesian approach	0,8470	0,6790	MLP	0.8986	0.7618
Boosting	0,7600	0,7000	RSM	0,8520	0,6770
Bagging	0,8470	0,6840			

Our intention was to make it clear whether our approach could give results competitive to alternative techniques without attempting to develop the best tool for bank credit scoring. This is why we did not use a larger ANN and limited computational resources to 500 generations (rather than 1000 as, e.g., in [12]), etc.

We conclude that the self-configuring genetic algorithm proposed in this paper can produce competitive results, it has the usual drawbacks of any general-purpose technique losing to the problem specific algorithms on corresponding problems but has the advantage of requiring no algorithm adjustment.

5 Conclusions

In this paper, special kind of uniform crossover operator that introduces selective pressure on the recombination stage for GA is proposed. They are fitness proportional, tournament-based and rank-based uniform crossover operators similar to the well known selection schemes of evolutionary computations. It is demonstrated on the benchmarks that the use of these operators gives positive impact on GA performance. In particular, the rank-based uniform crossover is the best among all types of crossover operators.

Then we presented a modified approach to probability based operator rate assignments for the automated configuration of algorithms that assumes the probabilistic choice of operators for every next off-spring and updates rates after every generation. This modification also turned to be useful as was demonstrated with test problems.

And at last, we checked our GA modifications using them in training ANN-based classifiers for solving two hard classification problems that demonstrated the usefulness and perspectiveness of the proposed modifications.

Acknowledgment. The research is partially supported through Governmental contracts № 16.740.11.0742 and № 11.519.11.4002.

References

1. Schaefer, R., Cotta, C., Kołodziej, J., Rudolph, G. (eds.): PPSN XI. LNCS, vol. 6238. Springer, Heidelberg (2010)
2. Angeline, P.J.: Adaptive and self-adaptive evolutionary computations. In: Palaniswami, M., Attikiouzel, Y., Marks, R., Fogel, D., Fukuda, T. (eds.) Computational Intelligence: A Dynamic Systems Perspective, pp. 152–163. IEEE Press, Piscataway (1995)

3. Meyer-Nieberg, S., Beyer, H.-G.: Self-Adaptation in Evolutionary Algorithms. In: Lobo, F.G., Lima, C.F., Michalewicz, Z. (eds.) Parameter Setting in Evolutionary Algorithm. SCI, vol. 54, pp. 47–75. Springer, Heidelberg (2007)
4. Gomez, J.: Self Adaptation of Operator Rates in Evolutionary Algorithms. In: Deb, K., et al. (eds.) GECCO 2004, Part I. LNCS, vol. 3102, pp. 1162–1173. Springer, Heidelberg (2004)
5. Syswerda, G.: Uniform crossover in genetic algorithms. In: Schaffer, J.D. (ed.) Proc. of the 3rd International Conference on Genetic Algorithms, pp. 2–9. Morgan Kaufmann (1989)
6. Spears, W., De Jong, K.A.: On the Virtues of Parameterized Uniform Crossover. In: Belew, R.K., Booker, L.B. (eds.) Proceedings of the 4th International Conference on Genetic Algorithms, pp. 230–236. Morgan Kaufmann (1991)
7. Haupt, R.L., Haupt, S.E.: Practical genetic algorithms. John Wiley & Sons, Inc., Hoboken (2004)
8. Eiben, A.E., Smith, J.E.: Introduction to evolutionary computing. Springer, Heidelberg (2003)
9. Semenkin, E.S., Semenkina, M.E.: Application of genetic algorithm with modified uniform recombination operator for automated implementation of intellectual information technologies. Vestnik. Scientific Journal of the Siberian State Aerospace University named after academician M.F. Reshetnev. 3(16), 27–32 (2007) (in Russian, abstract in English)
10. Finck, S., et al.: Real-parameter black-box optimization benchmarking 2009: Presentation of the noiseless functions. Technical Report 2009/20, Researh Center PPE (2009)
11. Frank, A., Asuncion, A.: UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science (2010), http://archive.ics.uci.edu/ml
12. Huang, J.-J., Tzeng, G.-H., Ong, C.-S.: Two-stage genetic programming (2SGP) for the credit scoring model. Applied Mathematics and Computation 174, 1039–1053 (2006)
13. Sergienko, R., Semenkin, E., Bukhtoyarov, V.: Michigan and Pittsburgh Methods Combining for Fuzzy Classifier Generating with Coevolutionary Algorithm for Strategy Adaptation. In: IEEE Congress on Evolutionary Computation. IEEE Press, New Orleans (2011)

Fitness Function Based on Binding and Recall Rate for Genetic Inductive Logic Programming

Yanjuan Li[1,2] and Maozu Guo[1]

[1] School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
[2] School of Information and Computer Engineering,
North-East Forestry University, Harbin, China

Abstract. The key of using genetic inductive logic programming (GILP) algorithm to learn first-order rules is how to precisely evaluate the quality of first-order rules. That is, the fitness of rules should rightly score their quality and effectively guide GILP algorithm to be close to the target rule. In this paper, a new fitness function is proposed. By adopting the concept of binding, the new fitness function can adequately utilize the information hidden in background knowledge and training examples. By considering recall rate of rules, the new fitness function can avoid generating over-specific rules. Experiments on benchmark data set show that comparing with the common fitness function based on amount of examples covered by rules, the new fitness function can measure quality of first-order rules more precisely and enhance predictive accuracy of GILP.

Keywords: machine learning, inductive logic programming, genetic inductive logic programming, genetic algorithm.

1 Introduction

The field of inductive logic programming (ILP) has matured enough in recent years so that researchers in this field now tackle real world problems, such as web page classification [1], gene regulation prediction [2], metabolic networks [3] and grammar rules of building parts [4]. One strength of ILP lies in the fact that a first-order representation is employed. Such a representation makes ILP overcome the two main limitations of classical machine learning techniques: 1) the use of a limited knowledge representation formalism (essentially a propositional logic), and 2) difficulties in using substantial background knowledge in the learning process [5].

ILP system can be formulated as a search problem in a hypotheses space of logic programs [6]. Various approaches [7], [8] in ILP systems are mainly different in the search strategy and the heuristics which are used to guide the search. The search space is extremely large, so strong heuristics are required. Greedy search strategy is used in most systems. These systems generate a sequence of logic programs from general to specific (or from specific to general) until a consistent program is found. Each program in the sequence is obtained by specializing (or generalizing) the previous one. For example, FOIL [9] applies a hill climbing search strategy guided by an information-gain heuristic

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 422–427, 2012.

to search programs from general to specific. CLAUDIEN [10] uses a measure that takes into account the length of the clause. But these strategies and heuristics are not always applicable because the systems may become trapped in local maxima. In order to overcome the problem of local maxima, non-greedy strategies such as genetic algorithms are adopted. Genetic algorithms [11] are alternative search strategies which perform an implicitly parallel search. An inductive logic programming system that employs genetic algorithm to search the whole candidate is called as genetic inductive logic programming (GILP). GILP employs genetic algorithm for searching the space of candidate clauses, and the stochastic search method can overcome the problem of local optima of deterministic search. Therefore, GILP approaches become an attractive topic in inductive logic programming.

GA-SMART [12] is the first GILP system in which the individuals are fixing length binary strings encoding formula, and the size of an individual grows with the number of predicates appearing in it. QG/GA [13] carries out a random-restart stochastic bottom-up search which efficiently generates a consistent clause on the fringe of the refinement graph search without needing to explore the graph in detail, then uses a genetic algorithm to evolve and re-combine the generated clauses. PGA [14], [15] uses a multiple level encoding structure that can represent three different types of relationships between two numerical data. However, the common fitness function used in GILP is based on the number of examples covered by rules, which can not adequately utilize the information hidden in background knowledge and training examples and is apt to generating over-specific hypotheses. By adopting the concept of binding and considering the recall rate of rules, a new fitness function is proposed. An analysis and contrastive experiment demonstrate that the new fitness function can more effectively guide the search direction of algorithm and enhance predictive accuracy of GILP algorithm.

2 Fitness Function

In this section, we firstly introduce the common fitness function based on the number of examples covered by rules and analyze its weakness. Then, a new fitness function is proposed. By adopting the concept of binding, the new fitness function can adequately utilize the information hidden in background knowledge and training examples. By considering the recall rate of rules, the new fitness function can avoid generating over-specific rules.

2.1 Common Fitness Function and Its Weakness

The common fitness function used in genetic inductive logic programming algorithm is based on the number of examples covered by rules and is defined as:

$$fitnesscommon(H) = \frac{TP}{TP + FP} \tag{1}$$

Where H is an individual, that is a hypotheses(rule); TP=|{e | e∈E^+∧B∪H|=e}|, the number of the set of true positive examples, i.e., the number of the set of positive

examples that are covered by the hypotheses H; FP=|{ e | e∈E⁻∧B∪H|=e}|, the number of the set of false positive examples, i.e., the number of the set of negative examples that are covered by the hypotheses H.

Fitness function *fitnesscommon* is only decided by the ratio of the true positive examples to the false positive example, and does not consider the exact number of true positive examples. There are three weaknesses for the common fitness function defined in Eq. (1) as follows.

(1) It does not distinguish equivalence rules. Let H_1 and H_2 be two rules (hypotheses), if H_1 and H_2 have same TP and FP value, then H_1 and H_2 are equivalence rules. According to Eq. (1), two equivalence rules have same fitness. Therefore, the common fitness function does not distinguish equivalence rules.

(2) It does not distinguish the two hypotheses that they have the same ratio of the true positive examples to the false positive examples but have different number of true positive examples. According to Eq. (1), if two hypotheses have the same ratio of the true positive examples to the false positive examples, then the two hypotheses have same fitness. For example, H_1, H_2 are two hypotheses, *TP* of H_1 is 1, *FP* of H_1 is 0, *TP* of H_2 is 100, *FP* of H_2 is 0. According to Eq. (1), both the fitness of H_1 and the fitness of H_2 are equal to 1. That is H_1 and H_2 have same fitness. In fact, the performance of H_2 is obviously better than H_1.

(3) It is biased to generate over-specific hypotheses. For example, H_1, H_2 are two hypotheses, *TP* of H_1 is 1, *FP* of H_1 is 0, *TP* of H_2 is 500, *FP* of H_2 is 1. According to Eq. (1), the fitness of H_1 is higher than that of H_2, thus GILP algorithm employing *fitnesscommon* as fitness function is apt to generate over-specific hypotheses.

2.2 New Fitness Function

To overcome the weaknesses of fitness function *fitnesscommon*, a new fitness function *fitnessnew* is proposed. Before giving the definition of the new fitness function, we firstly induce the concept of binding [16] and recall rate [17]. A substitution $\theta=\{X_{p1}/t_{p1},\ldots, X_{pn}/t_{pn}, X_{q1}/t_{q1},\ldots, X_{qm}/t_{qm}, X_{r1}/t_{r1},\ldots, X_{ri}/t_{ri}\}$ maps all variables of rule $R=p(X_{p1},\ldots,X_{pn})$:-$q(X_{q1},\ldots,X_{qm})$, $r(X_{r1},\ldots,X_{ri})$ into the constants in data.

Definition1. *Let R be a rule (hypotheses), if under a substitution θ, the condition of rule R is true, then the substitution θ is called as a binding of rule R. Let θ be a binding of rule R, if under the binding θ, the conclusion of rule R is also true, i.e. there is a corresponding positive example in data, then the binding θ is called as a positive binding of rule R, otherwise, it is a negative binding of rule R.*

The concept of binding considers various cases under which the condition of rules is true, so it can thoroughly utilize the information hidden in background knowledge and examples to measure the quality of rules.

Definition2. *Given a hypotheses (rule) H and an example E. If the example is positive and it is covered by H, it is counted as a true positive; if it is not covered by H, it is counted as a false negative. Given a hypotheses H and a set of examples, the recall rate of H is equal to the ratio of the number of true positive examples to the sum of true positive examples and false negative examples.*

By adopting the concept of binding and considering the recall rate of hypotheses, a new fitness function is defined as:

$$fitnessnew(H) = \begin{cases} \dfrac{TB}{TB+FB} & RR > FS \\ \dfrac{RR}{FS} * \dfrac{TB}{TB+FB} & else \end{cases} \quad (2)$$

Where H is a hypotheses, the definitions of *TP* and *FP* are same as those in Eq. (1); TB is the number of positive binding of hypotheses H; FB is the number of negative binding of hypotheses H; *RR* is the recall rate of H; FS is a real number and FS∈[0,1].

The new fitness function can overcome the three weaknesses of *fitnesscommon*. In detail, for weakness (1), H_1 and H_2 are equivalence rules, that is H_1 and H_2 have same TP and FP. but H_1 and H_2 have different TB and FB. Therefore, according to Eq. (2), H_1 and H_2 have different fitness, that is new fitness function overcomes the weakness (1) of the common fitness function; for weakness (2), let H_1 and H_2 have same ratio of the true positive examples to the false positive examples but have different number of true positive examples. According to Eq. (2), the fitness of H_1 is different to the fitness of H_2; for weakness (3), because *fitnessnew* considers the recall rate of rules, *fitnessnew* can avoid generating over-specific hypotheses.

3 Experiment

3.1 Date Set

Four benchmark data sets [18] are used in the experiments. Information on these data sets is tabulated in Table 1, where "Bk relation" presents the number of predicates in background knowledge, "size" presents the number of examples, "class" presents the number of category, "pos/neg" presents the percentage of positive examples against that of negative examples. For each data set, about 25 percent of the data are kept as test examples to evaluate the performance of the learned hypothesis, while the rest are used as training examples. In principle, the pos/neg ratio on training set and test set are identical to that on the original data set.

Table 1. Benchmark data sets used in experiments

dataset	Bk relation	size	class	pos/neg
Alzheimer amine	20	686	2	50%/50%
Alzheimer toxic	20	886	2	50%/50%
Alzheimer acetyl	20	1326	2	50%/50%
Alzheimer memory	20	642	2	50%/50%

3.2 Results

The details of the coding, crossover and mutation operator of GILP algorithm are given in the reference [19, 20]. We respectively test the common fitness function

defined in Eq. (1) and the new fitness function defined in Eq. (2). Table 2 shows the experiment results of *fitnesscommon* and *fitnessnew*. The column |*h*| in table 2 presents the number of learned hypotheses, the column *Iterations Num.* presents the iteration number when the algorithm is over, the column *trainacc* presents the accuracy of hypotheses on the training data set, the column *testacc* presents the accuracy of hypotheses on the test data set. For each data set row, the highest predictive accuracy on test data has been boldfaced.

Table 2. The experiment result of GILP adopting two different fitness functions

dataset	*fitnessnew*				*fitnesscommon*			
	|*h*|	*Iterations Num.*	*trainacc*	*testacc*	|*h*|	*Iterations Num.*	*trainacc*	*testacc*
amine	9	43	0.8158	**0.8117**	67	204	0.9244	0.7352
toxic	12	16	0.9157	**0.9045**	43	196	0.9519	0.8954
acetyl	46	167	0.8483	**0.7606**	92	388	0.8534	0.7454
memory	10	38	0.7219	**0.7187**	56	124	0.7966	0.6750
Average	19.2	66	0.8254	**0.7989**	64.5	228	0.8816	0.7627

Table 2 shows that the *Iterations Num.* of *fitnessnew* is lower than that of *fitnesscommon*. This fact proves that *fitnessnew* has a rapid convergence compared with *fitnesscommon*. It is shown in table 2, on all data set, the hypotheses number and training accuracy of *fitnesscommon* are higher than the ones of *fitnessnew*, while the test accuracy of *fitnesscommon* is lower than that of *fitnessnew*. This fact proves that *fitnesscommon* learned over-specific hypotheses. The predictive accuracy of *fitnessnew* is higher than that of *fitnesscommon* on all dataset, this proved that the performance of *fitnessnew* is higher than *fitnesscommon*.

4 Conclusion

The key of using genetic inductive logic programming (GILP) algorithm to learn first-order rules is how to precisely evaluate the quality of first-order rules. A new fitness function is proposed. By adequately utilizing the information hidden in background knowledge and training examples and considering the recall rate of rules, the new fitness function not only measures the quality of first-order rules precisely, but also it can avoid generating over-specific rules. Experiments on benchmark dataset demonstrate that the new fitness function can more effectively guide the search direction of algorithm and enhance predictive accuracy of algorithm.

Acknowledgements. The work was supported by the Natural Science Foundation of China under Grant No. 61171185, No. 60932008, and No. 60832010, the Specialized Research Fund for the Doctoral Program of Higher Education of China under Grant No. 20112302110040 and China Postdoctoral Science Special Foundation under Grant No. 201003446, and the Fundamental Research Funds for the Central Universities under Grant No. DL12AB02, and China Postdoctoral Science Foundation under Grant No. 20110491059.

References

1. Li, Y., Guo, M.: Web page classification using relational learning algorithm and unlabeled data. Journal of Computers 6(3), 474–479 (2011)
2. Fröhler, S., Kramer, S.: Inductive logic programming for gene regulation prediction. Machine Learning 70(2), 225–240 (2008)
3. Biba, M., Xhafa, F., Esposito, F., Ferilli, S.: Stochastic simulation and modeling of metabolic networks in a machine learning framework. Simulation Modeling Practice and Theory 19, 1957–1966 (2011)
4. Dehbi, Y., Plmer, L.: Learning grammar rules of building parts from precise models and noisy observations. Journal of Photogrammetry and Remote Sensing 66, 166–176 (2011)
5. Kavurucu, Y., Senkul, P., Toroslu, I.H.: A comparative study on ILP-based concept discovery systems. Expert Systems with Applications 38, 11598–11607 (2011)
6. De Raedt, L.: Logical and relational learning. Springer (2008)
7. Kavurucu, Y., Senkul, P., Toroslu, I.H.: Concept discovery on relational databases: New techniques for search space pruning and rule quality improvement. Knowledge-Based Systems 23, 747–756 (2010)
8. Mutlu, A., Senkul, P., Kavurucu, Y.: Improving the scalability of ILP-based multi-relational concept discovery system through parallelization. Knowledge-Based Systems 27, 352–368 (2012)
9. Quinlan, J.R.: Learning Logical Definitions from Relations. Machine Learning 5(3), 239–266 (1990)
10. Raedt, L.D., Dehaspe, L.: Clausal discovery. Machine Learning 26, 99–146 (1997)
11. Jiang, H., Yang, X., Yin, K., Zhang, S., Cristoforo, J.A.: Multi-path QoS-Aware Web Service Composition using Variable Length Chromosome Genetic Algorithm. Information Technology Journal 10, 113–119 (2011)
12. Giordana, A., Sale, C.: Learning structured concepts using genetic algorithms. In: Sleeman, D., Edwards, P. (eds.) The 9th International Workshop on Machine Learning. Morgan Kaufmann (1992)
13. Muggleton, S., Tamaddoni-Nezhad, A.: QG/GA: a stochastic search for Progol. Machine Learning 70, 121–133 (2008)
14. Chien, Y.-W.C., Chen, Y.-L.: A phenotypic genetic algorithm for inductive logic programming. Expert Systems with Applications 36, 6935–6944 (2009)
15. Chien, Y.-W.C., Chen, Y.-L.: Mining associative classification rules with stock trading data-A GA-based method. Knowledge-Based Systems 23, 605–614 (2010)
16. Quinlan, J.R.: Learning logical definitions from relations. Machine Learning 5(3), 239–266 (1990)
17. Davis, J., Goadrich, M.: The relationship between precision-recall and roc curves. In: Proceedings of the 23rd International Conference on Machine Learning (2006)
18. Li, Y., Guo, M.: A relational learning algorithm combining relational tri-training and relational instance-based learning. Journal of Information and Computational Science 9(2), 425–436 (2012)
19. Yang, X., Liu, C., Zhang, J.: Using genetic algorithm to mine first-order rules. Computer Engineering and Applications 38(17), 28–30 (2002) (in Chinese)
20. Yang, X., Liu, C.: Grouwh phenomenon of individuals' code length in genetic inductive logic programming. Journal of Computer Research and Development 8, 1238–1243 (2003) (in Chinese)

LMI-Based Lagrange Stability of CGNNs with General Activation Functions and Mixed Delays

Xiaohong Wang and Huan Qi

Department of Control Science and Engineering,
Huazhong University of Science and Technology, Wuhan, Hubei, 430074, China
wxhong2006@163.com

Abstract. This paper deals with the problem on Lagrange stability of Cohen-Grossberg neural networks (CGNNs) with both mixed time delays and general activation functions. By virtue of Lyapunov functional and Halanay delay differential inequality, several criteria in linear matrix inequality form for Lagrange stability of CGNNs are obtained. Meanwhile, the limitation on the activation functions being bounded, monotonous and differentiable is released and detailed estimation of the globally exponentially attractive sets are also given out. Finally, concluding remark is given.

Keywords: Cohen-Grossberg neural networks, Lagrange stability, Globally exponentially attractive set, Time-varying and finite distributed delays.

1 Introduction

Cohen-Grossberg neural networks (CGNNs) [1] have been extensively studied in recent years, and they have been found many applications in solving a number of problems in various scientific disciplines. Such applications very much depend on the dynamical behavior [2-3], especially Lyapunov stability analysis [4-7]. However, these conclusions are not appropriate in the multistable dynamics which have multiple equilibria and some of them are unstable. In this case, it is worth mentioning that Lagrange stability refers to the stability of the total system which does not require the information of equilibrium points. So, it is important to study Lagrange stability. But until now, only a part of works have studied the Lagrange stability for neural networks with time-delays [8-12]. For instance, in [9], the authors studied the globally exponentially stability (GES) in the Lagrange sense for recurrent neural networks basing on [12], and then [10] continued to extend the Lagrange stability to CGNNs based on three kinds of specific activation functions. Similarly, basing on the results of [8], [11] further searched the GES in the Lagrange sense for neutral type recurrent neural networks. However, in allusion to the case of general activation functions, the Lagrange stability analysis for CGNNs by means of Linear Matrix Inequality (LMI) does not appear.

Motivated by the above discussion, our objective in this paper is to study the global exponential stability in the Lagrange sense for the addressed CGNNs. Firstly, section 2 describes some preliminaries, including some necessary notations, definitions, assumptions and lemmas. And then on the basis of the general activation functions

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 428–436, 2012.

and Linear Matrix equality technique, main results are obtained in Section 3. Finally, concluding remark is given in Section 4.

2 Model Description and Preliminaries

Considering the following CGNNs:

$$\dot{x}(t)=-\alpha(x(t))[Dx(t)-Ag(x(t))-Bg(x(t-\tau(t)))-C\int_{t-\sigma(t)}^{t} g(x(s))ds+U]. \quad (1)$$

Where $x(t)=(x_1(t),\cdots,x_n(t))^T$ and $x_i(t)$ is the state variable associated with the neuron; $\alpha(x(t))=diag\{\alpha_1(x_1(t)),\cdots,\alpha_n(x_n(t))\}$ and α_i is an appropriately amplification function. $D=diag\{d_1,\cdots,d_n\}$, and d_i denotes the behaved function. $g(x(t))=(g_1(x_1(t)),\cdots,g_n(x_n(t)))^T$, $g(x(t-\tau(t)))=(g_1(x_1(t-\tau_1(t))),\cdots,g_n(x_n(t-\tau_n(t))))^T$. The activation function g_j shows how the neurons respond to each other. $\int_{t-\sigma(t)}^{t} g(x(s))ds=(\int_{t-\sigma_1(t)}^{t} g_1(x_1(s))ds,\cdots,\int_{t-\sigma_n(t)}^{t} g_n(x_n(s))ds)^T$. The time-varying delay $\tau(t)=(\tau_1(t),\cdots,\tau_n(t))^T$ satisfies $0\le\tau_i(t)\le\tau_i$, and the finite distributed delay $\sigma(t)=(\sigma_1(t),\cdots,\sigma_n(t))^T$ satisfies $0\le\sigma_i(t)\le\sigma_i$. Here τ_i and σ_i are constants. $A=(a_{ij})$, $B=(b_{ij})$, $C=(c_{ij})\in R^{n\times n}$ tell us how the neurons are connected in the network. $U(t)=(U_1(t),\cdots,U_n(t))^T$ and U_i is the input. Function α_i is continuous and satisfies $0<\alpha_i^-\le\alpha_i(\cdot)\le\alpha_i^+$. Here, let $\tau=\max_{1\le i\le n}\{\tau_i\}$, and $\sigma=\max_{1\le i\le n}\{\sigma_i\}$. C[X,Y] is a class of continuous mapping set from the topological space X to the topological space Y. Especially, $C\triangleq[[-h,0],R^n]$, where $h=\max\{\tau,\sigma\}$. For any initial function $\forall\varphi(s)\in C, s\in[t_0-h,t_0]$, the solution of (1) that starts from the initial condition φ will be denoted by $x(t,t_0,\varphi)$ or simply $x(t)$ if no confusion should occur.

Throughout this paper, the symbols R^n and $R^{n\times m}$ stand for the n-dimensional Euclidean space and the set of all $n\times m$ real matrices, respectively. A^T and A^{-1} denote the matrix transpose and matrix inverse. $A>0$ or $A<0$ denotes that the matrix A is a symmetric and positive definite or negative definite matrix. Meanwhile, $A<B$ indicates $A-B<0$ and $\|\bullet\|$ is the Euclidean vector norm. Moreover, in symmetric block matrices, we use $*$ as an ellipsis for the terms that are introduced by symmetry.

Assumption 1. There exist two diagonal matrices $L=diag\{L_1,\cdots,L_n\}$ and $F=diag\{F_1,\cdots,F_n\}$ such that for any $x, y\in R$ and $x\neq y$, the following inequalities hold: $L_i \le \frac{g_i(x)-g_i(y)}{x-y} \le F_i$.

Remark 1. It should be noted that as pointed out in [6-7], the constants $L_i, F_i, i\in R$ in assumption 1 are allowed to be positive, negative or zero. So, the assumption 1 of this paper is weaker than the literatures [2-3], [10-12].

Remark 2. In the literature [4-5] and [9-12], the results were obtained under the condition that time-varying delays were continuously differentiable, of which the derivative was bounded and smaller than one, limiting activation functions being bounded and monotonically non-decreasing. We like to point out that, in our paper, the presented results do not need the conditions mentioned above.

Definition 1. *[12] If there exists a radially unbounded and positive definite Lyapunov function* $V(x(t))$, *which satisfies* $V(x(t)) \le \|x\|^{\tilde{\beta}}$, *where* $\tilde{\beta}>0$ *is a constant, and constants* $\varsigma>0$, $\beta>0$, *such that for* $V(x(t_0))>\varsigma$, $V(x(t))>\varsigma$, $t \ge t_0$, *the inequality* $V(x(t))-\zeta \le (\bar{V}(x(t_0))-\zeta)\exp\{-\beta(t-t_0)\}$ *always holds. Then,* $\{x \mid V(x(t)) \le \zeta\}$ *is said to be a globally exponentially attractive set of (1), where* $\bar{V}(x(t_0)) \ge V(x(t_0))$ *and* $V(x(t_0))$ *is a constant.*

Definition 2. *CGNNs (1) with globally exponentially attractive set is said to be globally exponentially stable in the Lagrange sense. CGNNs (1) with globally attractive set is said to be ultimately bounded.*

Lemma 1. For any vectors $a, b\in R^n$, the inequality $\pm 2a^Tb \le a^TY^{-1}a+b^TYb$ holds, in which Y is any matrix with $Y>0$.

Proof: Since $Y>0$, we have

$$a^TYa \pm 2a^Tb + b^TY^{-1}b = (Y^{1/2}a \pm Y^{-1/2}b)^T(Y^{1/2}a \pm Y^{-1/2}b) \ge 0.$$

From this, we can easily obtain the above inequality of Lemma 1.

Lemma 2. (Jensen's Inequality [13]) For any constant matrix $P\in R^{n\times n}$, $P^T=P>0, \gamma>0$, vector function $\omega:[0,\gamma]\to R^n$ such that the integrations concerned are well defined, then

$$(\int_0^\gamma \omega(s)ds)^T P(\int_0^\gamma \omega(s)ds) \le \gamma\int_0^\gamma \omega^T(s)P\omega(s)ds.$$

Lemma 3. (Schur Complement [14]) The LMI $\begin{pmatrix} P & R \\ R^T & Q \end{pmatrix} < 0$ with $P^T = P, Q^T = Q$ is equivalent to one of the following conditions: $(1) Q < 0, P - RQ^{-1}R^T < 0;\ \ (2) P < 0, Q - R^T P^{-1} R < 0.$

Lemma 4. (Halanay Inequality [15]) Assume there exist $r_1 > r_2 > 0$ and a nonnegative continuous quantity function $x(t)$, which satisfies $D^+ x(t) \le -r_1 x(t) + r_2 \overline{x}(t),$ for all $t \in [t_0 - h, t_0],$ then $x(t) \le \overline{x}(t_0)\exp(-\lambda(t - t_0))$ holds for $\forall t \ge t_0,$ where $\overline{x}(t) = \sup_{t-h \le s \le t} x(s), h \ge 0,$ and λ is the unique positive root of equation $\lambda = r_1 - r_2 e^{\lambda h}.$

Lemma 5. Given constant matrices $A_1, A_2, A_3, B_1, B_2, B_3 \in R^{n\times n}$, and appropriate reversible matrices X, Y, Z, let

$$\Sigma_1 \triangleq \begin{pmatrix} A_1 \\ B_1 \end{pmatrix} X^{-1} \begin{pmatrix} A_1 \\ B_1 \end{pmatrix}^T + \begin{pmatrix} A_2 \\ B_2 \end{pmatrix} Y^{-1} \begin{pmatrix} A_2 \\ B_2 \end{pmatrix}^T + \begin{pmatrix} A_3 \\ B_3 \end{pmatrix} Z^{-1} \begin{pmatrix} A_3 \\ B_3 \end{pmatrix}^T,$$

$$\Sigma_2 \triangleq \begin{pmatrix} A_1 & A_2 & A_3 \\ B_1 & B_2 & B_3 \end{pmatrix} \begin{pmatrix} X^{-1} & 0 & 0 \\ 0 & Y^{-1} & 0 \\ 0 & 0 & Z^{-1} \end{pmatrix} \begin{pmatrix} A_1 & A_2 & A_3 \\ B_1 & B_2 & B_3 \end{pmatrix}^T.$$

Then $\Sigma_1 = \Sigma_2.$

Proof: Making use of the rule of the product of matrices, we can obtain the results easily. So, the course of relate proof is omitted here.

3 Main Results

Theorem 1. Assume that assumption 1. holds, then the CGNNs system (1) is globally exponentially stability in the Lagrange sense if there exist five positive diagonal matrices P, Q, R, S, T and a positive definite matrix $H \in R^{n\times n}$ such that the following LMIs hold:

$$\begin{pmatrix} \Theta & PA - LQA - DQ & PB - LQB & PC - LQC & P - LQ \\ * & QA + A^T Q - R & QB & QC & Q \\ * & * & -S & 0 & 0 \\ * & * & * & -T & 0 \\ * & * & * & * & -H \end{pmatrix} < 0, \tag{2}$$

$$WSW \le P. \tag{3}$$

Where

$$\Theta = \underline{\alpha}^{-1}(P + Q(F - L)) - PD - DP + 2LQD + W(R + \sigma^2 T)W,$$
$$\underline{\alpha} = \mathrm{diag}\{\alpha_1^-, \ldots, \alpha_n^-\}, \overline{\alpha} = \mathrm{diag}\{\alpha_1^+, \ldots, \alpha_n^+\}, W = \mathrm{diag}\{w_1, \ldots, w_n\},$$
$$w_i = \max\{|L_i|, |F_i|\}, \forall i = 1, \ldots, n. \Omega = \{x \in R^n \mid x^T(t)\overline{\alpha}^{-1}Px(t) \le U^T HU / \varepsilon\}$$

is a globally exponentially attractive set of (1), where $0 < \varepsilon \ll 1$.

Proof: We consider the following radially unbounded and positive definite Lyapunov functional with the given positive definite diagonal matrix $P = \mathrm{diag}\{p_1, \ldots, p_n\}$ and positive diagonal matrix $Q = \mathrm{diag}\{q_1, \ldots, q_n\}$,

$$V(x(t)) = 2\sum_{i=1}^{n} p_i \int_0^{x_i(t)} \frac{s}{\alpha_i(s)} ds + 2\sum_{i=1}^{n} q_i \int_0^{x_i(t)} \frac{1}{\alpha_i(s)} (g_i(s) - l_i s) ds. \tag{4}$$

Calculating the derivative of $V(x(t))$ along the positive semi-trajectory of (1), we can obtain

$$\begin{aligned} \frac{dV(x(t))}{dt}|_{(1)} \le\ & 2(x^T(t)P + g^T(x(t))Q - x^T(t)LQ)(-Dx(t) + Ag(x(t))) + \\ & 2(x^T(t)PB + g^T(x(t))QB - x^T(t)LQB)g(x(t - \tau(t))) + 2(x^T(t)PC + \\ & g^T(x(t))QC - x^T(t)LQC)\int_{t-\sigma(t)}^{t} g(x(s))ds + 2(x^T(t)P + g^T(x(t))Q \\ & - x^T(t)LQ)U. \end{aligned} \tag{5}$$

From assumption 1, for given positive diagonal matrix R we derive

$$\begin{aligned} & 2(x^T(t)P + g^T(x(t))Q - x^T(t)LQ)(-Dx(t) + Ag(x(t))) \le \\ & \begin{pmatrix} x(t) \\ g(x(t)) \end{pmatrix}^T \begin{pmatrix} -PD - DP + 2LQD + WRW & PA - LQA - DQ \\ * & QA + A^T Q - R \end{pmatrix} \begin{pmatrix} x(t) \\ g(x(t)) \end{pmatrix} \end{aligned} \tag{6}$$

By using assumption 1, Lemma 1 and Lemma 2, we know that there exist two positive diagonal matrices S, T and a positive definite matrix H such that the following inequalities hold.

$$\begin{aligned} & 2(x^T(t)PB + g^T(x(t))QB - x^T(t)LQB)g(x(t - \tau(t))) \le \\ & \begin{pmatrix} x(t) \\ g(x(t)) \end{pmatrix}^T \begin{pmatrix} (P - LQ)B \\ QB \end{pmatrix} S^{-1} \begin{pmatrix} (P - LQ)B \\ QB \end{pmatrix}^T \begin{pmatrix} x(t) \\ g(x(t)) \end{pmatrix} \\ & + x^T(t - \tau(t))WSWx(t - \tau(t)), \end{aligned} \tag{7}$$

$$2(x^T(t)PC + g^T(x(t))QC - x^T(t)LQC)\int_{t-\sigma(t)}^{t} g(x(s))ds \le$$
$$\begin{pmatrix} x(t) \\ g(x(t)) \end{pmatrix}^T \begin{pmatrix} (P-LQ)C \\ QC \end{pmatrix} T^{-1} \begin{pmatrix} (P-LQ)C \\ QC \end{pmatrix}^T \begin{pmatrix} x(t) \\ g(x(t)) \end{pmatrix} \tag{8}$$
$$+\sigma^2 x^T(t)WTWx(t),$$

$$2(x^T(t)P + g^T(x(t))Q - x^T(t)LQ)U \le U^T HU +$$
$$\begin{pmatrix} x(t) \\ g(x(t)) \end{pmatrix}^T \begin{pmatrix} P-LQ \\ Q \end{pmatrix} H^{-1} \begin{pmatrix} P-LQ \\ Q \end{pmatrix}^T \begin{pmatrix} x(t) \\ g(x(t)) \end{pmatrix}. \tag{9}$$

Based on Lemma 5 and (5)-(9), we have

$$\frac{dV(x(t))}{dt}|_{(1)} \le \begin{pmatrix} x(t) \\ g(x(t)) \end{pmatrix}^T \Pi \begin{pmatrix} x(t) \\ g(x(t)) \end{pmatrix} + \begin{pmatrix} x(t) \\ g(x(t)) \end{pmatrix}^T \Lambda \begin{pmatrix} x(t) \\ g(x(t)) \end{pmatrix} \tag{10}$$
$$+x^T(t-\tau(t))WSWx(t-\tau(t)) + U^T HU,$$

Where $\Pi = \begin{pmatrix} -PD - DP + 2LQD + W(R+\sigma^2 T)W & PA - LQA - DQ \\ * & QA + A^T Q - R \end{pmatrix}$,

$$\Lambda = \begin{pmatrix} (P-LQ)B & (P-LQ)C & P-LQ \\ QB & QC & Q \end{pmatrix} \begin{pmatrix} S^{-1} & 0 & 0 \\ 0 & T^{-1} & 0 \\ 0 & 0 & H^{-1} \end{pmatrix}$$
$$\begin{pmatrix} (P-LQ)B & (P-LQ)C & P-LQ \\ QB & QC & Q \end{pmatrix}^T.$$

Following from (2), there exists $0 < \varepsilon \ll 1$ such that

$$\begin{pmatrix} \tilde{\Theta} & PA - LQA - DQ & PB - LQB & PC - LQC & P - LQ \\ * & QA + A^T Q - R & QB & QC & Q \\ * & * & -S & 0 & 0 \\ * & * & * & -T & 0 \\ * & * & * & * & -H \end{pmatrix} < 0,$$

Where $\tilde{\Theta} = (1+\varepsilon)\underline{\alpha}^{-1}(P + Q(F-L)) - PD - DP + 2LQD + W(R+\sigma^2 T)W.$

In the light of Lemma 3, one gets $\Pi + \mathrm{diag}\{(1+\varepsilon)\underline{\alpha}^{-1}(\mathrm{P+Q(F-L)}),0\}+\Lambda < 0.$ Therefore, combining (3) and (10), we could derive that

$$\begin{aligned}&\frac{dV(x(t))}{dt}|_{(1)} \le -(1+\varepsilon)x^T(t)\underline{\alpha}^{-1}(P+Q(F-L))x(t)\\&+x^T(t-\tau(t))Px(t-\tau(t))+U^THU,\ t\ge t_0.\end{aligned} \tag{11}$$

From assumption 1 and the formula (4), one has

$$V(x(t)) \le x^T(t)\underline{\alpha}^{-1}(P+Q(F-L))x(t),\ t\ge t_0. \tag{12}$$

According to (11) and (12), we obtain

$$\frac{dV(x(t))}{dt}|_{(1)} \le -(1+\varepsilon)V(x(t))+\bar{V}(x(t))+U^THU,\ t\ge t_0, \tag{13}$$

Where $\bar{V}(x(t)) = \sup_{t-h\le s\le t} V(s).$

On the basis of (13), when $V(x(t))>\eta, \bar{V}(x(t))>\eta,$ one gets

$$\frac{d(V(x(t))-\eta)}{dt}|_{(1)} \le -(1+\varepsilon)(V(x(t))-\eta)+(\bar{V}(x(t))-\eta),\ t\ge t_0,$$

Where $\eta = \frac{U^THU}{\varepsilon}$. According to Lemma 4, we are able to derive $(V(x(t))-\eta) \le \bar{V}((x(t))-\eta)\exp(-\lambda t),$ where λ is the unique positive root of $\lambda = (1+\varepsilon)-e^{\lambda h}$. And judging by [16], it is easy to prove that there exists a constant $\tilde{\beta}$ such that $V(x(t)) \ge \| x \|^{\tilde{\beta}}$. In terms of Definition 1 and noticing $V(x(t)) \ge x^T(t)\bar{\alpha}^{-1}Px(t),$ then $\Omega = \{x\in R^n \mid x^T(t)\bar{\alpha}^{-1}Px(t) \le \frac{U^THU}{\varepsilon}\}$ is a globally exponentially attractive and positive invariant set of system (1). Hence, the CGNNs system (1) is globally exponentially stable in the Lagrange sense via the Definition 2. So, the proof of Theorem 1 is completed.

Corollay 1. Assume that assumptions 1 holds and let $\underline{\alpha} = \mathrm{diag}\{\alpha_1^-,\ldots,\alpha_n^-\}$, $\bar{\alpha}=\mathrm{diag}\{\alpha_1^+,\ldots,\alpha_n^+\}, \sigma(\mathrm{t})=0.$ The CGNNs (1) is globally exponentially stability in the Lagrange sense if there exist four positive diagonal matrices P, Q, R, S and a positive definite matrix $H \in R^{n\times n}$ such that the following LMIs hold:

$$\begin{pmatrix} \Xi & PA-LQA-DQ & PB-LQB & P-LQ \\ * & QA+A^TQ-R & QB & Q \\ * & * & -S & 0 \\ * & * & * & -H \end{pmatrix} < 0, \tag{14}$$

$$WSW \le P. \tag{15}$$

Where $\Xi = \underline{\alpha}^{-1}(P+Q(F-L)) - PD - DP + 2LQD + WRW$, similarly, $W = \mathrm{diag}\{w_1, \ldots, w_n\}, w_i = \max\{|L_i|, |F_i|\}, \forall i = 1, \ldots, n.$ What's more, the set $\Omega = \{x \in R^n \mid x^T(t)\bar{\alpha}^{-1}Px(t) \le U^THU/\varepsilon\}$ is a globally exponentially attractive set of (1), where $0 < \varepsilon \ll 1.$

Proof: The course of proof is almost parallel to that of Theorem 1, except for getting rid of the inequality (8) in the theorem 1. So the process of the proof is omitted in here.

Remark 3. When considered $\alpha(t) \equiv 1$ in corollary 1, its conclusion will turn to the main result of Theorem 1 in [7].

4 Conclusion

In this paper, the Lagrange stability problem for CGNNs with both general activation functions and time-varying and finite distributed delays is investigated. By utilizing a new Lyapunov-Krasovskii functional, the Halanay inequality and Linear matrix inequality technique, a set of novel sufficient conditions are obtained to ensure the globally exponentially stability in the Lagrange sense of CGNNs. Obviously, the results show that globally exponentially attractive set does contribute to the Lagrange stability.

References

1. Cohen, M.A., Grossberg, S.: Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. IEEE Trans. Syst. Man Cybern. 13, 815–826 (1983)
2. Li, J.X., Yan, J.R., Jia, X.C.: Dynamical analysis of Cohen-Grossberg neural networks with time-delays and impulses. Comput. Math. Appl. 58, 1142–1151 (2009)
3. Huang, C.X., Cao, J.D.: Covergence dynamics of stochastic Cohen-Grossberg neural networks with unbounded distributed delays. IEEE Trans. Neural Netw. 22(4), 561–572 (2011)

4. Wang, Z.S., Zhang, H.G.: Global asymptotic stability of reaction-diffusion Cohen-Grossberg neural networks with continuously distributed delays. IEEE Trans. Neural Netw. 21(1), 39–49 (2010)
5. Wang, Z.S., Zhang, H.G., Li, P.: An LMI approach to stability analysis of reaction-diffusion Cohen-Grossberg neural networks concerning dirichlet boundary conditions and distributed delays. IEEE Trans. Syst. Man Cyber. 40(6), 1596–1606 (2010)
6. Li, X.D., Fu, X.L., Balasubramaniamc, P., et al.: Existence, uniqueness and stability analysis of recurrent neural networks with time delay in the leakage term under impulsive perturbations. Nonlinear Anal. RWA 11, 4092–4108 (2010)
7. Song, Q.K.: Exponential stability of recurrent neural networks with both time-varying delays and general activation functions via LMI approach. Neurocomputing 71, 2823–2830 (2008)
8. Yu, P., Liao, X.X., Xie, S.L., Fu, Y.L.: A constructive proof on the existence of globally exponentially attractive set and positive invariant set of general Lorenz family. Commun. Nonlinear Sci. Numer. Simulat. 14(7), 2886–2896 (2009)
9. Liao, X.X., Luo, Q., Zeng, Z.G., Guo, Y.X.: Global exponential stability in Lagrange sense for recurrent neural networks with time delays. Nonlinear Anal. RWA 9, 1535–1557 (2008)
10. Wang, X.H., Jiang, M.H., Fang, S.L.: Stability analysis in Lagrange sense for a non-autonomous Cohen-Grossberg neural network with mixed delays. Nonlinear Anal. TMA 70, 4294–4306 (2009)
11. Luo, Q., Zeng, Z.G., Liao, X.X.: Global exponential stability in Lagrange sense for neutral type recurrent neural networks. Neurocomputing 74, 638–645 (2011)
12. Liao, X.X., Luo, Q., Zeng, Z.G.: Positive invariant and global exponential attractive sets of neural networks with time-varying delays. Neurocomputing 71, 513–518 (2008)
13. Gu, K., Kharitonov, V., Chen, J.: Stability of time-delay systems. Birkhauser, Boston (2003)
14. Boyd, B., Ghoui, L., Feron, E., Balakrishnan, V.: Linear Matrix Inequalities in System and Control Theory. SIAM, Philadelphia (1994)
15. Halanay, A.: Differential Equations: Stability, Oscillations, Time Lags. Academic Press, New York (1966)
16. Liao, X., Fu, Y., Xie, S.: Globally exponential stability of Hopfied networks. Adv. Syst. Sci. Appl. 5, 533–545 (2005)

Research of Triple Inverted Pendulum Based on Neural Network of Genetic Algorithm

Xiaoping Huang[1],Ying Zhang[1], and Junlong Zheng[2]

[1] Technology Insitute of Yongjiang University, Nanning, 530200, China
[2] Guangxi Electric Power Institute of Vocational Training, Nanning, 530007, China
hxpgx@163.com

Abstract. In the topic, I carried out the simulation of triple inverted pendulum system by intelligent control, which was combinated of neural network and genetic algorithm. Neural network oriented genetic algorithm was put forward and discussed its realization, it overcomed the shortcoming of slow convergent speed, immature convergence and lots of iterations.

Keywords: Triple inverted pendulum, Neural network(NN), Genetic Algorithm(GA).

1 The Model of Triple Inverted Pendulum

The triple inverted pendulum system was made up of control object, guide, motor, pulley, belts and electrical measurement device. The Controlled object was composed by dolly, pendulum 1, pendulum 2 and pendulum 3. Pendulum 1, pendulum 2 and pendulum 3 were connected by bearing, and the bearing can be rotated freely in vertical plane of the parallel guide. Three potentiometers were installed in joint respectively to measure the relative drift angle $\theta_3, -\theta_2, \theta_2, -\theta_1, \theta_1$.

Assumptions: The upper, the medium, the hem and the small car are all rigid bodies. There was no relative sliding between belt pulleys and driving band. There was no elongation phenomenon for the transmission belt. It was direct ratio between the car driving force and the input of direct current amplifier, and it has no lag. Ignored the armature inductance of motor, the car movement from the friction was directly proportional to the speed of the car, the pendulum of the friction torque and the relative speed (angular velocity) becomed direct ratio. The motion analysis of the system diagram was shown in Fig.1.

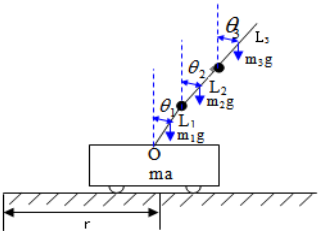


Fig. 1. The triple inverted pendulum system

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 437–443, 2012.

Set Lagrange function: $L = E_k - E_p$, according to the Euler-Lagrange equation:

$$\frac{d}{dt}\left[\frac{\partial L}{\partial \dot{q}_k}\right] - \frac{\partial L}{\partial q_k} + \frac{\partial_{ED}}{\partial \dot{q}_k} = Q_k$$

Among them: Q_k—Generalized force, q_k—Generalized coordinates. Then we can obtain the nonlinear equations of triple inverted pendulum system as follows:

$$M(\theta_1,\theta_2,\theta_3)\begin{bmatrix}\ddot{\gamma}\\ \ddot{\theta}_1\\ \ddot{\theta}_2\\ \ddot{\theta}_3\end{bmatrix} = F(\theta_1,\theta_2,\theta_3,\dot{\theta}_1,\dot{\theta}_2,\dot{\theta}_3)\begin{bmatrix}\dot{\gamma}_1\\ \dot{\theta}_1\\ \dot{\theta}_2\\ \dot{\theta}_3\end{bmatrix} + N(\theta_1,\theta_2,\theta_3) + \begin{bmatrix}G_0U\\ 0\\ 0\\ 0\end{bmatrix}$$

Among the equation:

$$M(\theta_1,\theta_2,\theta_3) = \begin{bmatrix} m_0+m_1+m_2+m_3 & m_1d_1\cos\theta_1 + m_2d_4\cos\theta_1 + m_3d_4\cos\theta_1 \\ m_1d_1\cos\theta_1 + m_2d_4\cos\theta_1 + m_3d_4\cos\theta_1 & J_1 + m_1d_1^{\,2} + m_2d_4^{\,2} + m_3d_4^{\,2} \\ m_2d_2\cos\theta_2 + m_3d_5\cos\theta_2 & m_2d_2d_4\cos(\theta_2-\theta_1) + m_3d_4d_5\cos(\theta_2-\theta_1) \\ m_3d_3\cos\theta_3 & m_3d_4d_5\cos(\theta_3-\theta_1) \end{bmatrix}$$

$$\begin{bmatrix} m_2d_2\cos\theta_2 + m_3d_3\cos\theta_2 & m_3d_3\cos\theta_3 \\ m_2d_2d_4\cos(\theta_2-\theta_1) + m_3d_4d_5\cos\cos(\theta_2-\theta_1) & m_3d_3d_4\cos(\theta_3-\theta_1) \\ J_2 + m_2d_2^{\,2} + m_3d_5^{\,2} & m_3d_3d_5\cos(\theta_3-\theta_2) \\ m_3d_3d_5\cos(\theta_3-\theta_2) & J_3 + m_3d_3^{\,2} \end{bmatrix}$$

$$F(\theta_1,\theta_2,\theta_3,\dot{\theta}_1,\dot{\theta}_2,\dot{\theta}_3) = \begin{bmatrix} -f_0 & (m_1d_1\sin\theta_1 + m_2d_4\sin\theta_1 + m_3d_4\sin\theta_1)\bullet\dot{\theta}_1 \\ 0 & -(f_1+f_2) \\ 0 & f_2 - m_2d_2d_4\dot{\theta}_1\sin(\theta_2-\theta_1) + m_3d_4d_5\dot{\theta}_1\sin(\theta_2-\theta_1) \\ 0 & -m_3d_3d_4\dot{\theta}_1\sin(\theta_3-\theta_1) \end{bmatrix}$$

$$\begin{bmatrix} (m_2d_2\sin\theta_2 + m_2d_3\sin\theta_2)\dot{\theta}_2 & m_3d_3\sin\theta_3\bullet\dot{\theta}_3 \\ f_2 + (m_3d_3d_4 + m_2d_4d_5)\dot{\theta}_2\sin(\theta_2-\theta_1) & m_3d_3d_4\dot{\theta}_3\sin(\theta_3-\theta_1) \\ -(f_2+f_3) & f_3 + m_3d_3d_5\dot{\theta}_3\sin(\theta_3-\theta_2) \\ f_3 - m_3d_3d_5\dot{\theta}_2\sin(\theta_3-\theta_2) & -f_3 \end{bmatrix}$$

$$N(\theta_1,\theta_2,\theta_3) = \begin{bmatrix} 0 \\ (m_1d_1 + m_2d_4 + m_3d_4)g\sin\theta_1 \\ (m_2d_2 + m_3d_3)g\sin\theta_2 \\ m3d3g\sin\theta3 \end{bmatrix}$$

2 The Control Simulation of the Inverted Triple Pendulum

Selected the S-function(System-function) of SIMULINK in MATLAB to realize the simulation, the S-function included three parts: mdlInitializeSizes, mdlDerivatives and mdlOutputs. Then we could obtain the results as below.

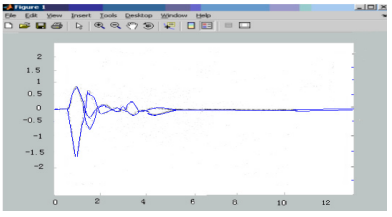


Fig. 2. The state response of $r, \theta_1, \theta_2, \theta_3$

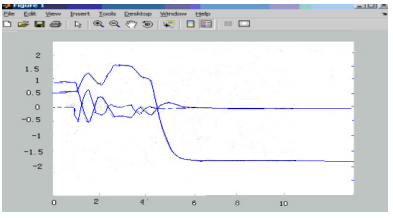


Fig. 3. The state response of $\dot{r}, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3$

3 Used Genetic-Neural Network to Realize the Control of the Inverted Triple Pendulum

Used the genetic algorithm that based on the neural network to train the neural network weights, to design the genetic neural network controller of the inverted pendulum. I realized this algorithm of data structure in the C language, program diagram and part of the important function. in the end.

3.1 The Steps of the Genetic Algorithm That Faces to Neural Network

① Structured the chromosomes to meet the constraint condition. Because the genetic algorithm (GA) could not directly deal with the solution of space. So we expressed the solution of appropriate chromosome through the code. There had a variety of encoding of chromosomes of the actual problem, the selection of chromosome should be meet problem constraint as far as possible, otherwise it would affect the computing efficiency.

② Determined the control parameters of genetic algorithm(GA), such as the number of chromosomes, chromosome length, etc.

③ Generated the initial population by random.The initial population was to search a group chromosome of the initiatory, the amount should be appropriate,we should make the initial population of individuals in the solution space on evenly distributed, in order to search the optimal solution rapidly.

④ Calculated the fitness of each chromosome. Fitness was the only index to reflect the fulu of the chromosome.GA was used to find the biggest fitness chromosomes.

⑤ Used duplicate, crossover and mutation calculate to generate the son group. The three operators were the basic operator of GA, the copy reflected the laws of nature survival for the fittest.Cross embodied the thought of sexual reproduction, variation reflected the gene mutations in the process of evolution. copy, crossover and mutation were used to the method of this section.

⑥ Repeated the“step ③” ,the“step ④” and the“step ⑤” ,until it fitted the termination conditions.

3.2 The Neural Network（NN）Controller of Triple Inverted Pendulum

In the control system, I appointed the level state variables of triple inverted pendulum:

$$x = (r, \theta_1, \theta_2, \theta_3, \dot{r}, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3)$$

Taked sample, p was the input vector,it should be contained all the input value of the maximum and minimum value, here was for:

p = [-0.2 0.3;-0.57 0.57;-0.57 0.57;-0.57 0.57;-2 3;-3.14 3.14;-3.14 3.14;-3.14 3.14]

t was the objective vector,and t=Kx, K was the feedback gain. Designed a three layers network of BP neural,then trained it.

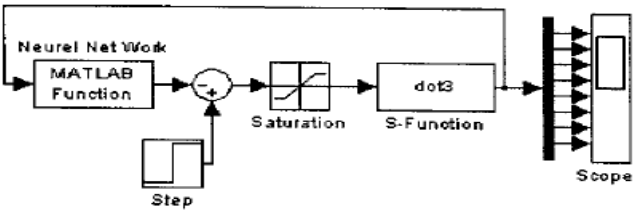


Fig. 4. The structre of NN controller of the triple inverted pendulum

The network layer was made up of the input, the hidden layer and output layers. The number of the input layer nodes was 8, I/0 function was tansig function. The number of the nodes hidden layer was 8, I/0 function was tansig function. There was only one node in output layer, its I/0 function was purelin function. The following datum were the training program of Back-Propagation network:

```
p = [-0.2 0.3;-0.57 0.57;-0.57 0.57;-0.57 0.57;-2 3;-3.14 3.14;-3.14 3.14;-3.14 3.14];
k = [-6.3246 -72.1652 -7.8966 -272.8343 -11.6811 -34.1877 -30.3206 -37.8689];
t = k * x;
[w1, b1, w2, b2, w3, b3] = initff (p,8,'tan sig',4,'tan sin g',1,' purelin ')

disp_fgre=l ;          % indicates that frequency in training process
max_epoch=1000 ;   % the maximum number of training
err_goal=0.0001 ;    % Error index
tp=[disp_fgre max_epoch err_goal]

[w1, b1, w2, b2, w3, b3, ep, tr] = trainlm(w 1, b 1,'tansig' , w2, b2,'tansig' , w3, b3,'purelin' , x, t, tp)
```

3.3 The Diagram of Training NN Weights with Neural Network of Genetic Algorithm (NNOGA)

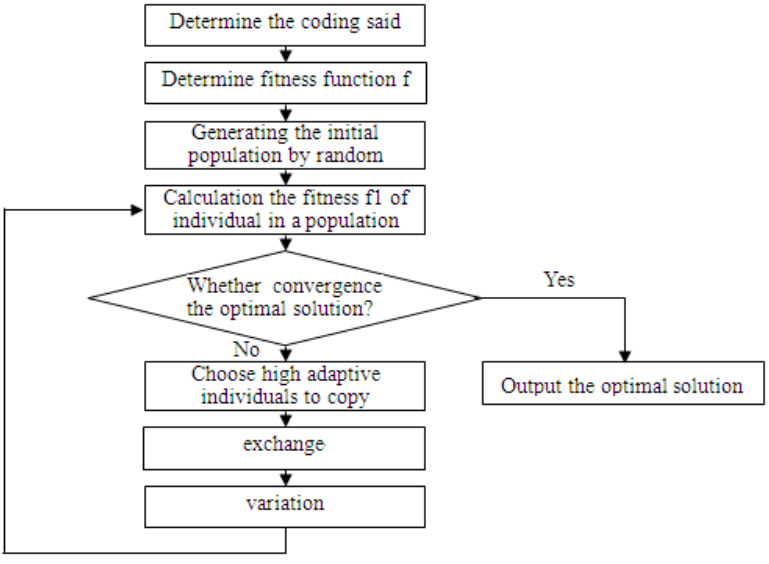


Fig. 5. The basic flow chart of NNOGA

3.4 Trained the Data Structures of NN Weights with NNOGA

This mainly data structure of procedure is the population(population).

```
# define POPULATION_SIZE 30          /*The number of group */
# define WEIGHT_NUM 31               /*Connection power number*/
# define SAMPLE_NUM 500              /*Sample logarithm*/
# define FIRST 4                     /*The points of NN input layer */
# define SECOND 5                    /*The points of NN hidden layer*/
# define CHROM_LENGTH 31             /*Length of string*/
# define K 1.5                       /*Adjustment factor of adaptive value
                                       for string length*/
# define P 0.8                       /*Selectional parameters*/
FILE *fp ;
int generation ;                     /*The number of producing offspring */
int selected[POPULATION_SIZE] ;
struct population
{
   double value[WEIGHT_NUM] ;        /*Network weights*/
   double string[CHROM_LENGTH] ;     /*The string of Chromosome*/
   double fitness ;                  /*Adaptive value*/
 }
```

3.5 System MATLAB Simulation of NN Control

Used the mathematical model that had been built foward, we carried out the simulation of the triple inverted pendulum neural network control system in figure 6. Get the response of the system state variables as Fig.6.

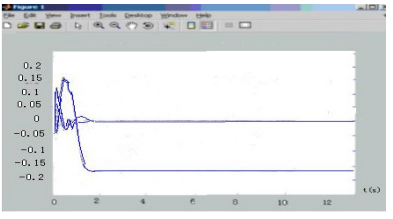


Fig.6. The state response of $r,\theta_1,\theta_2,\theta_3$

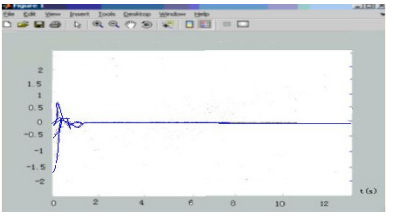


Fig.7. The state response of $\dot{r},\dot{\theta}_1,\dot{\theta}_2,\dot{\theta}_3$

In the above image of Fig.6, '—'、'——'、'—.'、'——'respectively stands on the state response curve of $r,\theta_1,\theta_2,\theta_3$, the dooly was nearby in the balance position, swinging rod angle tends to 0 degree.

In the above image of Fig.7, '—'、'——'、'—.'、'——'respectively standed on the state response curve of $\dot{r},\dot{\theta}_1,\dot{\theta}_2,\dot{\theta}_3$, the small car translation speed and swinging rod speed were tend to be 0.

4 Conclusion

Inverted pendulum device was typical nonlinear object.I used the improved genetic algorithm-facing neural network of genetic algorithm (NNOGA) to train network weights, and realize the control of the inverted pendulum. This algorithm was based on the global scope search, it could avoid the local minimum faults that used the traditional BP algorithm. In addition, the algorithm demanded little required for the objective function and constraints, it could complete training of the multilayer neural network for the non- excitation funceion of nerve. From results of the experiment, the control effect of the inverted pendulum was good.

Innovation points: Put forward neural network to the genetic algorithm (NNOGA), and realized the control to the triple inverted pendulum with the implementation of algorithm.

References

1. Huang, X., Niu, Q.: The Realization of Second-order Linear Optimal Control in the Inverted Pendulum System. Computer Measurement and Control 14(12), 1641–1642, 1662 (2006)
2. Hou, T., Dong, H.: Designs and simulations of double closed-loopcascad fuzzy control for inerted pendulum based on T-S model. Journal of System Simulation 19(11), 2477–2479, 2526 (2007)

3. Duan, X., Qiu, Y., Duan, B.: Adaptive sliding mode fuzzy control of planar inverted pendulum. Control and Decision 22(7), 774–777 (2007)
4. Lin, F., Zhao, Y.Q., Jiang, H.: Simulatin of neural network control strategy for four wheel steering vehicle based on Simulink. Journal of JianRsu University 29(5), 390–393 (2008)
5. Metni, N.: Neuro-Control of an Inverted Pendulum using Genetic Algorithm. In: IEEE Conf. of ACTEA 2009, pp. 27–33 (July 2009)

Evolving Neural Network Using Hybrid Genetic Algorithm and Simulated Annealing for Rainfall-Runoff Forecasting

Hong Ding[1,2], Jiansheng Wu[3], and Xianghui Li[4]

[1] School of Information Engineering, Wuhan University of Technology
Wuhan, 430070, Hubei, P.R. China
micro_ding@tom.com
[2] Department of Physics and Information Science, Liuzhou Teachers College
Guangxi, Liuzhou, 545004, P.R. China
[3] Department of Mathematics and Computer, Liuzhou Teachers College
Liuzhou, 545004, Guangxi, P.R. China
wjsh2002168@163.com
[4] Liuzhou City Flood Control and Drainage Project Management Office
Liuzhou, 545002, Guangxi, P.R. China
2402908211@qq.com

Abstract. Accurately rainfall–runoff forecasting modeling is a challenging task. Recent neural network (NN) has provided an alternative approach for developing rainfall–runoff forecasting model, which performed a nonlinear mapping between inputs and outputs. In this paper, an effective hybrid optimization strategy by incorporating the jumping property of simulated annealing (SA) into Genetic Algorithm (GA), namely GASA, is used to train and optimize the network architecture and connection weights of neural networks for rainfall–runoff forecasting in a catchment located Liujiang River, which is a watershed from Guangxi of China. This new algorithm incorporates metropolis acceptance criterion into crossover operator, which could maintain the good characteristics of the previous generation and reduce the disruptive effects of genetic operators. The results indicated that compared with pure NN, the GASA algorithm increased the diversity of the individuals, accelerated the evolution process and avoided sinking into the local optimal solution early. Results obtained were compared with existent bibliography, showing an improvement over the published methods for rainfall–runoff prediction.

Keywords: Genetic Algorithm, Simulated Annealing, Neural Network, Rainfall–runoff, Forecasting.

1 Introduction

Forecasting rainfall–runoff process is always a especially difficult task in simulation of the whole hydrological cycle, because the rainfall–runoff relationship is one of the most complex hydrologic phenomena to comprehend due to the tremendous spatial and temporal variability of watershed characteristics and

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 444–451, 2012.

precipitation patterns, and the number of variables involved in the modeling of the physical processes [1]. In recent years, neural network (NN) have been successfully applied in hydrological forecasting [2] [3], mainly because of NN's ability of nonlinear mapping mechanisms. As neural network approaches want of a rigorous theoretical support, effects of applications are strongly depend upon operator's experience. If carelessly used, it can easily learn irrelevant information (noises) in the system (over-fitting). In the practical application, researchers determine appropriate net- work architecture and values of different parameters with trial and error due to short of prior knowledge [4] [5].

Recently some investigations into neural network training using genetic algorithm (GA) have been successfully employed to overcome the inherent limitations of the NN [6] [7]. Genetic algorithms have been used with neural network to search for input variables or to determine the number of nodes or connections in the network. Because GA search from not only a single point but a large population of points, many researchers have actively exploited the combining of multiple NN's which have evolved in the last generation. However, these NN's tend to be too similar to each other because in each case, the individual with the highest fitness has prevailed even after certain generations. This phenomenon is known as premature convergence towards a local minimum [8].

Different from the previous work in this paper, one of the main purposes is to develop an effective hybrid optimization strategy by incorporating Simulated Annealing (SA) into Genetic Algorithm (GA) for NN training, namely NN–GASA, to overcome the weaknesses of GA and to avoid premature convergence towards a local minimum. GASA is used to train and optimize the network architecture and connection weights of neural networks for rainfall–runoff forecasting in a catchment located Liujiang River, which is a watershed from Guangxi of China. The rainfall–runoff data of Liuzhou in Guangxi is predicted as a case study for our proposed method. An actual case of forecasting monthly runoff is illustrated the improvement in predictive accuracy and capability of generalization achieved.

The organization of the paper is as follows. Section 2 describes the proposed GASA, ideas and procedures. For further illustration, this work employs the method set up a prediction model for rainfall-runoff forecasting in Section 3. Discussions are presented in Section 4 and conclusions are drawn in the final Section.

2 Methodology

2.1 Genetic Algorithm and Simulated Annealing Algorithm

Genetic Algorithm is heuristic optimization algorithms based on principles inspired from the genetic and evolution mechanisms observed in natural systems and populations of living beings [9] [10]. The algorithm mimics the evolution of a population of computer representations of the solutions by iteratively applying genetic operators, such as recombination and mutation, to the solutions that have the highest fitness in the population. Their basic principle is the maintenance of

a population of solutions to a problem (genotypes) as encoded information individuals that evolve in time. Readers interested in a more detailed introduction about GA implementation are referred to the related literature [11].

Simulated Annealing (SA) was a metanephritic which has been considered a good tool for complex nonlinear optimization problems [12] [13]. The basic idea of the standard SA is that it tries to avoid being trapped in local minima by making uphill move with the probability $p = exp(-\Delta E/T)$, where ΔE is the amount of increase in the objective value caused by the uphill move and T is a parameter referred to as "annealing temperature". To avoid accepting large uphill move in the later stage of the search, the parameter T could be decreased over time by a schedule which is called "the cooling schedule". More detailed introduction about SA algorithm are referred to the related literature [14].

2.2 Evolving Neural Network Using Hybrid GASA

The most widely used neural network model is the multi–layer perception (MLP), in which the connection weight training is normally completed by a back–propagation (BP) learning algorithm based on gradient descent. In addition, lots of issues are based on n-dimensional curve surfaces, which makes BP algorithm to converge slowly and fall in a local minima easily [15]. The network used in this paper consists of an input layer, one hidden layer, and an output layer as shown in Fig. 1.

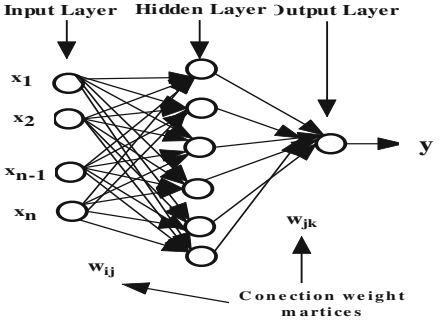


Fig. 1. Architecture of the neural network model used in this paper

Referring to Fig. 1, each neuron in the network computers by using the sum of its weighted inputs and passing the result through a nonlinear activation function (transfer function). In this paper the tangent function is used to transfer the values of the input layer nodes to the hidden layer nodes, whereas the linear transfer function is adopted to transfer the values from the hidden layer to the output layer. Each hidden neuron's output is calculated using Eq. (1), while the output neurons output is calculated using Eq. (2)

$$X_i = tanh(\sum_{i=1}^{n} x_i \omega_{ij} + \omega_{j0}) \tag{1}$$

$$\hat{y}_i = \sum_{i=1}^{m} X_j \omega_{jk} + \omega_{k0} \tag{2}$$

This paper evolving neural networks by using GASA consists of two major phases: (i) using global searching ability of GASA to find an appropriate network architecture and connection weights; (ii) using BP algorithm to search peak value(s) in detail; Mathematically, optimization problems of GASA-neural network can be described as follows:

$$\begin{cases} \min E(\omega) = \frac{1}{n} \sum_{i=1}^{n} [y_i - \hat{y}_i]^2 \\ \hat{y}_i = \sum_{i=1}^{m} X_j \omega_{jk} + \omega_{k0} \\ X_i = tanh(\sum_{i=1}^{n} x_i \omega_{ij} + \omega_{j0}) \end{cases} \tag{3}$$

The fitness function is defined as follows:

$$f(\omega) = 1/(1 + minE(\omega)) \tag{4}$$

The major steps of the proposed algorithm are as follows:

1. Initialize the variables of GA and SA. The hidden nodes are encoded as binary code string, 1 with connection and 0 without connection. The connection weights are encoded as float string, randomly generated within $[-1, 1]$.
2. Creating an initial population by randomly generating a set of feasible solutions (chromosomes).
3. Computing each chromosome by running the load runoff data.
4. Apply the crossover operator, the child chromosomes are accepted according to eq. 8,

$$P = \begin{cases} 1 & if \quad f^* \leq f \\ exp(\frac{f - f^*}{T}) & if \quad f^* > f \end{cases} \tag{5}$$

   where f^* and f are the fitness values of parent and child chromosomes, respectively, T is the temperature.
5. Apply the mutation operator to the new population.
6. Let the current population be the new population.
7. If the convergence criterion is satisfied, stop. Otherwise go to step 3. Obtain the appropriate network architecture and connection weights.
8. Apply the Back–Propagation Algorithm training Neural Network.

Fig.2 shows flowchart of the proposed algorithm. Selection of chromosomes for applying various GA operators is based on their scaled fitness function in accordance to the roulette wheel selection rule. The roulette wheel slots are sized according to the accumulated probabilities of reproducing each chromosome. Crossover and mutation operators are carried with the pre-specified probabilities, but the resultant chromosomes of crossover operator are accepted according to step5 in the proposed algorithm steps.

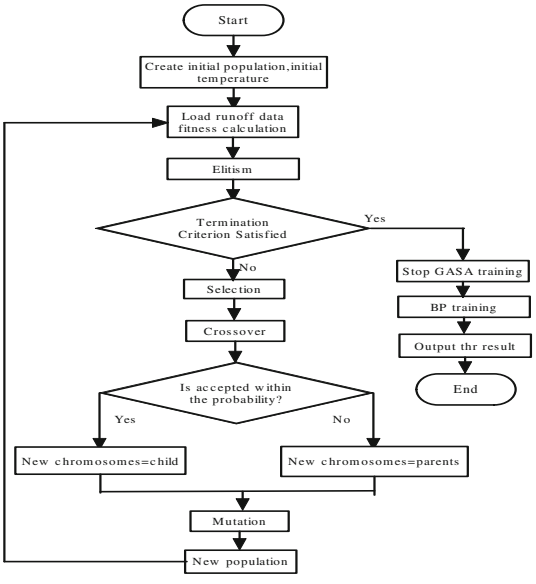


Fig. 2. Flowchart of algorithm

3 Experimental Results and Discussion

3.1 Study Area and Data

Data used in this paper are from the Liujinag basin, located in in the southwest of China and is the most important sub–catchment of Xijiang basin drainage. The data contains information for a period of 68 years (1941–2009). Furthermore, data from 1941 to 2004 constitute the training set and the 5 remaining data is used in the testing phase.

3.2 Evaluation of Model Performances

Three different types of standard statistical performance evaluation criteria were employed to evaluate the performance of various models developed in this paper. These are average absolute relative error (AARE), root mean square error (RMSE), and the correlation coefficient (CC)which be found in many paper [16].

For the purpose of comparison, we also built the other two prediction models for fitting and forecasting, such as Simple Moving Average (SMA) model and Autoregressive Integrated Moving Average (ARIMA) model. Simple moving average model could be expressed as

$$\hat{y}_{t+1} = \frac{1}{N}(y_t + \cdots + y_{t-N+1}), \quad t = N, N+1, \cdots, T \tag{6}$$

where $\hat{y}_{t+1}$ is prediction valuer, y_T are observation sequence, N is the number of moving average terms, $N < T$. The valuer of N is 5 by the least sum of square

error. Through calculating autocorrelation function, partial correlation function and AIC, The ARIMA model is as follows:

$$-BX_t = (1 + 0.8732B - 0.26B^2)\varepsilon_t \tag{7}$$

3.3 Analysis of the Results

In the process of determining model inputs, the nine lags was chosen as the NN–GASA models inputs by trying different lags with the best performance. This paper GA parameters are set as follows: the iteration times are 1000, the crossover probability is 0.9 and mutation probability is 0.05. Fig.3 shows the curve of fitness of training NN in the learning stage for GASA approach arising from the generation number. One can see that the maximum, average and the minimum fitness are tending towards stability with increase of iteration number.

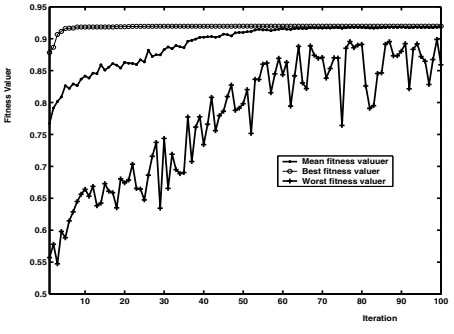


Fig. 3. Fitness in GASA approach

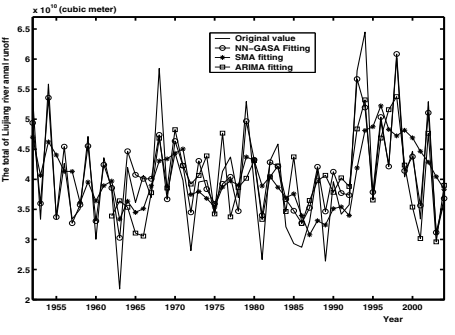


Fig. 4. Comparison predicted results

Fig.4 shows the comparison between predicted and measured runoff values at training and testing phases by three different models model using the maximum annual runoff data from the Liuzhou catchment. In Fig.4 the output of the NN–GASA model, simulated with test data, shows a good agreement with the target.

For the total of annal runoff for Liujing river forecasting, the performance of three model is summarized in Table 1. From the graphs and table, we can generally see that the forecasting results are very promising in the rainfall–runoff forecasting under the research where either the measurement of fitting performance is goodness of fit, such as AARE, RMSE and PRC, where the forecasting performance performance.

In this paper, a new forecast method NN–GASA, by us is proposed. The NN–GASA method adopts neural network combined with a goal-search genetic algorithm and simulated annealing and is found to be able to exploit all the origin methods' advantages. The proposed method is next applied to runoff forecasting, as an example test, in an actual Liujiang river and demonstrates an encouraging degree of accuracy superior to other commonly used forecasting methods in

Table 1. Performance statistics of the three models for runoff fitting and forecasting

Model	Training data			Testing data		
	NN–GASA	ARIMA	SMA	NN–GASA	ARIMA	SMA
AARE	0.085	0.149	0.160	0.085	0.123	0.102
RMSE	41.746	69.521	75.843	47.091	51.240	53.953
PRC	0.926	0.661	0.575	0.806	0.632	0.456

several time-period cases reported in this paper. The forecasting results are tabulated and partially converted into figure for evaluation and comparisons, the results indicate the NN–GASA is a very efficient and robust model.

4 Conclusion

In this paper, the advantages and the key issues of the Genetic Algorithm and Simulated Annealing evolved Neural Network has been presented to model the rainfallCrunoff relationship in Liujiang catchment. Our methodology adopts a real coded GASA strategy and hybrid with back-propagation algorithm. The GASA operators are carefully designed to optimize the neural network, avoiding problems premature convergence , permutation and escaping from local optima. The experiment with real rainfall–runoff data have showed that the predictive performance of the proposed model is better than that of the traditional model. So the NN–GASA ensemble forecasting model can be used as an alternative tool for monthly rainfall forecasting to obtain greater forecasting accuracy and improve the prediction quality further in view of empirical results, and can provide more useful information, avoid invalid information for the future forecasting.

Acknowledgment. The authors would like to express their sincere thanks to the editor and anonymous reviewers comments and suggestions for the improvement of this paper. This work was supported by Program for Excellent Talents in Guangxi Higher Education Institutions, by Natural Science Foundation of Guangxi under Grant No. 2011GXNSFE018006 and by the Natural Science Foundation of China under Grant No.11161029.

References

1. Pan, T.Y., Wang, R.Y.: State Space Neural Networks for Short Term Rainfall–runoff Forecasting. Journal of Hydrology 297, 34–50 (2004)
2. Wu, J., Jin, L.: Study on the Meteorological Prediction Model Using the Learning Algorithm of Neural Networks Ensemble Based on PSO agorithm. Journal of Tropical Meteorology 15(1), 83–88 (2009)
3. French, M.N., Krajewski, W.F., Cuykendall, R.R.: Rainfall Forecasting in Space and Time Using Neural Network. Journal of Hydrology 137, 1–31 (1992)

4. Wu, J., Chen, E.: A Novel Nonparametric Regression Ensemble for Rainfall Forecasting Using Particle Swarm Optimization Technique Coupled with Artificial Neural Network. In: Yu, W., He, H., Zhang, N. (eds.) ISNN 2009, Part III. LNCS, vol. 5553, pp. 49–58. Springer, Heidelberg (2009)
5. Wu, J.: An Effective Hybrid Semi-Parametric Regression Strategy for Rainfall Forecasting Combining Linear and Nonlinear Regression. International Journal of Applied Evolutionary Computation 2(4), 50–65 (2011)
6. Hansen, J.V., McDonald, J.B., Nelson, R.D.: Time Series Prediction with Genetic Algorithm Designed Neural Networks: An Experimental Comparison with Modern statistical Models. Computational Intelligence 15(3), 171–184 (1999)
7. Wu, J.: A Semiparametric Regression Ensemble Model for Rainfall Forecasting Based on RBF Neural Network. In: Wang, F.L., Deng, H., Gao, Y., Lei, J. (eds.) AICI 2010, Part II. LNCS, vol. 6320, pp. 284–292. Springer, Heidelberg (2010)
8. Rogers, A., Prü-Bennett, A.: Genetic Drift in Genetic Algorithm Selection Schemes. IEEE Transaction Evoling of Computation 3(4), 298–303 (1999)
9. Holland, J.H.: Adaptation in Natural andArtificial Systems. University of Michigan Press, MI (1975)
10. Mitchell, M.: An Introduction to GeneticAlgorithms. MIT Press, Cambridge (1996)
11. Goldberg, D.: Genetic Algorithms in Search, Optimization and Machine Learning, pp. 1–25. Addison-Wesley, Reading (1989)
12. Eglese, R.W.: Simulated annealing: A Tool for Operation Research. European Journal of Operational Research 46, 271–281 (1990)
13. Anagnostopoulos, A., Michel, L., Van Hentenryck, P.: A Simulated Annealing Approach to The Traveling Tournament Problem, doi: 10.1007/s10951-006-7187-8
14. Laarhoven, P.J., Aarts, E.H.: Simulated Annealing: Theory and Applications. D. Reidel Publishing Company, Dordrecht (1987)
15. Wu, J., Jin, L., Liu, M.: Modeling Meteorological Prediction Using Particle Swarm Optimization and Neural Network Ensemble. In: Wang, J., Yi, Z., Żurada, J.M., Lu, B.-L., Yin, H. (eds.) ISNN 2006, Part III. LNCS, vol. 3973, pp. 1202–1209. Springer, Heidelberg (2006)
16. Wu, J.: An Effective Hybrid Semi-Parametric Regression Strategy for Rainfall Forecasting Combining Linear and Nonlinear Regression. International Journal of Applied Evolutionary Computation 2(4), 50–65 (2011)

Multistep Fuzzy Classifier Forming with Cooperative-Competitive Coevolutionary Algorithm

Roman Sergienko and Eugene Semenkin

Siberian State Aerospace University, Departement of System Analysis and Operation Research,
Krasnoyarskiy Rabochiy Avenue 31,
660037 Krasnoyarsk, Russian Federation
romaserg@list.ru, eugenesemenkin@yandex.ru

Abstract. This paper is about multistep fuzzy classifier forming method with cooperative-competitive coevolutionary algorithm. Cooperative-competitive coevolutionary algorithm automatically allows avoiding the problem of genetic algorithm parameters setting. This approach is included in a new method combining Michigan and Pittsburgh approaches for fuzzy classifier design. The procedure is performed several times. After each step classification efficiency is increased and standard deviation of values is decreased. Results of numerical experiments for machine learning problems from UCI repository are presented.

Keywords: fuzzy classifier, coevolutionary algorithm, classification.

1 Introduction

A fuzzy classifier [1] is a classification algorithm based on fuzzy rules extraction from numerical data. Superiority of this method upon other classification algorithms (for example, neural networks) is provided by fuzzy rules which are linguistic expressions and are available for humans understanding. Thus fuzzy classifier is one of the data mining methods for knowledge discovery.

Fuzzy classifier forming can be considered as an optimization problem. In this case we need to find the best fuzzy classifier. Fuzzy classifier forming includes two problems. The first one is a rule base generating and the second one is membership functions tuning. It should be noted that the first problem is more sophisticated due to huge dimension and discrete variables. So in this paper we present only fuzzy rule base generating problem.

As fuzzy rule base generating is a complicated computational problem, the popular method of its solving is genetic-based machine learning [2, 3]. There are two basic ways for genetic algorithm applying to get fuzzy rule base: Michigan-style and Pittsburgh-style. In Michigan approach [3] chromosomes are individual rules; and a rule set is represented by the entire population. In Pittsburg method [3] chromosomes are rule sets at whole. The problem in the Michigan approach is the conflict between individual rule fitness and performance of fuzzy rule set. Pittsburgh-style systems require a lot of computational efforts. So a combination of Michigan and Pittsburgh methods is a promising approach. In [4] the hybridization of both approaches by using Michigan method as a mutation operator in Pittsburgh-style algorithm is presented.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 452–459, 2012.

A new method of Michigan and Pittsburgh approaches combination for fuzzy classifier rule base design with coevolutionary algorithms was developed in [5]. Fuzzy classifier rule base design consists of two main stages excepting initial population of fuzzy rules forming using a-priori information from a learning sample. At the first stage Michigan method is used for fuzzy rules search with high grade of certainty. At the second stage Pittsburgh method is applied for searching subset of rules with good performance and given number of rules. Constraint for number of rules is used at the second stage of fuzzy classifier generating. This method requires less computational efforts than multiobjective optimization for fuzzy rules extraction. Besides this method has some advantages that were showed by numerical experiments.

Another problem with genetic algorithm applying is the algorithm parameters setting. This problem is especially essential for optimization problems with high computational complexity such as fuzzy rule base generating. There are some methods for GA parameter setting. We suggest special procedure named cooperative-competitive coevolutionary algorithm for this problem solving [6]. This method combines ideas of cooperation and competition among subpopulations in the coevolutionary algorithm. We have tested this algorithm for some computationally simple problems for proving its efficiency and then we used it for fuzzy rule base forming. Coevolutionary algorithm for unconstrained optimization is applied at the first stage (Michigan approach) and coevolutionary algorithm for constrained optimization is used at the second stage (Pittsburgh approach).

This method for fuzzy classifier rule base design was applied for some machine learning problems from UCI repository [7]. Some statistical investigations were performed. So we got a set of fuzzy classifiers for each classification problem.

The next idea is the multistep procedure. After multiple fuzzy classifiers forming we got a set of fuzzy classifiers for each classification problem. The natural step is collective forming fuzzy rule base using a set of classifiers that were generated with our approach. For collective forming of fuzzy classifier cooperate-competitive coevolutionary algorithm can be applied again. Thus we can repeat this procedure more times. So we formulated multistep procedure of fuzzy classifier forming. We have implemented this method and got good results for all classification problems mentioned above.

The details of the cooperative-competitive coevolutionary genetic algorithm for strategy adaptation are described in Section 2. The development of the new Michigan and Pittsburgh method combination for fuzzy classifier rule base design and details of multistep procedure are discussed in Section 3. The machine learning problems from UCI repository and the results of numerical experiments are presented in Section 4. Conclusions are listed in Section 5.

2 Cooperative-Competitive Coevolutionary Algorithm

One of the most complicated problems for genetic algorithm application is the algorithm parameters setting. Conventional genetic algorithm has at least three methods of selection (proportional, tournament, and rank) and three methods of recombination (one-point, two-point, and uniform). Mutation probability requires tuning as well. It is necessary to choose constraint handling method for constrained

optimization problems. A number of various combinations can be estimated at tens. Exhaustive search of combinations requires a lot of time and computational efforts. Parameters combination selection by chance is also a bad idea as algorithm efficiency for the same problem can differ very much for different parameters setting.

In [8] strategy adaptation by competing subpopulations has been suggested. Each subpopulation has its own search strategy (algorithm parameters combination). Resource redistribution provides domination of the subpopulation with the best search strategy for the problem-in-hand. This method can be considered as an example of coevolutionary genetic algorithm.

We have developed slightly different approach [6] that uses both competition and cooperation of individual genetic algorithms each having its own parameters setting. There are resource redistribution and migration operators simultaneously. Cooperation of individual conventional genetic algorithms is provided by migration of the best solutions to all the individual genetic algorithms. So, coevolutionary algorithm efficiency can increase due to the positive effect of subpopulations interacting. This cooperative-competitive coevolutionary genetic algorithm needs no tuning of special parameters. The details of the algorithm one can read in [6].

Cooperative-competitive coevolutionary genetic algorithm for unconstrained optimization has been tested on typical set of GA-community unconstrained optimization test problems. The reliability of optimum point catching has been used as the efficiency criterion. The common result of those investigations was the observation that coevolutionary algorithm competing with some dozens individual algorithms was mostly the second or third best ones among them and it is always more effective than individual genetic algorithm with average effectiveness. It seems to be close to success as coevolutionary algorithm has demonstrated its ability to provide effective problem-in-hand solving procedure without extra efforts for setting GA parameters.

The main idea of the coevolutionary algorithm adaptation for constrained optimization is using different methods of constraint handling ("death" penalty, dynamic or adaptive penalty function [9]) in search strategies of individual genetic algorithm. A migration method was modified for algorithm adaption for constrained optimization.

Cooperative-competitive coevolutionary genetic algorithm for constrained optimization has equal or better reliability than the best conventional genetic algorithm. Also coevolutionary algorithm has a better convergence rate than the best conventional genetic algorithm. These effects are provided by competitive cooperation between subpopulations in coevolutionary algorithm. Besides, coevolutionary genetic algorithm is a very appropriate tool for parallel computing with multiprocessors.

3 Multistep Fuzzy Classifier Forming

For the first step of fuzzy classifier forming combination of Michigan and Pittsburgh is used [5]. Michigan method is used for fuzzy rules with a high grade of certainty determination. Pittsburgh method is applied to determine of rules subset with good performance. Constraint for number of rules is used at the second stage of fuzzy classifier generating.

A prior to main stages of our method there are two important preparatory steps of fuzzy classifier generating: attribute fuzzification (fuzzy number semantic setting) and initial population of fuzzy rules forming for Michigan approach using a priori information from a learning sample.

Michigan-style stage. The chromosomes are fuzzy rules. Chromosome length is equal to the number of attributes; each gene is an index for the corresponding fuzzy number. Fitness function is certainty grade of the fuzzy rule calculated by a learning sample [1]. Genetic algorithm for unconstrained optimization is applied. New population forming method is modified. After generation performing, parents and children are combined to common array. Different fuzzy rules with the best values of fitness function for each class are selected to the next generation. This new population forming method provides diversity of rules for each class and diversity of classes in population. For each generation classification performance is calculated for population at whole. Population with the best value of classification performance is used for the next stage of fuzzy classifier generating. Cooperate-competitive coevolutionary genetic algorithm for unconstrained optimization is applied.

Pittsburgh-style stage. Chromosomes are the fuzzy rule sets. Chromosome length is equal to the population size for Michigan-style stage. Chromosome genes are binary. Value "1" means using the corresponding rule in the set, value "0" means not using the corresponding rule. Fitness function is classification performance. Constraint for number of rules is used. This value is specified by researcher. The constraint is used because it is better to have small number of rules in the final rule base. Cooperate-competitive coevolutionary genetic algorithm for constrained optimization is applied. New generation forming method is standard.

After multiple fuzzy classifiers forming we got a set of fuzzy classifiers for each classification problem. The natural step is collective forming fuzzy rule base using a set of classifiers that were generated with our approach. In this case Pittsburgh-style stage of fuzzy classifier forming is performed again. A set of fuzzy rule base is analog of fuzzy rule base generated after Michigan-style stage. We also use constraint for feasible number of rules. For collective forming of fuzzy classifier cooperate-competitive coevolutionary algorithm can be applied again. Thus we can repeat this procedure more times. So we formulated multistep procedure of fuzzy classifier forming. We have implemented this method and got good results for some classification problems from UCI repository.

4 Test Problems and Numerical Experiments

The developed method of fuzzy classifier rule base design has been applied for a number of classification machine learning problems from UCI repository [7]:

- Credit (Australia-1) (14 attributes, 2 classes);
- Credit (Germany) (24 attributes, 2 classes);
- Liver Disorder (6 attributes, 2 classes);
- Iris (4 attributes, 3 classes);
- Yeast (8 attributes, 10 classes);
- Glass Identification (9 attributes, 7 classes);
- Landsat Images (4 attributes, 6 classes).

Some statistical investigations were performed for all problems. For each problem classification performance values (correctly classified part of test sample) for each stage and other parameters are presented in Tables 1-7. We can see classification efficiency values and standard deviation values after multiple performing of the algorithm for one-step and for two-step fuzzy classifier forming. There is feasible number of rules in brackets. For the first three problems comparison with alternative classification methods has been performed (Table 8). These algorithms are Bayesian approach, multilayer perceptron, boosting, bagging, random subspace method (RSM), and cooperative coevolution ensemble learning (CCEL).

Table 1. Results of Fuzzy Classifier Forming for Credit (Australia-1)

Parameter	One-step fuzzy classifier forming	Two-step fuzzy classifier forming
Maximum performance	0,870 (10)	**0,891** (10)
	0,890 (20)	**0,919** (20)
	0,891 (30)	**0,926** (30)
Average performance	0,827 (10)	**0,888** (10)
	0,861 (20)	**0,918** (20)
	0,873 (30)	**0,924** (30)
Minimum performance	0,758 (10)	**0,886** (10)
	0,841 (20)	**0,910** (20)
	0,854 (30)	**0,922** (30)
Standard deviation	0,02482 (10)	**0,00174** (10)
	0,01231 (20)	**0,00269** (20)
	0,01035 (30)	**0,00171** (30)

Table 2. Results of Fuzzy Classifier Forming for Credit (Germany)

Parameter	One-step fuzzy classifier forming	Two-step fuzzy classifier forming
Maximum performance	0,767 (50)	**0,795** (50)
	0,794 (80)	**0,821** (80)
Average performance	0,762 (50)	**0,791** (50)
	0,790 (80)	**0,815** (80)
Minimum performance	0,755 (50)	**0,783** (50)
	0,784 (80)	**0,809** (80)
Standard deviation	**0,00357** (50)	**0,00431** (50)
	0,00296 (80)	**0,00534** (80)

Table 3. Results of Fuzzy Classifier Forming for Liver Disorder

Parameter	One-step fuzzy classifier forming	Two-step fuzzy classifier forming
Maximum performance	0,687 (10)	**0,713** (10)
	0,710 (15)	**0,739** (15)
	0,725 (20)	**0,757** (20)
Average performance	0,666 (10)	**0,705** (10)
	0,682 (15)	**0,731** (15)
	0,692 (20)	**0,748** (20)
Minimum performance	0,632 (10)	**0,699** (10)
	0,655 (15)	**0,719** (15)
	0,655 (20)	**0,739** (20)
Standard deviation	0,01500 (10)	**0,00449** (10)
	0,01669 (15)	**0,00608** (15)
	0,01731 (20)	**0,00554** (20)

Table 4. Results of Fuzzy Classifier Forming for Yeast

Parameter	One-step fuzzy classifier forming	Two-step fuzzy classifier forming
Maximum performance	0,598 (20)	**0,609** (20)
	0,606 (30)	**0,641** (30)
	0,626 (60)	**0,674** (60)
Average performance	0,573 (20)	**0,605** (20)
	0,586 (30)	**0,633** (30)
	0,593 (60)	**0,668** (60)
Minimum performance	0,540 (20)	**0,602** (20)
	0,555 (30)	**0,625** (30)
	0,542 (60)	**0,662** (60)
Standard deviation	0,01801 (20)	**0,00241** (20)
	0,01710 (30)	**0,00431** (30)
	0,02207 (60)	**0,00429** (60)

Table 5. Results of Collective Fuzzy Classifier Forming for Iris

Parameter	One-step fuzzy classifier forming	Two-step fuzzy classifier forming
Maximum performance	0,947 (3)	**0,980** (3)
	0,973 (4)	**0,980** (4)
	0,987 (5)	**0,987** (5)
	0,987 (6)	**0,993** (6)
Average performance	0,908 (3)	**0,980** (3)
	0,951 (4)	**0,980** (4)
	0,971 (5)	**0,987** (5)
	0,975 (6)	**0,993** (6)
Minimum performance	0,767 (3)	**0,980** (3)
	0,900 (4)	**0,980** (4)
	0,940 (5)	**0,987** (5)
	0,933 (6)	**0,987** (6)
Standard deviation	0,05643 (3)	**0,00000** (3)
	0,02623 (4)	**0,00000** (4)
	0,01303 (5)	**0,00000** (5)
	0,01073 (6)	**0,00211** (6)

Table 6. Results of Collective Fuzzy Classifier Forming for Glass Identification

Parameter	One-step fuzzy classifier forming	Two-step fuzzy classifier forming
Maximum performance	0,757 (20)	**0,836** (20)
	0,827 (30)	**0,874** (30)
Average performance	0,737 (20)	**0,824** (20)
	0,781 (30)	**0,861** (30)
Minimum performance	0,706 (20)	**0,813** (20)
	0,757 (30)	**0,827** (30)
Standard deviation	0,01388 (20)	**0,00737** (20)
	0,01831 (30)	**0,01354** (30)

Table 7. Results of Collective Fuzzy Classifier Forming for Landsat Images

Parameter	One-step fuzzy classifier forming	Two-step fuzzy classifier forming
Maximum performance	0,849 (10)	**0,851** (10)
	0,857 (15)	**0,861** (15)
	0,857 (20)	**0,864** (20)
Average performance	0,838 (10)	**0,850** (10)
	0,847 (15)	**0,859** (15)
	0,849 (20)	**0,863** (20)
Minimum performance	0,821 (10)	**0,848** (10)
	0,836 (15)	**0,856** (15)
	0,835 (20)	**0,862** (20)
Standard deviation	0,00783 (10)	**0,00107** (10)
	0,00416 (15)	**0,00144** (15)
	0,00546 (20)	**0,00090** (20)

Table 8. The Classification Performance Comparing for Different Algorithms

Algorithm	Credit (Australia-1)	Credit (Germany)	Liver Disorder
Collective method of fuzzy classifier forming	**0,928**	**0,821**	**0,757**
Basic method of fuzzy classifier forming	0,891	0,794	0,725
Bayesian approach	0,847	0,679	0,629
Multilayer perception	0,833	0,716	0,693
Boosting	0,760	0,700	0,656
Bagging	0,847	0,684	0,630
Random Subspace meethod	0,852	0,677	0,632
Cooperative Coevolution Ensemble Learning	0,866	0,746	0,644

We can see that classification efficiency is increased and standard deviation of values is decreased for two-step fuzzy classifier forming.

5 Conclusions

The main result of our work is collective fuzzy classifier forming method. Having generated some fuzzy classifiers we are able to construct more effective classifier from previous classifiers using again cooperative-competitive coevolutionary algorithm. A number of using fuzzy rules isn't increasing with this method. The approach of multistep fuzzy classifier forming has the following features:

1) The method improves classification performance without increasing number of rules.
2) The method reduces diversity of performance values for multiple algorithm runs, i.e. the method has higher statistical stability.
3) The method is more effective for more complicated classification problems (more attributes and classes).

For following investigations we should test our approach for more steps. Essential question is convergence of the method. Besides it may be not only convergence of classification efficiency value but also convergence of linguistic fuzzy rules.

Fuzzy classifier forming methods comparison with alternative classification methods by performance value demonstrates that both fuzzy classifier forming methods have better efficiency that present-day classification algorithms.

References

1. Ishibuchi, H., Nakashima, T., Murata, T.: Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems. IEEE Trans. on Systems, Man, and Cybernetics 29, 601–618 (1999)
2. Cordón, O., Herrera, F., Hoffmann, F., Magdalena, L.: Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases. World Scientific, Singapore (2001)
3. Herrera, F.: Genetic fuzzy systems: taxonomy, current research trends and prospects. Evol. Intel. 1(1), 27–46 (2008)
4. Ishibuchi, H., Nakashima, T., Kuroda, T.: A hybrid fuzzy GBML algorithm for designing compact fuzzy rule-based classification systems. In: Proc. of 9th IEEE International Conference on Fuzzy Systems, pp. 706–711 (2000)
5. Sergienko, R.B., Semenkin, E.S., Bukhtoyarov, V.V.: Michigan and Pittsburgh Methods Combining for Fuzzy Classifier Generating with Coevolutionary Algorithm for Strategy Adaptation. In: Proc. of 2011 IEEE Congress on Evolutionary Computation, New Orleans, LA, USA (2011)
6. Sergienko, R.B., Semenkin, E.S.: Competitive cooperation for strategy adaptation in coevolutionary genetic algorithm for constrained optimization. In: Proc. of 2010 IEEE Congress on Evolutionary Computation, pp. 1626–1631 (2010)
7. UCI Machine Learning Repository, `http://kdd.ics.uci.edu/`
8. Schlierkamp-Voosen, D., Mühlenbein, H.: Strategy Adaptation by Competing Subpopulations. In: Davidor, Y., Männer, R., Schwefel, H.-P. (eds.) PPSN III 1994. LNCS, vol. 866, pp. 199–208. Springer, Heidelberg (1994)
9. Michalewicz, Z., Schoenauer, M.: Evolutionary algorithms for constrained optimization parameter optimization problems. Trans. Evolutionary Computation 4, 1–32 (1996)

Particle Swarm Optimize Fuzzy Logic Memberships of AC-Drive

Nasseer K. Bachache and Jinyu Wen

College of Electrical and Electronic Engineering, Huazhong University of Science and Technology (HUST), Wuhan 430074, China
tech_n2008@yahoo.com

Abstract. A numerous industries need to the applications of a variable speed underneath a high quality controller, in recent years the ac drive has been one of the most important strategies in speed control due to a high reliability of induction motor and the development in power electronic field, this paper proposed the Fuzzy Logic Controller (FLC) to get a superior behavior over a wide range of speed variation. Fuzzy logic is a robust controller for linear and non-linear system, but adjusting fuzzy controller parameters is a challenging problem, it depends on operator's experience. (Nowadays, many intelligent techniques are used for this task). In this work Particle Swarm Optimization (PSO) algorithm is utilized to adapting centers and width of triangle membership functions, this method deal with a simulation of a complete mathematical model of an induction motor and its inverter. The simulation results demonstrate that the proposed PSO-FLC speed controller realizes a good dynamic behavior of the I.M compared with conventional FLC and PID controller.

Keywords: Particle Swarm Optimization PSO, Fuzzy Logic Control FLC, Voltage Source Inverter VSI, Induction Motor IM.

1 Introduction

Ac drive is a widespread application of poly phase inverters to adjusting speed motor. The typical inverter drives are voltage source inverter producing Pulse Width Modulated (PWM) signals with a sinusoidal fundamental to get a lower scale of harmonics. The controller of this system must be designed for the specified performance. The conventional control systems are normally based on the mathematical model of a plant as long as it is available with ac motor parameters (stator flux orientation) [1]. However, the nonlinear dependencies with stator flux must be developed. A highly nonlinear system is difficult to obtain an exact mathematical model. Such procedures are tedious and time-consuming, in addition to its complexity. Fuzzy logic control (FLC) or Fuzzy Inference System (FIS) are a powerful controller tool, even if system is non-linear and ill-defined and accurate mathematical model is unavailable. But, fuzzy controller suffers from the drawback of tuning of parameters (number of membership functions and its type, rules number,

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 460–469, 2012.

and formulating rules). The tuning of scaling factors for this parameter done either interactively by trial and error or human expert [2]. Therefore, the tunings of the FLC parameters are necessitated to an effective method for tuning. Nowadays, several new intelligent optimization techniques have been emerged, such as Genetic Algorithms (GA), exploiting the ideas of Darwinian evolution, Simulated Annealing (SA), Ant Colony Optimization (ACO) and Bacteria Foraging Optimization (BFO) among these nature-inspired strategies the Particle Swarm Optimization (PSO) algorithm is relatively novel [3], PSO has received great attention in a control system as such as the search of optimal PID controller. In this paper, generating fuzzy controller parameters are based PSO proposes as a modern intelligent algorithm.

2 Modeling and Simulation of Three Phases I.M.

One of the most popular induction motor models derived from its equivalent circuit is Krause's model is based on a transformation of the stator's currents and of the magnetic fluxes of the rotor to the reference frame "$d-q$" which rotates together with the rotor [4]. While, axis transformation is applied to transfer the three-phase parameters (voltage, current and flux) according to (d-q axes stationary frame), then coupling coefficients between the stator and rotor phases change continuously with the change of rotor position can be solved. Which stator and rotor parameters rotate at synchronous speed, and all simulated variables in the stationary frame can consider d.c quantities [5].The per-phase equivalent circuit diagrams of an I.M. in two-axis synchronously rotating reference frame are illustrated in figure (1).

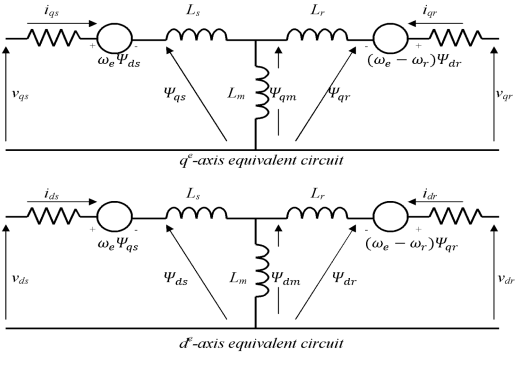


Fig. 1. I.M equivalent circuit is d-q axes components

Where: "V_{ds} V_{qs}", "V_{dr} V_{qr}", "i_{ds} i_{qs}" and "i_{dr} i_{qr}" d-q axes components stator voltage, rotor voltage, stator current and rotor current respectively. $(\Psi_{ds}, \Psi_{qs}, \Psi_{dr}, \Psi_{qr})$ d-q axes components flux linkage of the stator and rotor. "R_s R_r" Stator and rotor winding resistance. (L_{sl}, L_{rl}) Stator and rotor inductance. L_m Magnetizing inductance. w_r Rotor speed. "w_e" Synchronous speed. From the circuit diagram the following equations can obtained:

$$\frac{d}{dt}\begin{bmatrix}\Psi_{qs}\\ \Psi_{ds}\\ \Psi_{qr}\\ \Psi_{dr}\end{bmatrix}=\begin{bmatrix}0 & -w_e & 0 & 0\\ -w_e & 0 & 0 & 0\\ 0 & 0 & 0 & w_r-w_e\\ 0 & 0 & w_r-w_e & 0\end{bmatrix}\begin{bmatrix}\Psi_{qs}\\ \Psi_{ds}\\ \Psi_{qr}\\ \Psi_{dr}\end{bmatrix}+\begin{bmatrix}V_{qs}-R_s i_{qs}\\ V_{ds}-R_s i_{ds}\\ V_{qr}-R_r i_{qr}\\ V_{dr}-R_r i_{dr}\end{bmatrix} \tag{1}$$

$$\begin{bmatrix}\Psi_{qs}\\ \Psi_{ds}\\ \Psi_{qr}\\ \Psi_{dr}\end{bmatrix}=-\begin{bmatrix}L_{sl}+L_m & 0 & L_m & 0\\ 0 & L_{sl}+L_m & 0 & L_m\\ L_m & 0 & L_{sl}+L_m & 0\\ 0 & L_m & 0 & L_{sl}+L_m\end{bmatrix}\begin{bmatrix}i_{qs}\\ i_{ds}\\ i_{qr}\\ i_{dr}\end{bmatrix} \tag{2}$$

The development torque$T_e$can be obtained by interaction of air gap flux and rotor current I_r and solve the variables into *dq*-axes stationary frame to get the equation (3):

$$T_e=\left(\frac{3}{2}\right)\left(\frac{p}{2}\right)\left(\Psi_{dr}i_{qr}-\Psi_{qr}i_{dr}\right) \tag{3}$$

The dynamic torque equation of the rotor is formed in (4). Where, J is the rotor's inertia, and T_L is the external load torque [6].

$$T_e=T_L+\left(\frac{2}{p}\right)J\frac{dw_r}{dt} \tag{4}$$

The previous differential equations and dynamic torque were simulated using MATLAB / SIMULINK. The dynamic and static performance of the drive system under different load was tested.

3 SPWM Inverter Simulation

The Sinusoidal Pulse Width Modulation (SPWM) is the most popular usage in A.C drives. So; its performance should be a Voltage Source Inverter (VSI) and have a stiff source at the input [7]. A practical (VSI) consists of power bridge devices with three outputs; each one consists of two power switches and two freewheeling diodes. The inverter is supplied from D.C. voltage source via LC filter. In SPWM, the three output legs considered as three independent push-pull amplifiers as shown in fig. (2).

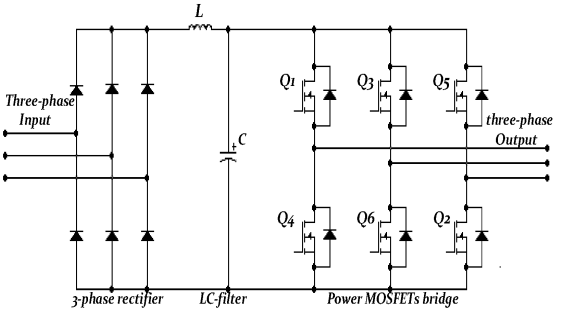


Fig. 2. A three phase (VSI) with three phase rectifier

SPWM inverter can be simulated by MATLAB/SIMULINK, firstly generate a carrier triangle signal and the three modulating signals to obtain the angular speed ($\omega_e t$), secondly compared the two signal sets to generate the switching signals of three push-pull devices. The output of the switches gives (V_{ao}, V_{bo}, V_{co}) then the three phases to load neutral(V_{an}, V_{bn}, V_{cn}) can be achieved by implementing equation (5).

$$\begin{bmatrix} \mathrm{V_{an}} \\ \mathrm{V_{bn}} \\ \mathrm{V_{cn}} \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}\begin{bmatrix} \mathrm{V_{ao}} \\ \mathrm{V_{bo}} \\ \mathrm{V_{co}} \end{bmatrix} \tag{5}$$

4 Particle Swarm Optimization

Particle swarm optimization (PSO) is a computation technique first proposed in 1995 by Kennedy and Eberhart [8,9]. This method has been found to be a robust method in solving non-linearity or non-differentiability problems, PSO algorithm didn't use evolutionary operators (mutation or crossover to manipulate algorithms). However, it simulates a dynamically population behavior (fish swarm or bird flocks), where social sharing of information takes place and individuals can profit from the discoveries and previous experience of all the other companions during the search for food. Thus, each companion is called particle and the population is called swarm it is assumed to fly in many directions over the search space in order to meet the demand fitness function [10, 11, 12]. For n-variables optimization problem, a flock of particles are put into the n-dimensional search space with randomly chosen velocities and positions knowing their best values, so far *(Pbest)* and the best position in the *n*-dimensional space. The velocity of each particle, adjusted accordingly to its own flying experience and the other particles flying experience. For the *ith* particle and *n*-dimensional space can be represented as equation (6), the best previous position of its particle is recorded as equation (7):

$$\mathrm{x_i} = \left(\mathrm{x_{i,1}}, \mathrm{x_{i,2}}, \ldots\ldots\ldots \mathrm{x_{i,n}}\right) \tag{6}$$

$$\mathrm{P_{best_i}} = \left(\mathrm{P_{best_{i,1}}}, \mathrm{P_{best_{i,2}}}, \ldots\ldots \mathrm{P_{best_{i,n}}}\right) \tag{7}$$

The velocity is an essential part of how PSO work [13] so as modified velocity and position of each particle can be calculated using the current velocity and distance from *($P_{best_{i,d}}$)* to *(g_{best_d})* as shown in equations. (8, 9):

$$\mathrm{V_{i,m}^{(It.+1)}} = \mathrm{W} * \mathrm{V_{i,m}^{(It.)}} + \mathrm{c1.r.}\left(\mathrm{P_{best_{i,m}}} - \mathrm{x_{i,m}^{(It.)}}\right) + \mathrm{c2.r.}\left(\mathrm{g_{best_m}} - \mathrm{x_{i,m}^{(It.)}}\right) \tag{8}$$

$$\mathrm{x_{i,m}^{(It.+1)}} = \mathrm{x_{i,m}^{(It.)}} + \mathrm{v_{i,m}^{(It.)}} \tag{9}$$

Where

i=1, 2,…, Number of particles.
m=1, 2,…, Dimension.
It.: Iterations pointer.

$V_{i,m}^{(It.)}$: Velocity of particle no. *i* at iteration *It.*
W: Inertia weight factor.
$c1$, $c2$: Acceleration constant.
r: Random number between(0-1).
$x_{i,m}^{(It.)}$: Current position of particle *i* at iteration *It.*
P_{best_i} : Best previous position of *ith* particle.
$\mathrm{g_{best_m}}$: Global best particle among all the particles in the population.

5 PSO Implementation Adapts FLC

As mentioned before the selection of Membership Functions (MFs) for the input and output variables and the determination of fuzzy rules are not available so there designing are very difficult, especially for one don't have experience of the system behavior. The conventional method (trial-and-error method) can be used in such situations [14]. There is no formal framework for the choice of the parameters of FLC and hence the means of tuning them and learning models in general have become an important subject of fuzzy control. The function of the fuzzy controller is to observe the pattern of the speed loop error signal and correspondingly updates the control signal, there are two input signals to the fuzzy controller: error (*E*) and change of error (*CE*), *CE* means derivative of error (*dE/dt*). A simple fuzzy logic controller of two inputs and one output can be designed; a seven triangle member-ships for each inputs, and nine memberships for output and forty nine *"if"* statement rules are used. PSO was utilized off line to design positions of triangle shape for input/output memberships. The completely system simulation using MATLAB/SIMULINK program are presented including IM model, inverter and FLC block sets. But, the optimization algorithm is implemented by using MATLAB/m-file program and linked with the system simulation program MATLAB/SIMULINK the performance of the system must be examined in each iteration and particles position during the optimization algorithm. Therefore, to check the system performance in each iteration and optimize position of the two inputs (*E, CE*) memberships. Fig (3) shows the simulation of AC drive based to fuzzy logic control. The optimization criteria *(Integrated of Time Weight Square Error ITSE)* equation (10) is used to evaluate accuracy performance of the fuzzy controller.

$$\mathrm{FF} = \mathrm{ITSE} = \int_0^{\mathrm{t}} \mathrm{t} * \mathrm{e}^2(\mathrm{t})\mathrm{dt} \tag{10}$$

The objective function is to minimize the fitness function FF. the PSO algorithm process can be summarized in the flowchart shown in figure (4). A set of good control parameters can yield a good step response that will result in performance criteria minimization in the time domain; this performance criterion is called Fitness Function (FF) can be evaluated in the variable (e) shown in fig(3).

6 Simulation Result

A comparison performance between the proposed Fuzzy PSO method, the ordinary FLC and PID controller are illustrated for step response and an arbitrary speed desired in figure (5) and figure (6) respectively. Figure (7) shows the speed desired with deferent loud. Figure (8) shows fuzzy surface of the two inputs and one output designed by the proposed method. Figure (9) shows the fuzzy controller membership functions designed using PSO.

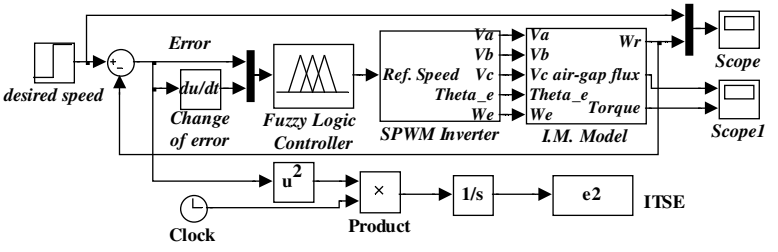


Fig. 3. Simulation of AC-derive with FLC

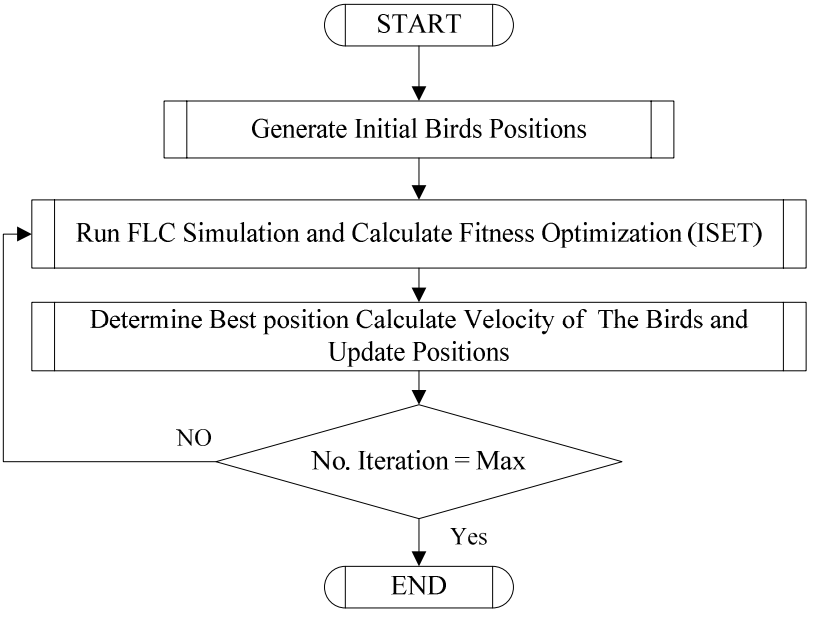


Fig. 4. Flowchart of PSO algorithm

7 Conclusions

A successfully generation of FLC parameters by PSO is demonstrated in this paper. It is adopted method to give a higher robust controller for this system (perfect speed-tracking and non-sluggish), figure (5) and figure (6) show the optimized FLC is more

closely with desired input speed and also figure(7) shows a strongly controller of speed with loud variation. For a complex systems the easy implementation SIMULINK-MATLAB can used in step "fitness function" of PSO algorithm. Increasing number of particles in PSO is more effective access than increasing number of iteration to optimize the system for a proposed method.

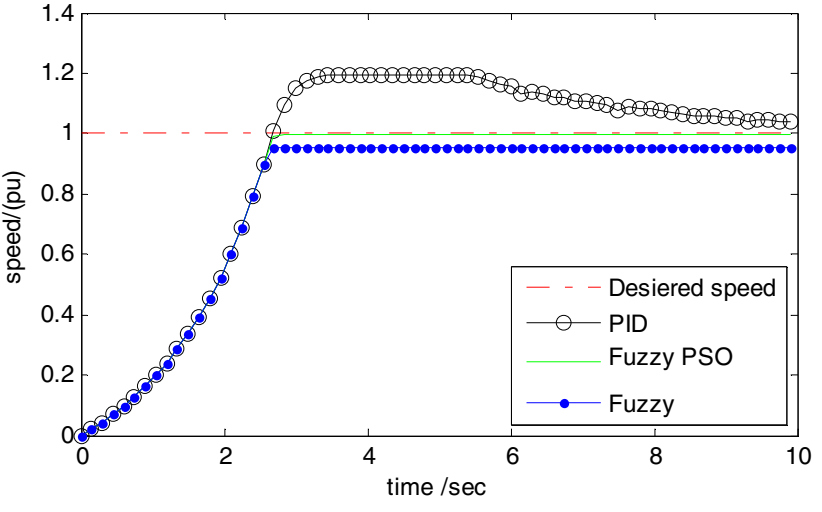


Fig. 5. IM speed step response of PID controller, ordinary FLC and optimized FLC (Fuzzy PSO)

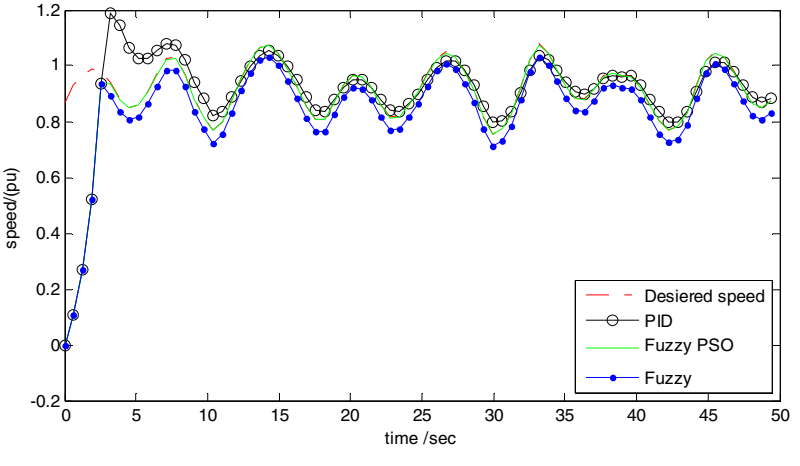


Fig. 6. An arbitrary speed between (1.1pu and 0.7pu) shows that the optimized FLC (Fuzzy_PSO) is closed with desired speed and its performance is the best compared with ordinary FLC and PID controller

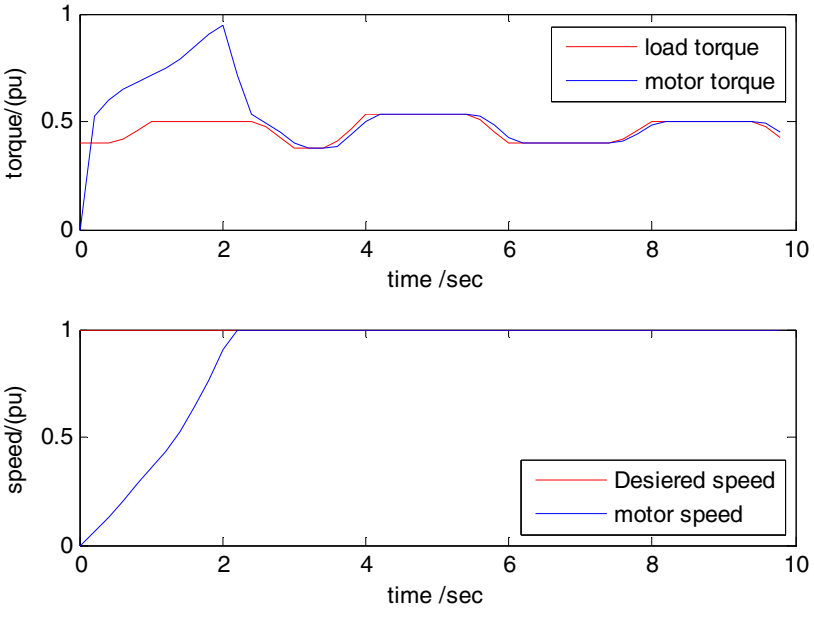


Fig. 7. Deferent loud derived by optimized FLC

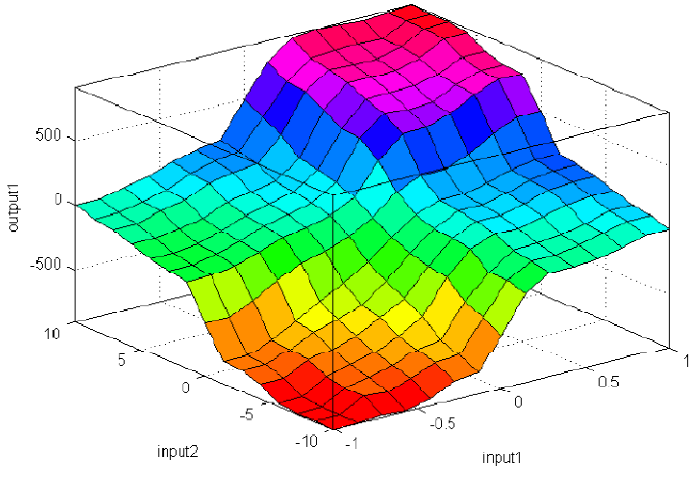


Fig. 8. Two inputs and output fuzzy surface designed using PSO method

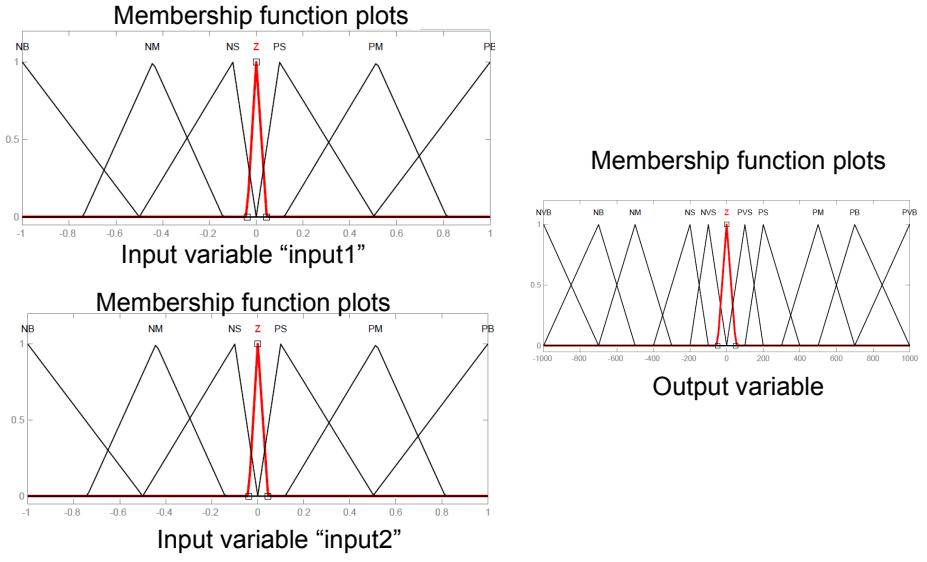


Fig. 9. Optimized F-variable triangle Memberships designed by PSO algorithm where “input1” is FLC (error (E)) and “input2” FLC (change of error (CE)) and “output1” signal FLC

References

1. Wlas, M., Abu-Rub, H., Holtz, J.: Speed Sensorless Nonlinear Control Of Induction Motor In The Field Weakening Region. In: IEEE 13th International Power Electronics and Motion Control Conference, pp. 1084–1089 (2008)
2. Dubey, M.: Design of Genetic Algorithm Based Fuzzy Logic Power System Stabilizers in Multimachine Power System. In: POWERCON 2008 & Power India Conference, October 12–15. IEEE, New Delhi (2008)
3. Rapaic, M.R., Kanovic, Z., Jelicic, Z.D.: A Theoretical and Empirical Analysis of Convergence Related Particle Swarm Optimization. Wseas Transactions on Systems and Control 11(4) (November 2009)
4. OzpineciLeon, B., Tolbert, M.: Simulink Implementation of Induction Machine Model A Modular Approach. IEEE International Electric Machines and Drives, 728–734 (2003)
5. Rigatos, G.G.: Modelling and Control for Intelligent Industrial Systems. Springer, Heidelberg (2011)
6. Shen, A., Huang, H., Kang, W.: Marine AC high-capacity drive experiment system. In: International Conference on Computer, Mechatronics, Control and Electronic Engineering, CMCE (2010)
7. Lara, O.A., Ekanayake, J., Cartwright, P., Hughes, M.: Wind Energy Generation Modelling and Control. A John Wiley and Sons, Ltd. (2009)
8. Chen, A.-L., Yang, G.-K., Wu, Z.-M.: Hybrid discrete particle swarm optimization algorithm for capacitated vehicle routing problem. Journal of Zhejiang University Science, 607–614 (2006)

9. Zhou, C., Gao, L., Gao, H.-B., Zan, K.: Particle Swarm Optimization for Simultaneous Optimization of Design and Machining Tolerances. In: Wang, T.-D., Li, X., Chen, S.-H., Wang, X., Abbass, H.A., Iba, H., Chen, G.-L., Yao, X. (eds.) SEAL 2006. LNCS, vol. 4247, pp. 873–880. Springer, Heidelberg (2006)
10. Xie, X.-F., Zhang, W.-J., Yang, Z.-L.: Adaptive Particle Swarm Optimization on Individual Level. In: International Conference on Signal Processing (ICSP), Beijing, China, pp. 1215–1218. IEEE (2002)
11. Esmin, A.A.A., Lambert-Torres, G., de Souza, A.C.Z.: A Hybrid Particle Swarm Optimization Applied to Loss Power Minimization. IEEE Trans. on Power Systems 20(2), 859–866 (2005)
12. Lin, W.-M., Hong, C.-M.: A New Elman Neural Network Based Control Algorithm for Adjustable Pitch Variable Speed Wind Energy Conversion Systems. IEEE Trans. on Power Electronics 26(2), 473–481 (2011)
13. Pedersen, M.E.H.: Tuning & Simplifying Heuristical Optimization Thesis for the degree of Doctor of Philosophy submitted to University of Southampton Computational Engineering and Design Group School of Engineering Sciences (January 2010)
14. Letting, L.K., Munda, J.L., Hamam, A.: Particle Swarm Optimized T-S Fuzzy Logic Controller for Maximum Power Point Tracking in a Photovoltaic System. In: 35th Photovoltaic Specialists Conference, pp. 89–94. IEEE (2010)

The Application of a Hybrid Algorithm to the Submersible Path-Planning*

Chongyang Lv[2], Fei Yu[1,2,**], Na Yang[2], Jin Feng[3], and Meikui Zou[2]

[1] College of Graduate, Harbin Engineering University, Harbin, China
[2] College of Science, Harbin Engineering University, Harbin, China
[3] College of Automation, Harbin Engineering University, Harbin, China
sglcy521@sohu.com

Abstract. The premature problem is always being a hot topic in the swarm intelligence research field. PSO could easily fall into local optima because the particles could quickly get closer to the best particle. To this end, this paper proposes a new hybrid PSO named HGC-PSO to solve this problem. The mutation mainly considers the $m+1$ particles which have the better fitness values. Firstly, we add the Gauss mutation to the current global optimal. Secondly, we use the Cauchy mutation to change the rest of the $m+1$ particles. The purpose of this method is to increase the population diversity and avoid the PSO fall into local optima. Finally, HGC-PSO is applied to path planning problem in 3D space for robot in this paper. The experiment of results prove that the proposed algorithm has higher convergence speed and precision, besides a path without collision is found.

Keywords: PSO, Path Planning, Mutation Operator, Cauchy Mutation.

1 Introduction

Particle swarm optimization (PSO) was come up in 1995 and soon became popular until now. The premature convergence is normal to the Global optimization algorithm, including PSO. While dealing with complex multi-modal searching problem, it is easy for the original PSO algorithm to fall into premature convergence.

Some research has been done to tackle this problem. One reason for PSO to converge to local optima is that particles in PSO can quickly converge to the best position once the best position has no change in a local optimum. Another is the diversity problem.

This paper presents a hybrid intelligent algorithm (HGC-PSO) to the premature convergence problem. Using the mutation operator of genetic algorithm, we can change the position of the particles. Thus, the proposed algorithm can get rid of the premature convergence and increase the diversity of the population.

* This paper is partially sponsored by National Natural Science Foundation of China Grant (51179039), and by grant from the PH. D. Programs Foundation of Ministry of Education of China (20102304110021).
** Corresponding author.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 470–478, 2012.

2 Particle Swarm Optimization Algorithm

PSO, which is originated from the birds and fish group behavior, is one represent of swarm intelligence algorithm.

PSO is applied in an extensive filed, such as the path planning, combinational optimization Problem. According to the three-dimensional space in the path planning problem, has adopted the method of artificial potential field method, A * search method , Visibility graphic method and so on. But these algorithms have some limitations. However, the path obtained from standard PSO is not the shortest. Thus, PSO often needs modified by the mutation operators.

3 Mutation Operator

3.1 Gauss Mutation

The one-dimensional Gauss density function centered at the origin is defined by:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{(x-\mu)^2}{2\sigma^2}}, -\infty < x < \infty.$$

The traditional Gauss mutation thought: On the individual $X_i = (X_{i1}, X_{i2}, \cdots, X_{in})$ in accordance with the following formula manipulation

$$X_{ij} = X_{ij} + \eta * G(0,1), \tag{1}$$

where $j = 1,2,\ldots n,$ and η is a constant which could control the variation of the step size, $G(0,1)$ is standard Gauss distribution.

3.2 Cauchy Mutation

The one-dimensional Cauchy density function centered at the origin is defined by:

$$f_i(x) = \frac{1}{\pi}\left(\frac{t}{(t^2 + x^2)}\right), \quad -\infty < x < \infty.$$

The traditional Cauchy mutation thought: If the individual $X_i = (X_{i1}, X_{i2}, \cdots, X_{in})$ satisfied $x_0 = 0$ and $\gamma = 1$ are called the standard Cauchy distribution $C(0,1)$. We can see it as follows:

$$X_{ij} = X_{ij} + \eta * C(0,1) \tag{2}$$

where $j = 1,2,\ldots n,$ and η is a constant which could control the variation of the step size.

4 HGC-PSO Algorithm and Numerical

According to, the convergence of PSO can be judged through the fitness variance. The definition of the fitness variance is given as follows.

$$\sigma^2 = \sum_{i=1}^{n} (\frac{f_i - f_{avg}}{f})^2 \tag{3}$$

where n is the particle population of the PSO, the fitness value of the i particle is f_i, The current average fitness was $f_{avg} = \frac{1}{n}\sum_{i=1}^{n} f_i$, f is a normalized scaling factor which limit to the size of σ^2 and can be expressed as

$$f = \max\{1, \max\{|f_i - f_{avg}|\}\} \; i \in [1, n].$$

No matter the PSO converges to the global optimum or local optimum, the particles will gather to one or more position of the global extremum. If the location is not a global optimum, the algorithm will fall into a premature convergence.

In this section, to solve the premature problem we propose a new hybrid algorithm by adding mutations.

4.1 HGC-PSO Algorithm

This paper considers the $m+1$ particles which have the better fitness values. Firstly, according to (1) we add the Gauss mutation on the current global optimal g_{best}. The rate of the mutation p_m is given.

$$p_m = \begin{cases} k, & \sigma^2 < \sigma_d^2 and f(gBest) > f_d \\ 0, & others \end{cases}, \tag{4}$$

where k is a random number independently generated from 0.1 to 0.3.

Secondly, we use the Cauchy mutation to change the rest of the $m+1$ particles. According to the fitness function, we can find the better adaptive values of the m particles. And corresponding to the particles, we generate the m random numbers $r_i, i = 1, 2, \cdots, m$. r_i distributed in [0,1]. If $r_i < p_m$, according to (2) we will produce the new location of the particles, and then enter into the next iteration.

4.2 Numerical Examples

In our Numerical examples, we choose the two test functions in table 1, where n is the dimension of the functions, $f_{\min}$ is the minimum values of the function, and $x \subseteq R_n$ is the search space.

The minimum values of the functions are desired. The population size is 40. The average values of the minimum values are obtained under ten times recycles. Then the comparisons between for the two functions are shown in Fig.1 and Fig.2, respectively.

Table 1. Test Functions

	n	$x \subseteq R_n$	$f_{\min}$
Unimodal Function: $f_1(x)=\sum_{i=1}^{n}(x_i^2-10\cos(2\pi x_i)+10)$	30	[-5.12,5.12]	$f_1(0, 0, \cdots, 0)$ =0
Multi-modal Function: $f_2(x)=\sum_{i=1}^{n}(100(x_{d+1}-x_d^2)^2+(x_d-1)^2)$	30	[-30,30]	$f_2(1, 1, \cdots, 1)$ =0

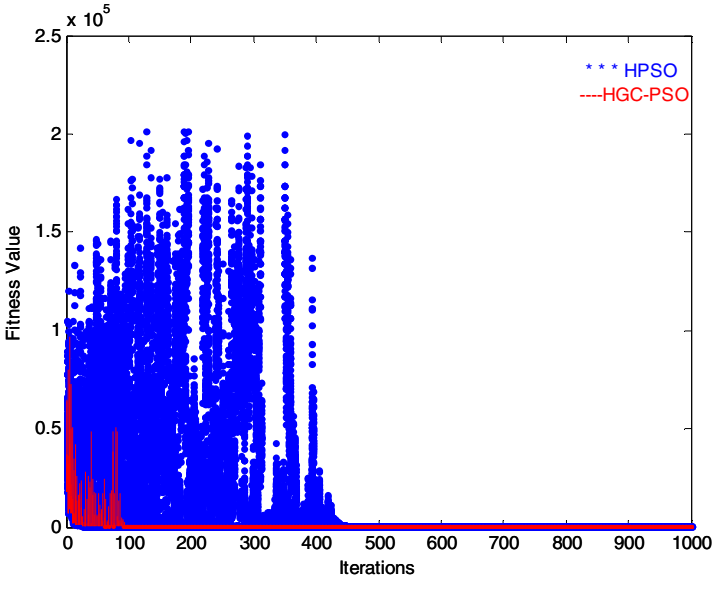


Fig. 1. The Unimodal Function of HPSO versus HGC-PSO

5 Simulation

5.1 Modeling

The purpose of this simulation is to find the path without obstacles. The starting point and finish point of the path are S and D, respectively. In order to make it close to the

underwater work environment, we indicate the obstacle by the spheres $O_1, O_2, \cdots, O_k$ $\{(o_i, r_i)|i = 1, 2, \cdots, k\}$, where o_i is the position of the centre of the sphere and r_i is the rad.

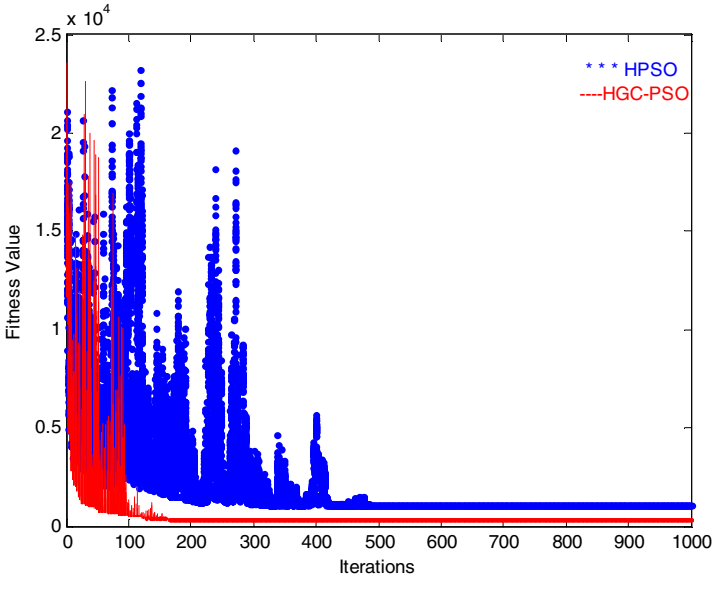


Fig. 2. The Multi-modal Function of HPSO versus HGC-PSO

The change of coordinates from Fig.3 to Fig.4 is as follows,

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \cos\alpha_x & \cos\alpha_y & \cos\alpha_z \\ \cos\beta_x & \cos\beta_y & \cos\beta_z \\ \cos\gamma_x & \cos\gamma_y & \cos\gamma_z \end{pmatrix} \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}.$$

Constructing a cube area $ABCDEFGH$, the path planning environment cube-space model is established. The surface $ABCE$ of the cube is in the surface $O'X'Y$. Take SD for $(n+1)$ equal portions, over each decile points, perpendicular to the z'-axis for the n plane $\prod_i (i = 1, 2, \cdots, n)$.

The path of the submersible from the S point to the point D make up with the various hierarchical point of connection components. Divided SD into n pieces equally, and draw planes through each point of division perpendicular to z-axis. By this way, the path from point S to point P of AUV is formed by the path points of each layer between them.

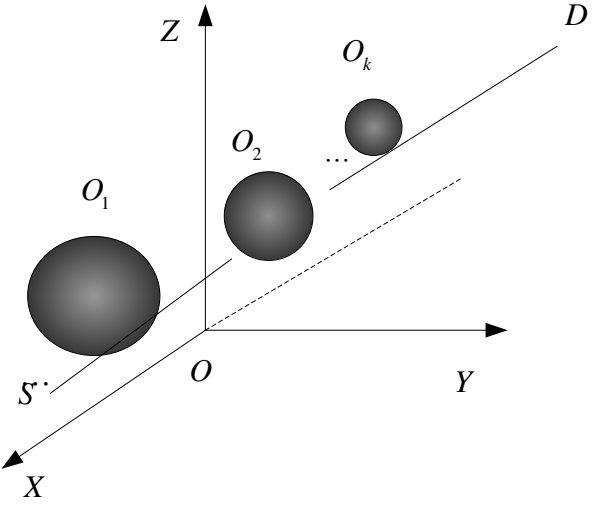


Fig. 3. The obstacles in $O-XYZ$

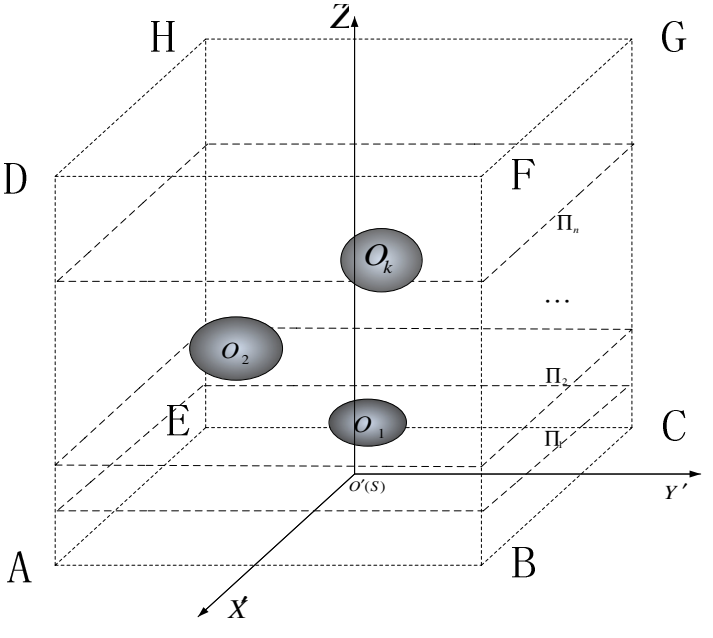


Fig. 4. The obstacles in $O'-X'Y'Z'$

The submersible start from the point S to reach the plane $\prod_1$, then find a point $P^1(k_1, j_1)$. Similarly, in order to find $P^{n-1}(k_{n-1}, j_{n-1})$ in the plane $\prod_{n-1}$, connecting these points will form a path from S to D. The path length is:

$$L_{SD} = \sum_{i=0}^{n+1} \sqrt{(x_{kj}^i - x_{kj}^{i+1})^2 + (y_{kj}^i - y_{kj}^{i+1})^2 + (z_{kj}^i - z_{kj}^{i+1})^2}.$$

5.2 Simulation Results

The start and target points coordinate respectively [0, 0, 0] and [200, 175, 140]. The submersible moves from point S to D. The three-dimensional space environment is [0, 200;0, 200;0, 150].

In this space there are six obstacles. Their coordinates as: [30, 60,25], [50, 100, 80], [75, 40, 45], [110, 80, 90], [140, 145, 120], [50, 50, 120] and radius respectively 20, 14, 20, 18, 20, 10. Using the HGC-PSO algorithm, we select 40 particles. The found path is shown in Fig.5-Fig.7 from different perspectives.

This path is a collision free from the starting point to the end and satisfies the requirements. The simulations verify the feasibility of the algorithm is available.

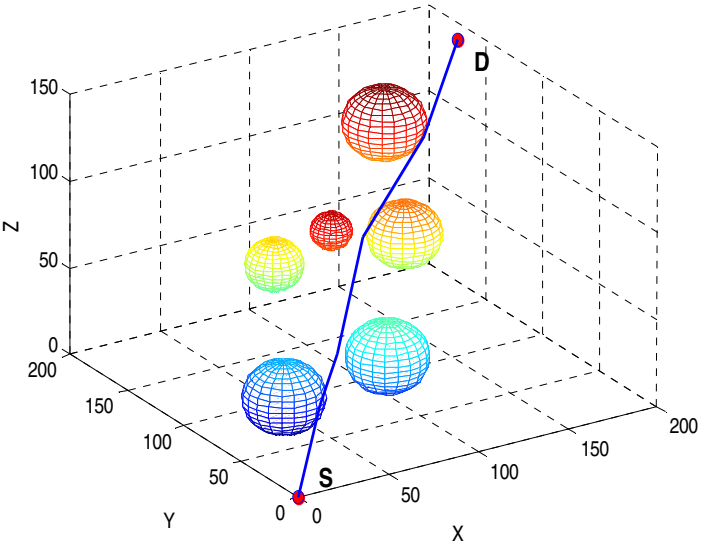


Fig. 5. Simulation results (visual angle A)

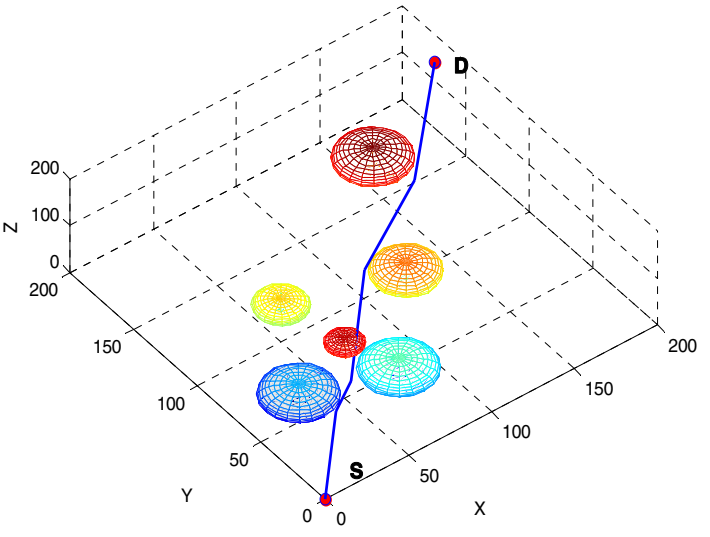


Fig. 6. Simulation results (visual angle B)

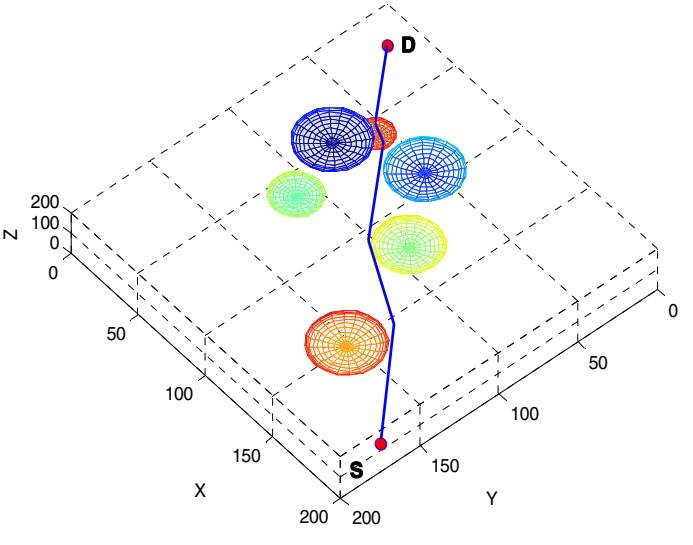


Fig. 7. Simulation results (visual angle C)

6 Conclusion

The idea of HGC-PSO is to use Gauss and Cauchy mutation operators to help PSO avoid local optima. After changing the direction of the particles by Gauss mutation, the diversity has been increased though Cauchy mutation. Compared to standard PSO, the proposed can find better solutions than PSO. Simulation experimental results have obtained a global collision-free path to validate the HGC-PSO. A simulation under known environment of the three-dimensional path planning is given.

References

1. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proc. IEEE Int'l. Conf. on Neural Networks, IV, pp. 1942–1948. IEEE Service Center, Piscataway (1995)
2. Lovbjerg, M., Krink, T.: Extending Particle Swarm Optimisers with Self-Organized Criticality. In: Proceedings of Fourth Congress on Evolutionary Computation, vol. 2, pp. 1588–1593 (2002)
3. Coelho, L.S., Krohling, R.A.: Predictive controller tuning using modified particle swarm optimization based on Cauchy and Gaussian distributions. In: Proceedings of the 8th On-Line World Conference on Soft Computing in Industrial Applications, WSC8 (2003)
4. Feller, W.: An Introduction to Probability Theory and Its Applications, 2nd edn., vol. 2. John Wiley & Sons, Inc. (1971); van den Bergh, F.: An Analysis of Particle Swarm Optimizers(1971)
5. Caroll, K.P., McClaran, S.R., Nelson, E.L., et al.: AUV Path Planning: An A* Approach to Path Planning width Consideration of Variable Vehicle Speeds and Multiple, Overlapping, Time-Dependent Exclusion Zones. In: Proc. Sym. on AUV Technology, pp. 79–84. IEEE, Washington, DC (1992)
6. Lee, S., Park, J.: Neural computation for collisionfree path planning. Journal of Intelligent Manufacturing (2), 315–326 (1991)
7. Holland, J.H.: Adaptation in Natural and Artificial System. The University of Michigan Press, Ann Arbor (1975)

Memetic Three-Dimensional Gabor Feature Extraction for Hyperspectral Imagery Classification

Zexuan Zhu[1], Linlin Shen[1], Yiwen Sun[2],
Shan He[3], and Zhen Ji[1,*]

[1] City Key Laboratory of Embedded System Design,
College of Computer Science and Software Engineering, Shenzhen University,
Shenzhen, China 518060
[2] Shenzhen Key Lab of Biomedical Engineering, School of Medicine,
Shenzhen University, China 518060
[3] School of Computer Science, University of Birmingham, Edgbaston, Birmingham,
B15 2TT, UK
jizhen@szu.edu.cn

Abstract. This paper proposes a three-dimensional Gabor feature extraction for pixel-based hyperspectral imagery classification using a memetic algorithm. The proposed algorithm named MGFE combines 3-D Gabor wavelet feature generation and feature selection together to capture the signal variances of hyperspectral imagery, thereby extracting the discriminative 3-D Gabor features for accurate classification. MGFE is characterized with a novel fitness evaluation function based on independent feature relevance and a pruning local search for eliminating redundant features. The experimental results on two real-world hyperspectral imagery datasets show that MGFE succeeds in obtaining significantly improved classification accuracy with parsimonious feature selection.

Keywords: Memetic Algorithm, Gabor Feature Extraction, Hyperspectral Imagery Classification.

1 Introduction

Hyperspectral imaging captures an image of objects with wavelengths ranging from the visible spectrum to the infrared region. The technology has allowed more elaborate spectral-spatial models for more accurate image classifications, object discrimination, and material identification. Hyperspectral imagery data usually contains tens and thousands of images simultaneously collected from various spaced spectral bands [1]. One key issue of hyperspectral imagery classification is how to deal with the high dimensionality and improve the class separability.

* Corresponding author.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 479–488, 2012.

Many dimensionality reduction techniques including feature selection and feature extraction have been used to address the problem. Feature selection [2,3], also known as band selection in the context of hyperspectral imagery classification, selects relevant bands and removes irrelevant/redundant ones in original feature space. Feature extraction techniques [4], transforms the given spectral bands to generate a new set of features possessing high information packing properties compared with the original bands. The most discriminative information is concentrated to relative small number of selected new features with which superior classification accuracy is permitted.

In this paper, we propose a novel memetic 3-D Gabor feature extraction called MGFE for pixel-based hyperspectral imagery classification. Particularly, MGFE conducts 3-D Gabor feature generation and selection simultaneously in a memetic algorithm (MA) framework [5]. The hyperspectral imagery cube first undergoes 3-D Gabor wavelet transformation and new 3-D Gabor features are generated. Afterward, the discriminative features are picked out as desirable signatures for final classification. The evolutionary search mechanism of MA is responsible for optimizing both the parameter settings of 3-D Gabor wavelet transformation and the selection of feature subset. MGFE is characterized with a novel feature subset evaluation measure based on independent relevance and a pruning local search, which is efficient for eliminating redundant features. The performance of MGFE is evaluated using the real-world hyperspectral imagery data of Kennedy Space Center (KSC) [6] and Indian Pine AVIRIS [7]. Comparison studies with other state-of-the-art dimensionality reduction methods show that MGFE succeeds in obtaining superior classification accuracy with compact feature subset.

The remainder of this paper is organized as follows. Section 2 describes the 3-D Gabor wavelet feature generation and the proposed memetic 3-D Gabor feature extraction algorithm. Section 3 presents the experimental results of MGFE and other compared algorithms on two real-world hyperspectral imagery datasets. Finally the conclusion is given in Section 4.

2 Methodology

2.1 Three-Dimensional Gabor Wavelet Feature Generation

Gabor wavelet is a power tool proposed to maximize joint time/frequency and space/frequency resolutions for signal analysis [8]. Gabor wavelet has been successfully used to extract features for texture classification [9], face recognition [10], medical image registration [11], etc. In this study, we apply 3D Gabor wavelet [11] in hyperspectral image cube to reveal the signal variances in space, spectrum, and joint spatial/spectral domains. Let V be a hyperspectral image cube of an $X \times Y$ region captured in B spectral bands. $V(x, y, b)$ is the signal information of an area at sampled spatial location (x, y) which is captured in

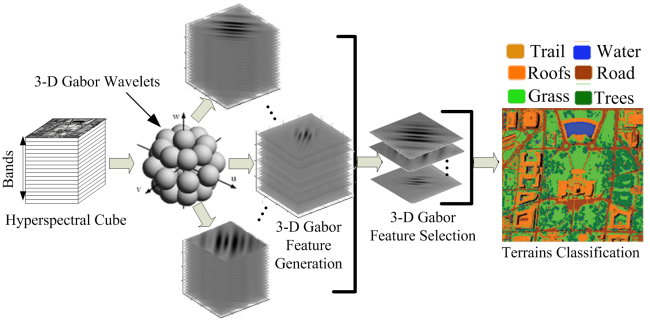


Fig. 1. Three-dimensional Gabor feature extraction for hyperspectral imagery classification

spectral band b. The circular 3-D Gabor wavelets in spatial-spectrum domain (x, y, b) is defined as follows:

$$\begin{aligned}\Psi_{f,\theta,\varphi}(x,y,b) = \frac{1}{S} \times \exp(j2\pi(xu + yv + bw)) \\ \times \exp(\frac{(x-x_c)^2 + (y-y_c)^2 + (b-b_c)^2}{-2\sigma^2})\end{aligned} \tag{1}$$

where $u = f \sin\varphi \cos\theta$, $v = f \sin\varphi \sin\theta$, and $w = f \cos\varphi$. Variable S is a normalization scale, f is the central frequency of the sinusoidal plane wave, φ and θ are the angles of the wave vector with w-axis and $u-v$ plane in frequency domain (u, v, w) (as shown in Figure 1), and σ is the width of Gaussian envelop in (x, y, b) domain. (x_c, y_c, b_c) is the position for signal analysis.

The response of signal to wavelet $\Psi_{f,\theta,\varphi}(x, y, b)$ represents the strength of variance with frequency amplitude f and orientation (φ, θ). The response of $V(x, y, b)$ to $\Psi_{f,\theta,\varphi}(x, y, b)$ is represented as:

$$\Theta_{f,\theta,\varphi}(x,y,b) = |(V \otimes \Psi_{f,\theta,\varphi})(x,y,b)| \tag{2}$$

where $\otimes$ is the convolution operation. $\Theta_{f,\theta,\varphi}(x, y, b)$ reveals the information of signal variances around location (x, y, b) with center frequency f and orientation (θ, φ) at joint spatial and spectral domain. In the following text, we refer the response of the whole image region to a Gabor wavelet, i.e., $\Theta_{f,\theta,\varphi}(x, y, b)$, as a 3-D Gabor feature which is characterized with f, θ, φ, and b.

With appropriate selections of f, θ, φ, and b, a 3-D Gabor feature is capable of capturing the desirable signatures of pixel-based terrains/objects classification from a specific aspect. The key issue is what kinds of 3-D Gabor features should be generated for achieving satisfactory classification accuracy. By searching the space of f, θ, φ, and b, one can find the solution with optimal classification accuracy. In the following subsection, we introduce the proposed memetic algorithm for searching the optimal 3-D Gabor feature subset. The procedure of 3-D Gabor feature generation and selection is illustrated in Fig. 1.

2.2 Memetic Algorithm for 3-D Gabor Feature Optimization

Memetic algorithm (MA) [5], a most well-known paradigm of memetic computing [12], is widely recognized as a form of population-based hybrid global evolutionary algorithm coupled with with individual learning or local search heuristic. Taking advantage of both global and local search, MAs succeeds in obtaining better performance than their conventional counterparts in various real-world applications. In this study, we propose a genetic algorithm (GA) [13] based MA namely MGFE for optimizing the 3-D Gabor feature extraction. The procedure of the MGFE algorithm is outlined in Algorithm 1.

Algorithm 1. MA Based 3-D Gabor Feature Extraction

1: **BEGIN**
2: Randomly initialize a population of chromosomes encoding candidate 3-D Gabor feature parameters;
3: **While** stopping criteria are not satisfied **do**
4: Generate 3-D Gabor features of the 4-tuples encoded in each chromosome based on (1) and (2);
5: Evaluate the fitness of each feature subset encoded in the population based on (5);
6: Perform **Pruning Local Search** on each chromosome to eliminate redundant features;
7: Update the population based on selection, crossover and mutation operations;
8: **End While**
9: **END**

Chromosome Encoding: At the beginning of MGFE, a population of chromosomes is generated randomly with each chromosome encoding a candidate 3-D Gabor feature subset. A chromosome (as shown in Fig. 2) is designed as a string of n 4-tuples (b,f,θ,φ) which each can be used to generate a corresponding 3-D Gabor feature based on (1) and (2). The number of features n is constrained in $[1, 1000]$.

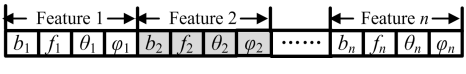


Fig. 2. A 3-D Gabor feature chromosome

In each 4-tuple, the band number b is an integer in $[1, B]$, f is in $[0, 0.5]$, θ and φ take real values in $[0, \pi]$. Without any constraints, the search space of 3-D Gabor features is infinite. In fact, features with similar wavelet parameters are redundant with each other for capturing similar signatures. Therefore, we could sample f, θ, and φ in certain intervals to reduce redundancy in the feature extraction, and meanwhile cutting down the size of the search space significantly. Particularly, the frequency f takes the values of $[0.5, 0.25, 0.125, 0.0625]$ with orientations θ and φ set as the values of $[0, \pi/4, \pi/2, 3\pi/4]$. Since the frequency

vector points to the same direction with different θ when $\varphi = 0$, there are in total 52 wavelets available for feature extraction. Let B denotes the total number of bands. Each pixel at (x, y) can be represented with $52B$ 3-D Gabor features and the complexity of searching the optimal feature subset is 2^{52B}. Since B usually ranges from hundreds to thousands, the feature space is challenging for most of search algorithms.

Fitness Evaluation: After the initialization, the chromosome population undergoes a loop of evolution until the predefined stopping criteria are satisfied. The stopping criteria could be a convergence to the global optimal or a maximum computational budget is reached. In each evolution generation, the goodness of the 3-D Gabor feature subset encoded in each chromosome should be evaluated based on a certain fitness function.

Since the ultimate objective is accurate classification of the pixels, the classification accuracy should be the first choice for chromosome fitness evaluation (FE). However, the evaluation of classification accuracy based on a classifier could be very time consuming especially when evolutionary algorithms like GA and MA need thousands of FEs to evolve a satisfying solution. Instead of using classification accuracy, MGFE evaluates the chromosome fitness based on non-redundant relevance, also named independent relevance, of the encoded features to the class labels. The independent relevance, whose evaluation is efficient, approximates the classification accuracy fairly well.

Before introducing the definition of independent relevance, we present some preliminary knowledge on feature relevance and redundancy. Let C_{xy} be the class labels of all pixels on the image, and $\Theta_i = \Theta_{f_i,\theta_i,\varphi_i}(x, y, b_i)$ be the ith feature encoded in a chromosome $\mathfrak{X}$. The relevance of Θ_i to C_{xy} is measured by the symmetrical uncertainty [14]:

$$Su(\Theta_i, C_{xy}) = 2\left[\frac{IG(\Theta_i|C_{xy})}{H(\Theta_i) + H(C_{xy})}\right] \tag{3}$$

where $IG(\Theta_i|C_{xy})$ denotes the information gain between Θ_i and C_{xy}, $H(\Theta_i)$ and $H(C_{xy})$ represent the entropies of Θ_i and C_{xy}, respectively.

The redundancy between two features is measured using approximated Markov blanket (AMB) [15]. Given two features Θ_i and Θ_j encoded in $\mathfrak{X}$, if and only if $Su(\Theta_i, C_{xy}) \geq Su(\Theta_j, C_{xy})$ and $Su(\Theta_i, \Theta_j) \geq Su(\Theta_j, C_{xy})$, Θ_i is said to be an AMB of Θ_j. Thereby Θ_j is redundant to Θ_i, and Θ_j gives no more discriminatory information of C_{xy} in the existence of its AMB Θ_i.

Based on the definition of feature relevance and redundancy, the independent relevance of an encoded 3-D Gabor feature $IR(\Theta_i)$ is defined as:

$$IR(\Theta_i) = \begin{cases} Su(\Theta_i, C_{xy}), & \text{if } \Theta_i \text{ has no AMB in } \mathfrak{X} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

The relevance of a feature is counted only if it is not redundant with any other features. The fitness of chromosome $\mathfrak{X}$ is evaluated based on independent relevance:

$$Fitness(\mathfrak{X}) = \sum_{i=1}^{|\mathfrak{X}|} IR(\Theta_i) \tag{5}$$

where $|\mathfrak{X}|$ denotes the number of features encoded in $\mathfrak{X}$.

Pruning Local Search. Since only the relevance of non-redundant features contribute to the fitness of a chromosome, the redundant 3-D Gabor features encoded in each chromosome can be removed for the sake of reducing computation complexity. After the fitness evaluation, each chromosome undergoes a local search to get rid of the redundant features. The procedure of local search is outlined in Algorithm 2, where features are checked pairwise and the one redundant to the other is removed. In practical, the identification of redundant features has been done during fitness evaluation, so the result can be saved and reused in local search.

Algorithm 2. Local Search

1: **INPUT:** a chromosome $\mathfrak{X}$
2: **BEGIN**
3: **For** $i = 1$ to $|\mathfrak{X}| - 1$ **do**
4: **For** $j = i + 1$ to $|\mathfrak{X}|$ **do**
5: **If** $Su(\Theta_i, C_{XY}) \geq Su(\Theta_j, C_{XY})$ and $Su(\Theta_i, \Theta_j) \geq Su(\Theta_j, C_{XY})$ **then**
6: Remove Θ_j;
7: **Else If** $Su(\Theta_j, C_{XY}) \geq Su(\Theta_i, C_{XY})$ and $Su(\Theta_i, \Theta_j) \geq Su(\Theta_i, C_{XY})$ **then**
8: Remove Θ_i and continue line 3.
9: **End If**
10: **End For**
11: **End For**
12: **END**

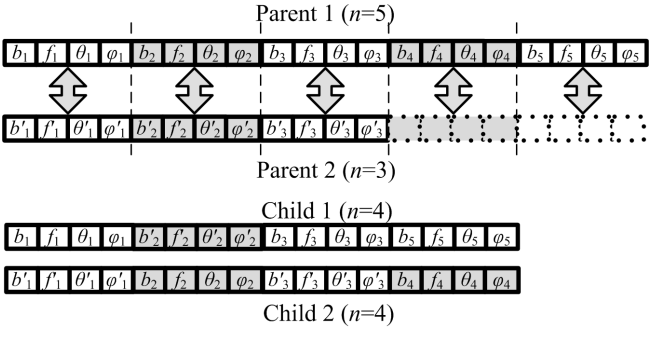


Fig. 3. An example of crossover

Evolutionary Operations. Following the fitness evaluation and local search, the population is evolved using evolution operators including linear ranking selection, uniform crossover, and mutation. Here, it is notable that the uniform crossover is performed on each 4-tuple, rather than each element, to ensure the consistency of the encoded 3-D Gabor features. Moreover, because the length of a chromosome is variable, a specific appending operation is used to make the normal crossover work on chromosomes of different lengths. For example, as shown in Figure 3, given two parent chromosomes of five and three 4-tuples respectively, two dummy 4-tuples are appended to the shorter one so that the two parent chromosomes have equal length. Afterward, the uniform crossover is applied on each 4-tuple and then the dummy 4-tuples are taken out, resulting in two children each of four 4-tuples. The mutation operation randomly changes an element in a chromosome to another value that falls in the corresponding range.

3 Experimental Results

The performance of MGFE is evaluated on two hyperspectral imagery datasets: Kennedy Space Center (KSC) [6] and Indiana pines AVIRIS (Indiana) [7]. The KSC data was acquired over the KSC, Florida, on March 23, 1996 using NASA's airborne visible infrared imaging spectrometer (AVIRIS). In the original 224 bands, 48 bands are identified as water absorption and low SNR bands, leaving 176 spectral bands for classification. Eight classes representing various land cover types were defined. The Indian data is a section of a scene taken over northwest Indiana's Indiana Pines by the AVIRIS sensor in 1992. It contains 10366 pixels, 220 bands, and 16 classes. The information of the datasets is summarized in Table 1.

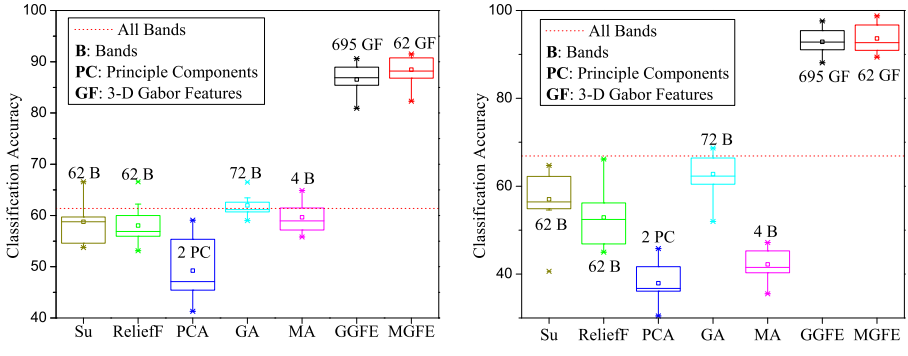
Table 1. The information of the two hyperspectral imagery datasets

Data	#Samples(pixels)	#Bands	#Classes
KSC	1379	176	8
Indiana	10366	220	16

For comparison study, we consider six computation efficient dimensionality reduction methods including ReliefF [16], Symmetrical Uncertainty (Su) based filter ranking method [17], Principle Component Analysis (PCA) [4], GA and MA based band selection, and also the counterpart GA based 3-D Gabor feature extraction (GGFE), i.e., MGFE without pruning local search. The MA and GA based band selection are directly applied to the original data by simplifying the 4-tuple in each chromosome to encode only a single band number b. GA, MA, GGFE, and MGFE use the same parameter settings with population size = 50, crossover probability = 0.6, and mutation rate = 0.1. All these four evolutionary algorithms are stopped when a maximum iteration number of 100 or a search convergence is reached. Unlike MA and GA, which automatically determine the number of selected features, ReliefF and Su need predefined number of selected

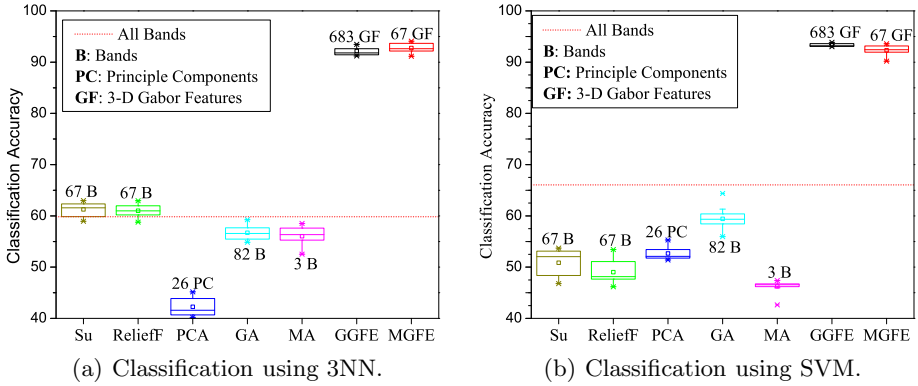
features. To make fair comparison, the selected feature size of ReliefF and Su are set according to that of MGFE. For PCA, the dimensionality reduction is accomplished by choosing enough eigenvectors to account for 95% percentage of the variance in the original data.

On each dataset, all algorithms are trained with 5% of randomly sampled pixels and tested on the remaining unseen 95% using 3-Nearest-Neighbor (3NN) and Support Vector Machine (SVM) [18] classifiers. SVM uses one-vs-one strategy to handle the multiclass problem. The experiment on each dataset is independently run for 30 times and the classification accuracy and the number of selected bands/3-D Gabor features are reported. The average classification accuracy using all bands is also provided as the baseline performance.



(a) Classification using 3NN. (b) Classification using SVM.

Fig. 4. Classification accuracy of all algorithms on KSC data



(a) Classification using 3NN. (b) Classification using SVM.

Fig. 5. Performance of all algorithms on Indiana data

The performance of all algorithms on KSC and Indiana datasets is depicted in Fig. 4 and Fig. 5, respectively. The classification accuracies of 30 runs are presented using box plot and the corresponding average numbers of selected

bands/pricinple components/3-D Gabor features are labeled around the box. It is shown that MGFE and GGFE obtain significant better classification accuracy with both 3NN and SVM than the other algorithms, which suggest the 3-D Gobor feature extraction does capture or even enhance the desirable signatures for pixel-based classification. The performance of MGFE and GGFE is competitive but MGFE manages to attain slightly better average accuracy with much more compact 3-D Gabor feature subset than GGFE. The pruning local search used in MGFE plays an key role for eliminating redundant features. The band selection algorithms, including Su, ReliefF, GA, and MA, fail to improve the classification accuracy with respect to the baseline performance of all bands. The classification accuracy of PCA is inferior to the other algorithms.

Comparing the classification accuracy of 3NN and SVM, we can see that some algorithms are sensitive to the classifier used. GGFE and MGFE show good robustness with both 3NN and SVM, which suggests that the independent relevance introduced for 3-D Gabor feature subset evaluation is of good generality when cooperating with different classifiers.

4 Conclusion

A 3-D Gabor feature extraction based on memetic algorithm (MGFE) was proposed for hyperspectral imagery classification in this study. Particularly, MGFE first applies 3-D Gabor wavelets to capture the signal variances, i.e., the 3-D Gabor features, in spatial-spectrum domain, and then a memetic algorithm is introduced to search the desirable 3-D Gabor features. The experimental results on the real-world hyperspectral imagery datasets demonstrate that MGFE is capable of revealing the information of signal variances efficiently, thereby resulting in significantly improved classification accuracy. With a novel feature subset evaluation measure based on independent relevance and a pruning local search, MGFE succeeded in identifying relevant features and eliminating redundant ones efficiently.

Acknowledgment. This work was supported partially by the National Natural Science Foundation of China, under Grants 61171125, 61001185, 61102119, and 60903112, the NSFC-RS joint project, the Fok Ying-Tung Education Foundation, Guangdong Natural Science Foundation, under Grants 10151806001000002 and S2011040004700, the Foundation for Distinguished Young Talents in Higher Education of Guangdong, under Grant LYM10113, Scientific Research Foundation for the Returned Overseas Chinese Scholars, Ministry of Education of China, the Open Research Fund of State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, under Grant 11R02, the Science Foundation of Shenzhen City, under grant JC201104210035A, and the Shenzhen City Foundation for Distinguished Young Scientists.

References

1. Chang, C.-I.: Hyperspectral Imaging: Techniques for Spectral Detection and Classification. Kluwer Academic/Plenum Publishers, New York (2003)
2. Dash, M., Liu, H.: Feature Selection for Classification. Intell. Data Anal. 1, 131–156 (1997)
3. Guyon, I., Elisseeff, A.: An Introduction to Variable and Feature Selection. J. Mach. Learn. Res. 3, 1157–1182 (2003)
4. Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L.: Feature Extraction, Foundations and Applications. Series Studies in Fuzziness and Soft Computing. Physica-Verlag, Springer (2006)
5. Moscato, P.: Memetic Algorithm: A Short Introduction. In: New Ideas in Optimization. McGraw-Hill, London (1999)
6. Morgan, J.T.: Adaptive Hierarchical Classifier with Limited Training Data. Ph.D thesis, University of Texas at Austin (2002)
7. AVIRIS NW Indiana's Indian Pines, Data Set (1992), https://engineering.purdue.edu/~biehl/MultiSpec/hyperspectral.html
8. Gabor, D.: Theory of Communication. Part 1: The Analysis of Information. J. Inst. Elect. Eng. III, Radio Commun. Eng. 93, 429–441 (1946)
9. Weldon, T.P., Higgins, W.E., Dunn, D.F.: Efficient Gabor Filter Design for Texture Segmentation. Pattern Recognit. 29, 2005–2015 (1996)
10. Shen, L., Bai, L.: A Review on Gabor Wavelets for Face Recognition. Pattern. Anal. Appl. 9, 273–292 (2006)
11. Shen, L., Bai, L.: 3D Gabor Wavelets for Evaluating SPM Normalization Algorithm. Med. Image Anal. 12, 375–383 (2008)
12. Chen, X.S., Ong, Y.S., Lim, M.H., Tan, K.C.: A Multi-Facet Survey on Memetic Computation. IEEE T. Evolut. Comput. 15, 591–607 (2011)
13. Holland, J.H.: Adaptation in Natural Artificial Systems, 2nd edn. MIT Press, Cambridge (1992)
14. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes in C. Cambridge University Press, Cambridge (1998)
15. Yu, L., Liu, H.: Efficient Feature Selection via Analysis of Relevance and Redundancy. J. Mach. Learn. Res. 5, 1205–1224 (2004)
16. Robnic-Sikonja, M., Kononenko, I.: Theoretical and Empirical Analysis of ReliefF and RReliefF. Mach. Learn. 53, 23–69 (2003)
17. Kohavi, R., John, G.H.: Wrapper for Feature Subset Selection. Artif. Intell. 97, 273–324 (1997)
18. Vapnik, V.: The Nature of Statistical Learning Theory. Springer (1995)

A Simple and Effective Immune Particle Swarm Optimization Algorithm

Wei Jiao, Weimin Cheng, Mei Zhang, and Tianli Song

China Academy of Space Technology, Youyi Road 102,
100094, Beijing, P.R. China
jiaowei1981@yahoo.com.cn

Abstract. The Particle Swarm Optimization (PSO) algorithm is a population based evolutional search strategy, which has easer implementation and fewer presetting parameters. But the most difficulty of PSO having to encounter with is premature convergence. This is due to a decrease of diversity during the evolutional process that leads to plunging into local optimum and ultimately fitness stagnation of the swarm. In order to maintain appropriate diversity, a simple and effective immune PSO (IPSO) algorithm is proposed in the paper. IPSO takes advantage of immune operators to update the particles when the algorithm fails to converge to a given threshold. The most difference of IPSO here among other optimization algorithms with immunity is that Gaussian mutation is executed before selecting particles from immune memory library. So the diversity of population is extended adequately, and the risk of trapping into local optimum is depressed effectively. Testing over the benchmark problems, the experimental results indicate the IPSO algorithm prevents premature convergence to a high degree and has better convergence performance than Standard PSO algorithm.

Keywords: Particle Swarm Optimization, immune system, immune memory, clonal selection, global search, diversity.

1 Introduction

Particle Swarm Optimization (PSO) is an evolutionary computation technique inspired by social behavior observable in nature, such as bird flocking and fish schooling, proposed by R. Eberhart and J. Kennedy in 1995 [1], [2]. The fundament for the development of PSO is hypothesis that a potential solution to an optimization problem is treated as a bird without quality and volume, flying through a D-dimensional space, adjusting its position in search space according to its own experience and that of its neighbors. Compared with other evolutionary algorithms, such as genetic algorithm (GA), PSO is easy in implementation and there are few parameters to adjust. So PSO has gained much attention and wide applications in many application areas, including electric power system [3], [4], robot [5], automatic system [6], [7] and other areas. But in real applications, premature convergence and low intelligence have become a bottleneck of PSO algorithm. This is due to a

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 489–496, 2012.

decrease of diversity in search space that leads to a total implosion and ultimately fitness stagnation of the swarm [8].

Inspired by the theory of immunology, various optimization methods, algorithms and technology were developed in recent years [9], [10], [11]. These optimization algorithms with immunity are based on immune memory, so the diversity during the process of evolution is maintained and the convergence of global optimization is guaranteed. Considering aforesaid merits, many researchers develop a hybrid algorithm based on PSO and immune principles to improve the precision and convergence of evolutional optimization algorithm [12], [13], [14].

A simple and effective Immune PSO algorithm (IPSO) based on immune memory and clonal selection is established in this paper in order to improve the performance of diversity and convergence. This new algorithm can improve global search ability to avoid falling into local optimum especially under the high dimension situation of optimization with small population size, and the characteristic of easy implementation is maintained in IPSO algorithm.

The rest of the paper is organized as follows. Section 2 describes the Standard PSO algorithm (SPSO). The basic conceptions of immune operator and the method of IPSO are expatiated in detail in Section 3. Some experimental results and the comparison analysis are presented in Section 4. Finally, Section 5 concludes some remarks.

2 Standard PSO

Assume that the search space is D-dimensional and a particle swarm consists of M particles. The ith particle is represented as $X_i = (x_{i1},x_{i2},\ldots,x_{iD})$ in D-dimensional space, where $x_{id} \in [l_d,u_d]$, $d \in [1,D]$, l_d, u_d are the lower and upper bounds of the dth dimension. The velocity of ith particle is represented as $V_i = (v_{i1},v_{i2},\ldots,v_{iD})$, which is clamped to a maximum value $V_{\max}$, specified by users. The particles' position and velocity updating rule is given by:

$$v_{id}(t+1) = \omega v_{id}(t) + c_1 rand_1[p_{id}(t) - x_{id}(t)] + c_2 rand_2[p_{gd}(t) - x_{id}(t)] \quad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (2)$$

where $i = 1,2,\ldots,M$, $d = 1,2,\ldots,D$. c_1 and c_2 are non-negative constants, generally assigned to 2.0. $rand_1$ and $rand_2$ are two independent random numbers uniformly distributed in the range of [0, 1]. $P_i = (p_{i1},p_{i2},\ldots,p_{iD})$ is the previous best position of the ith particle. $P_g = (p_{g1},p_{g2},\ldots,p_{gD})$ is the best previous position among all the particles in iteration t (memorized in a common repository). ω, named the inertia weight, usually decreases linearly from 0.9 to 0.4 during a run. This method, presented by Y. Shi and R. Eberhart [15], is called Standard Particle Swarm Optimization (SPSO) or Linearly Decreasing Weight Particle Swarm Optimization (LDW-PSO).

3 Immune Operator and IPSO

3.1 Immune Operator

Immune memory cell evolution refers to the partial cells are compartmentalized by the antibody to be joined into the memory library as the memory cell, the same and similar memory cells are eliminated. Its introduction not only provides the opportunity to effectively solve the similar question, but also provides the essential preparation for local search of the algorithm, thus improve the rate that the algorithm seek the superior solution. Immune memory cells are obtained through preserving P_g of each generation, and the size of immune memory is fixed as N_l (N_l =2M) in the paper.

Clonal selection only chooses the antibodies with higher affinity to reproduce and mutate. But the antibodies with lower affinity still exist in immune system and are gradually driven out. This makes the particle with better fitness value to be definitely chosen to participate in evolution, thus speeds up the search rate. In this paper, we choose particles (antibodies) with lower fitness (higher affinity) value from the current generation.

Inspired by immune theory that the antibodies (particles) which have higher density and lower appetency (higher fitness) are restrained and the antibodies (particles) which have lower density and higher appetency (lower fitness) are promoted. The algorithm ensures population diversity through selection probability of density base on above theory. The formula that the density and the selection probability of the ith particle is as follows [16]:

$$Density(x_i) = \frac{1}{\sum_{j=1}^{N_l} \left| f(x_i) - f(x_j) \right|}, i = 1, 2, \ldots, N_l \tag{3}$$

$$p(x_i) = \frac{\sum_{j=1}^{N_l} \left| f(x_i) - f(x_j) \right|}{\sum_{i=1}^{N_l} \sum_{j=1}^{N_l} \left| f(x_i) - f(x_j) \right|}, i, j = 1, 2, \ldots, N_l \tag{4}$$

where $Density(x_i)$ is density of the ith particle, $p(x_i)$ is selection probability of the ith particle and $f(x_i)$ is fitness value of the ith particle. Note that the optimal problem discussed in this paper is minimization problem.

3.2 Arithmetic Flow of IPSO

IPSO algorithm can be summarized in the following steps:

Step1: Initialize the state of each particle;

Step2: Evaluate the fitness value of each particle. Set P_i as the best current position of each particle, and P_g as the best current of whole population;

Step3: Update the velocity and position of particle according to equation (1) ~ equation (2);

Step4: If $V_i > V_{\max}$ or $V_i < V_{\min}$, set $V_i = V_{\max}$ or $V_i = V_{\min}$; if $X_i > X_{\max}$ or $X_i < X_{\min}$, set $X_i = X_{\max} \times rand$ or $X_i = X_{\min} \times rand$, where *rand* stands for a random number uniformly distributed in the range of [0, 1];

Step5: If necessary, update and store the individual best position and individual best fitness of each particle, update and store the global best position and global best fitness of whole population;

Step6: Generate immune memory particles (antibody). Each particle's fitness of the current population is evaluated. Set P_g as immune memory particles (antibody) to store in the immune memory library. If stopping condition is satisfied go to Step8. Otherwise, go to Step7;

Step7: If the algorithm fails to converge to a given *Threshold*, particles (antibody) are updated as following ways:

(1) N_m particles (antibody) with higher fitness (lower affinity) will be discarded, and coordinative number new particles (antibody) are generated randomly;

(2) Some particles (antibody) are selected randomly from memory library to execute Gaussian mutation according to equation (5), and then N_0 particles are picked up according to equation (3) ~ equation (4). The number of mutated particles is equal to αN_0 ($0<\alpha<1$), sign $N_0/M=\beta$;

(3) Clonal selection. N_c particles (antibody) with lower fitness (higher affinity) will be cloned (reproduced) from the current population to join in a new generated population.

Go to Step2, after a new population is generated;

Step8: Give the global best position P_g and global best fitness of whole population. Stop evolution.

Gaussian mutation operation is showed as the following equation:

$$P_g' = P_g(1+\eta) \tag{5}$$

where η is a random number generated from a standard normal distribution (mean = 0, variance = 1).

4 Simulations

4.1 Experimental Settings

A set of well-known benchmarks, which are commonly used in literatures, are used to evaluate the performance both in terms of solution quality and convergence rate of the proposed algorithms. The benchmark problems used are a set of four non-linear functions, as minimization problems, which present different difficulties to the algorithms to be evaluated. These benchmark functions are shown as follows:

Rosenbrock function: $f_1(x) = \sum_{i=1}^{n} 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$

Rastrigrin function: $f_2(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10)$

Griewank function: $f_3(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) + 1$

Ackley function: $f_4(x) = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)) + 20 - e$

Rosenbrock is a unimodal function, which has a global minimum of 0 at the point (1,…,1), it is usually used to test the local exploitation ability of algorithms. Rastrigrin, Griewank and Ackley are multimodal functions that have a large number local minima, and they have a global minimum of 0 when the vector is (0,…,0). The search space and initialization range for each test function are listed in Table 1.

Table 1. Search space and initialization range

F	Domain	Ini. Range	X_{max}	V_{max}
$f_1(x)$	[-100,100]	$(15,30)^n$	100	100
$f_2(x)$	[-10,10]	$(2.56,5.12)^n$	10	10
$f_3(x)$	[-600,600]	$(300,600)^n$	600	600
$f_4(x)$	[-30,30]	$(15,30)^n$	30	30

For all the test functions, three different dimension sizes are tested: 10, 20 and 30. The maximum number of generations is set as 1000, 1500 and 2000 corresponding to the dimensions 10, 20 and 30, respectively. In order to eliminate stochastic discrepancy, a total of 100 runs for each experimental setting were conducted. The population size, M, is only 20, inertial weight ω decreases linearly from 0.9 to 0.4 during a run. Set N_m=0.05M to mimic the extinct ratio of population during the process of clonal selection in immunology. The value of N_c is subject to N_0+N_c=M. Other presetting parameters of IPSO algorithm are listed in Table 2.

It is difficult to make a theoretical case for choosing the values of α and β, their presetting values in the algorithm are investigated experimentally.

Table 2. Presetting parameters in IPSO

F	α	β	*Threshold*
$f_1(x)$	0.5	0.8	10^{-1}
$f_2(x)$	0.5	0.8	10^{-1}
$f_3(x)$	0.3	0.5	10^{-4}
$f_4(x)$	0.3	0.5	10^{-4}

4.2 Experimental Results and Discussions

Table 3 lists the mean fitness values of the best solutions achieved by two algorithms on Rosenbrock, Rastrigrin, Griewank and Ackley functions with each experimental setting. The results of each algorithm were averaged over 100 trial runs respectively. Figure 1 shows the evolution of logarithmic average fitness of 30-dimensional test functions for SPSO and IPSO.

Table 3. Experimental values for benchmark functions

F	D	SPSO		IPSO	
		Mean	**Std.Dev**	**Mean**	**Std.Dev**
	10	49.0190	111.6057	5.4337	1.0910
$f_1(x)$	20	104.7374	177.2478	15.2165	0.2138
	30	174.3933	223.4577	25.2243	0.2136
	10	5.7908	2.6987	0.7175	1.8094
$f_2(x)$	20	23.1010	7.6722	0.0439	0.1843
	30	47.6659	10.9323	0.0240	0.1425
	10	0.1051	0.0863	0.1036	0.0649
$f_3(x)$	20	0.0344	0.0361	0.0234	0.0224
	30	0.0180	0.0245	0.0044	0.0058
	10	0.4005	2.8178	2.73e-11	4.47e-11
$f_4(x)$	20	1.0596	4.4753	0.0373	0.2145
	30	2.4599	6.3899	0.0749	0.3416

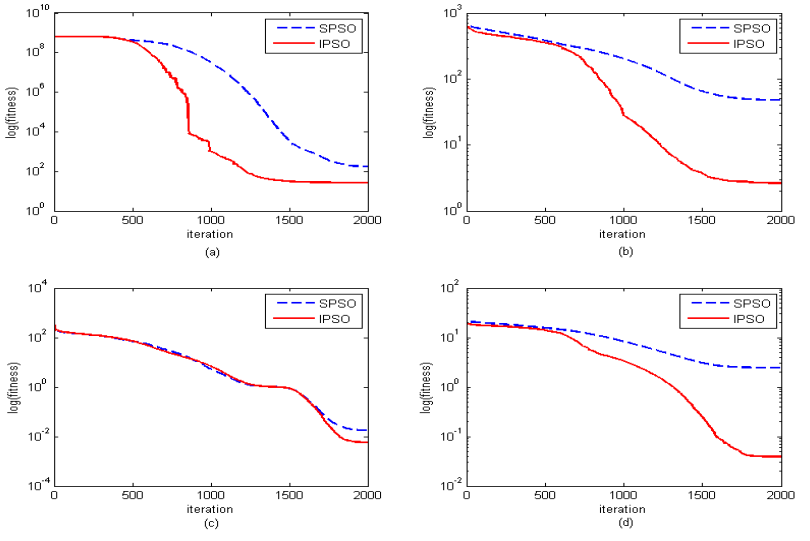


Fig. 1. Performance on 30-dimensional functions: (a) $f_1(x)$, (b) $f_2(x)$, (c) $f_3(x)$ and (d) $f_4(x)$

From the results listed in Table 3, we can see that not only the mean values of fitness are better but also the standard deviations are lower, i.e. the stochastic influence on evolution algorithm is abated in IPSO algorithm with immune operators. The experimental results indicate that IPSO algorithm exhibits good performance. It outperforms SPSO distinctly on all mentioned benchmark problems, in other words, IPSO algorithm has not only good local exploitation ability but also favorable global exploration ability. Figure 1 showing the process of iterations indicates that the evolutionary rate of convergence, especially during the latter phase of iterations, is also improved remarkably in IPSO algorithm.

5 Conclusions

Standard PSO is a simple, stochastic and global optimization technology. But its population diversity becomes worse and worse gradually with the search carrying on. So the algorithm is susceptible to falling into local optimum. Immune memory and clonal selection are introduced to update the particles when the algorithm fails to converge to a given threshold. The immune memory is used to retain the diversity of population and enhance global search capacity, and the clonal selection is adopted to accelerate the search. The results of simulation indicate that IPSO algorithm proposed in this paper improves precision of optimal solution; moreover it ensures the better convergence performance. Undoubted IPSO is an effective algorithm. The parameters that are introduced to the algorithm have effect on search performance to a certain extent, but the influence is not prominent. Further research would investigate and incorporate the mechanism of immune system, such as vaccination into PSO algorithm to increase its intelligence of search.

References

1. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceeding of IEEE International Conference on Neural Networks, Perth, Western Australia, pp. 1942–1947 (1995)
2. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, pp. 39–43 (1995)
3. Zhang, W., Liu, Y.: Adaptive Particle Swarm Optimization for Reactive Power and Voltage Control in Power Systems. In: Wang, L., Chen, K., S. Ong, Y. (eds.) ICNC 2005, Part III. LNCS, vol. 3612, pp. 449–452. Springer, Heidelberg (2005)
4. Chuanwen, J., Bompard, E.: A hybrid method of chaotic particle swarm optimization and linear interior for reactive power optimization. Mathematics and Computers in Simulation 68, 57–65 (2005)
5. Li, Y., Chen, X.: Mobile Robot Navigation Using Particle Swarm Optimization and Adaptive NN. In: Wang, L., Chen, K., S. Ong, Y. (eds.) ICNC 2005, Part III. LNCS, vol. 3612, pp. 628–631. Springer, Heidelberg (2005)
6. Su, C.-T., Wong, J.-T.: Designing MIMO controller by neuro-traveling particle swarm optimizer approach. Expert System with Applications 32, 848–855 (2007)

7. Mukherjee, V., Ghoshal, S.P.: Intelligent particle swarm optimized fuzzy PID controller for AVR system. Electric Power Systems Research 77, 1689–1698 (2007)
8. Riget, J., Vesterstrøm, J.S.: A diversity-guided particle swarm optimizer the ARPSO, University of Aarhus, Denmark, pp. 1–13, doi: http://citeseer.nj.nec.com/riget02diversityguided.html
9. de Castro, L.N., Timmis, J.: Artificial immune systems: a novel paradigm to pattern recognition. In: Artificial Neural Networks in Pattern Recognition, SOCO 2002, University of Paisley, UK, pp. 67–84 (2002)
10. Tsai, J.-T., Ho, W.-H., Liu, T.-K., Chou, J.-H.: Improved immune algorithm for global numerical optimization and job-shop scheduling problems. Applied Mathematics and Computation 194, 406–424 (2007)
11. Tan, K.C., Goh, C.K., Mamun, A.A., Ei, E.Z.: An evolutionary artificial immune system for multi-objective optimization. European Journal of Operational Research 187, 371–392 (2008)
12. Liu, J., Sun, J., Xu, W.: Quantum-Behaved Particle Swarm Optimization with Immune Operator. In: Esposito, F., Raś, Z.W., Malerba, D., Semeraro, G. (eds.) ISMIS 2006. LNCS (LNAI), vol. 4203, pp. 77–83. Springer, Heidelberg (2006)
13. Hu, C., Zeng, J., Jie, J.: Immune particle swarm optimization with diversity monitoring. Communications in Computer and Information Science 2, 380–387 (2007)
14. Zhang, R., Wu, C.: An effective immune particle swarm optimization algorithm for scheduling job shops. In: 3rd IEEE Conference on Industrial Electronics and Applications, Singapore, pp. 758–763 (2008)
15. Shi, Y., Eberhart, R.: A modified particle swarm optimizer. In: Proceedings of the IEEE Conference on Evolutionary Computation, Singapore, pp. 69–73 (1998)
16. Lu, G., Cheng, X.P., Tan, D.J.: Improvement on regulating definition of antibody density of immune algorithm. Journal of Data Acquisition & Processing 3, 44–88 (2003)

A Novel Two-Level Hybrid Algorithm for Multiple Traveling Salesman Problems

Qingsheng Yu[1], Dong Wang[2], Dongmei Lin[3], Ya Li[2], and Chen Wu[2]

[1] Department of Electronics and Information, Foshan Polytechnic, Foshan 528000, China
[2] Department of Computer, Foshan University, Foshan, Guangdong 528000, China
[3] Center of Information and Education Technology, Foshan University, Foshan, Guangdong 528000, China
{fsgarden,li-bh,wu_chk}@163.com, {wdong,dmlin}@fosu.edu.cn

Abstract. Multiple traveling salesmen problem is a NP-hard problem. The method for solving the problem must arrange with reason all cities among traveling salesman and find optimal solution for every traveling salesman. In this paper, two-level hybrid algorithm is put forward to take into account these two aspects. Top level is new designed genetic algorithm to implement city exchange among traveling salesmen with the result clustered by *k*-means. Bottom level employs branch-and-cut and Lin-kernighan algorithms to solve exactly sub-problems for every traveling salesman. This work has both the global optimization ability from genetic algorithm and the local optimization ability from branch-and-cut.

Keywords: multiple traveling salesmen problem, two-level hybrid algorithm, crossover operator, chromosome encoding.

1 Introduction

The multiple traveling salesman problem (mTSP) is a generalization of the well-known traveling salesman problem (TSP) [1-3]. It is more capable to model real life applications than TSP, since it handles more than one salesman. It is obvious that it is *NP*-hard problem also because of the fact that TSP belongs to the classical *NP*-complete problem.

In general, mTSP can be defined as follows: given a set of cities, let there be m salesmen located at a single or m depot cities. The remaining cities to be visited are called intermediate cities. Then, the mTSP consists of finding tours for all m salesmen, who all start and end at the depot city, such that each intermediate city is visited exactly once and the total cost of visiting all cities is minimized. The cost metric can be defined in terms of distance, time, cost, etc. Therefore, the problem solution consists of two levels. The first level is to determine the optimal sub-division of cities into some groups. The second level is to find the minimum length cycle for each group. Solving methods have been mostly heuristic approaches due to the complexity of the models. These methods could be classified into two broad groups, the "transformation-based" and the "direct" heuristics [2].

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 497–503, 2012.

One of two main "transformation-based" methods converts a mTSP to a classical TSP through appending some virtual cities, and then using those methods used to solve TSP to tackle it. Paul Oberlin, Sivakumar Rathinam, Swaroop Darbha (2009) [4] and Moustapha Diaby (2010) [5] utilized this method to solve unmanned aerial vehicles problem. The other method is dividing the cities into clusters and allotting each cluster to each salesman. Aristidis Likas, Nikos Vlassis, Jakob J.Verbeek (2002) [6] and LIN Dong-mei, WANG Dong, LI Ya (2010) [7] suggested clustering cities by using k-means clustering. The original mTSP would be divided into some sub-problems, and the complexity of mTSP is highly degenerated since these sub-problems could be tackled using various algorithms for TSP.

The "direct" methods tackle the problem in its natural form. Arthur. E. Carter, Cliff. T. Ragsdale (2007) proposes new set of chromosomes and related operators and compares theoretical properties and computational performance of the proposed technique [8]. Ding Chao, Cheng ye, He Miao (2007) have developed a two level genetic algorithm which favors neither intra-cluster path or inter-cluster path. The set of cities which are given in the graph were portioned into clusters and now this problem is converted to a cluster TSP [9]. Hannes Schabauer, et.al. (2005) have worked on to solve TSP heuristically by the parallelization of self-organizing maps on cluster architectures [10]. Klaus Meer (2007) has worked on the simulated annealing methodologies and has proved that this algorithm outperforms any metropolis or standard algorithm [11].

As we know, it is an effective method to combine meta-heuristic algorithms with heuristic or approximate algorithms for TSP. In this paper, we established a new two level hybrid algorithm for multi-depot and fixed-destination mTSPs. In the new hybrid algorithm, we employed k-means algorithm, genetic algorithm, branch-and-cut algorithm and Lin-kernighan algorithm. The new algorithm could find better solutions of those mTSP whose city number of each group is less than 200 cities.

2 Hybrid Solving Strategy and Algorithm

2.1 Research Gap

As mentioned above, we found that it is difficult to exchange the cities among the groups in both the "transformation-based" heuristic and the "direct" heuristic. In the "transformation-based" heuristics, the work to balance the number of cities visited by traveling salesman is too complex while appending some "virtual cities" and in the meanwhile the complexity of problems is increased because of new "virtual cities", and the clustering method losses the global superiority due to blocking the city exchange among traveling salesman. In the "direct" heuristics, the global superiority of these algorithms is better than most of "transformation-based" heuristics because all cities in problems are to be optimized as a whole, but some complicated strategies must be established to control the exchange of cities among traveling salesmen.

Our work proposed that different algorithms are employed to tackle different problems existing in different phase while solving a mTSP. We can receive good results if combining the global search capability for the problems and the local optimum search capability for sub-problems.

2.2 Hybrid Solving Strategy

The new hybrid solving strategy in this paper includes two levels also. The top level includes two stages, utilizing k-means algorithm to initialize the group of cities for each traveling salesman and using genetic algorithm to complement the city exchange among traveling salesmen according to evolutionary strategy. The bottom level mainly uses branch-and-cut algorithm to solve sub-problems which could be solved as TSP. Lin-kernighan algorithm is used also in the bottom level to improve the efficiency of the branch-and-cut algorithm and to expand the solution scale of the algorithm. The flow chart of new hybrid strategy is as Fig. 1.

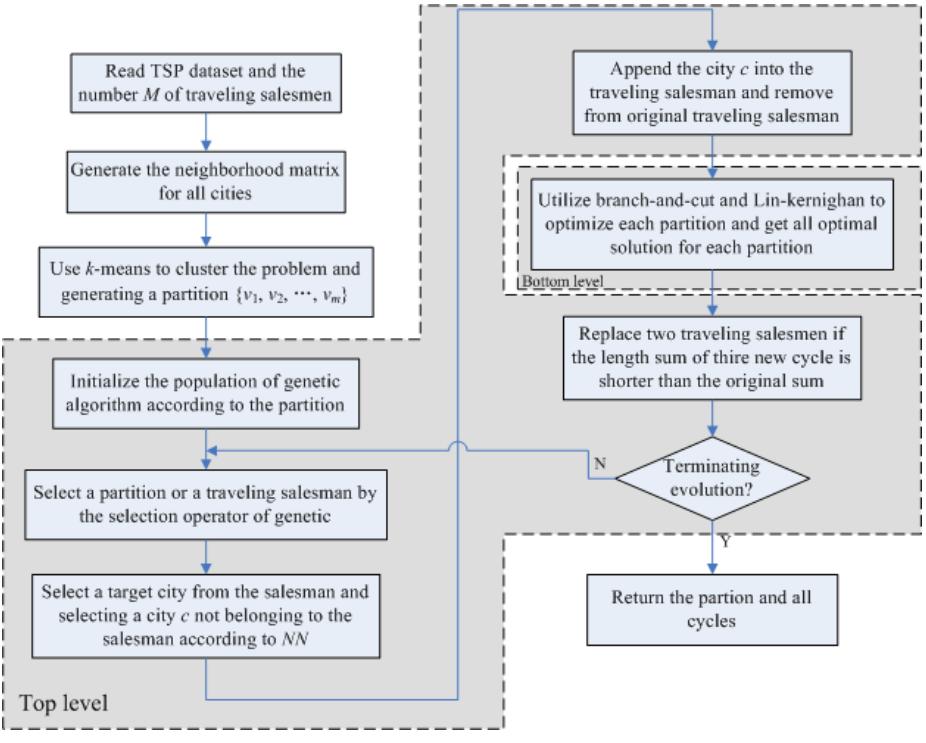


Fig. 1. The flow chart of new hybrid strategy

3 Hybrid Solving Algorithm

3.1 Genetic Algorithm in New Strategy

In the new solving strategy, the genetic algorithm located at top level is as the main framework for exchanging the cities among traveling salesman, so new evolutionary strategy and genetic operators must be redesigned to fit for our work.

Chromosome encoding: In order to provide equal opportunities for every traveling salesman, individual encoding is implemented based on traveling salesmen. Each traveling salesman is encoded by integer, starting at 0. The length of chromosome is equal to the number of traveling salesmen and each gene represents a traveling salesman code.

Selection operator: If using normal selection operator of genetic algorithm, we would get two individuals in population, but we can not ensure that the two traveling salesmen are adjacent in space. Therefore, the operator is designed as randomly selecting an individual p_i from the population $\{p_1,p_2,\ldots,p_n\}$, n is the population number, and selecting randomly a traveling salesman t_1 from p_i, remarking $C(t_1)$ as the city set of t_1, and then selecting randomly a city c_1 from $C(t_1)$, finally selecting randomly an adjacent city c_2 according to the adjacency matrix and $c_2 \notin C(t_1)$.

Crossover operator: Assumed that $c_2 \in C(t_2)$, $t_2 \in p_j$, $p_j \in \{p_1,p_2,\ldots,p_n\}$, the crossover operation is $C(t_1) \cup \{c_2\} \rightarrow C_1$ and $C(t_2)-\{c_2\} \rightarrow C_2$.

Evolutionary strategy: Assumed that $L[C(t)]$ is the optimal length of $C(t)$, replacing $C(t_1)$ with C_1 and $C(t_2)$ with C_2 if $(L[C_1]+L[C_2])<(L[C(t_1)]+L[C(t_2)])$.

There is no mutation operator in new genetic algorithm because the operation could provide very small probability to improve the solution just exchanging two cities between two selected traveling salesmen.

3.2 Two-Level Hybrid Solving Algorithm

Improved branch-and-cut algorithm is used to improve its performance [12], so new algorithm must establish the dataset of each sub-problem and branch-and-cut works on a subset of the problem. The new algorithm is as follows.

Algorithm name: Two-level hybrid algorithm for mTSP
Input: The number of traveling salesmen m and a mTSP dataset
Output: All cycles of traveling salesmen
Begin
1. Initialize
   1.1 Generate neighborhood matrix *NM* for all cities
   1.2 Clustering all cities according to m using k-means algorithm, and get a partition $\{p_1, p_2, \ldots, p_m\}$
   1.3 Initialize the population of genetic algorithm
      1.3.1 Allocate the space of population
      1.3.2 Initialize randomly individuals (**Section 3.1**: chromosome encoding)
2. Loop to termination condition, do
   2.1 Perform selection operation (**Section 3.1**: selection operator)
      2.1.1 Select randomly an individual p from the population
      2.1.2 Select randomly a traveling salesman t_1 from p
      2.1.3 Select randomly a city c_1 from t_1
      2.1.4 Select randomly another city c_2 by *NM*
   2.2 Perform crossover operation (**Section 3.1**: crossover operator)

2.3 Perform evolutionary strategy (**Section 3.1**: evolutionary strategy)
2.3.1 Construct two sub-problem datasets C_1 and C_2
2.3.2 Simplify two initial edge-sets of C_1 and C_2
2.3.3 Call branch-and-cut to compute $L(C_1)$ and $L(C_2)$
2.3.4 Replace $C(t_1)$ with C_1 and $C(t_2)$ with C_2 if $(L[C_1]+L[C_2])<(L[C(t_1)]+L[C(t_2)])$
3. Return all cycles of traveling salesmen
End.

4 Experimental Simulation and Analyze

The experimental environment is Intel T8300 2.39 GHz microprocessor, 4GB RAM, Microsoft Windows XP operating system. Experimental datasets are from TSPLIB95 [13]. Each of the datasets is repeated for 30 times, and is computed average values of the best tour length, because *k*-means algorithm is a kind of random algorithm also. In the new algorithm, the probability of crossover is 0.35, the number of population is 10, and the generation of evolution is 10. The compared algorithm is from [7] because the results are better than other similar algorithms. The algorithm includes two levels also, *k*-means algorithm at its top level and branch-and-cut algorithm at bottom level. The experimental results are shown in Table 1 and Table 2.

Table 1. Group 1 of two algorithms comparison experiment

Dataset Name	Cities	Traveling Salesmen	Algorithm from [7]			New algorithm		
			Best	Average	Time	Best	Average	Time
pr76	76	5	117 313	117 656	0.341s	115 390	116 414	4.500s
pr152	152	5	64 226	64 266	6.903s	64 226	64 226	105.311s
pr226	226	5	82 533	82 533	4.908s	82 533	82 533	88.281s
pr299	299	5	51 462	52 500	2.144s	50 492	51 225	40.037s
pr439	439	5	112 569	113 087	4.330s	112 196	112 910	77.806s
pr1002	1 002	5	264 558	264 558	9.462s	262 428	262 836	213.263

Table 2. Group 2 of two algorithms comparison experiment

Dataset Name	Cities	Traveling Salesmen	Algorithm from [7]			New algorithm		
			Best	Average	Time	Best	Average	Time
		2	434	435.40	0.235s	432	432.43	4.917s
eil51	51	3	447	448.60	0.234s	441	443.00	4.099s
		4	457	458.00	0.222s	448	449.50	2.563s
		2	558	559.10	0.378s	555	555.00	12.731s
eil76	76	3	563	569.30	0.316s	552	556.50	5.201s
		4	565	566.40	0.313s	559	560.50	4.198s
		2	651	651.90	0.633s	644	644.00	24.188s
eil101	101	3	644	651.90	0.486s	640	644.83	11.391s
		4	655	665.40	0.485s	650	653.33	9.193s

We can know from the experimental results according to Table1 and Table2.

1) Although new algorithm spends more time than the algorithm from [7] because new algorithm adapts the framework of genetic algorithm and the algorithm from [7] computes all sub-problems after clustering, new algorithm could find higher-quality solutions. The convergence performance analyze is shown in Figure 2.

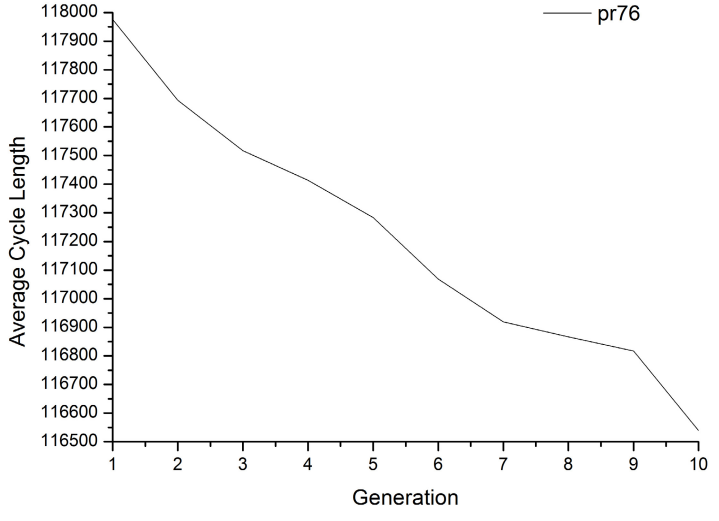


Fig. 2. Convergence performance analyze of pr76 from TSPLIB95

2) These are two special datasets with the classification stability, pr152 and pr226, so the clustering results for each time are the same. In the meanwhile, No city exchange could improve the quality of solutions, therefore the optimal solution does not change.
3) It also reveals that there is a certain space to improve the solution after clustering. The main reason is that it is difficult for clustering algorithms to take into account the global superiority of mTSP.

5 Conclusions

Our work mainly attempts to establish a kind of genetic algorithm used for exchanging cities among traveling salesmen. The new genetic algorithm utilizes the clustering results from *k*-means algorithm and uses new operators and strategy to implement the exchange function. For every single traveling salesman, the branch-and-cut and Lin-kernighan algorithms are employed to exactly solve, just as TSP. The experimental results show that new algorithm can find higher-quality solutions. The next work could focus on establishing genetic operators or evolutionary strategy to increase the genetic algorithm global search capability.

Acknowledgments. This paper is supported by Natural Science Foundation of Guangdong Province (10152800001000029) and Guangdong Province scientific and technological project (2011B010200031).

References

1. Applegate, D.L., Bixby, R.E., Chvátal, V., Cook, W.J.: The Traveling Salesman Problem: A Computational Study. Princeton University Press, Princeton (2006)
2. Bektas, T.: The multiple traveling salesman problem: an overview of formulations and solution procedures. Omega: The International Journal of Management Science 34(3), 209–219 (2006)
3. Greco, F.: Traveling Salesman Problem. InTech (2008)
4. Oberlin, P., Rathinam, S., Darbha, S.: A transformation for a heterogeneous, multiple depot, multiple traveling salesman problem. In: Proceedings of the 2009 Conference on American Control Conference, pp. 1292–1297. IEEE Press, Piscataway (2009)
5. Diaby, M.: Linear Programming Formulation of the Multi-Depot Multiple Traveling Salesman Problem with Differentiated Travel Costs. In: Davendra, D. (ed.) Traveling Salesman Problem, Theory and Applications. InTech (2010)
6. Likas, A., Vlassis, N.A., Verbeek, J.J.: The global k-means clustering algorithm. Pattern Recognition 36(2), 451–461 (2003)
7. Lin, D.-M., Wang, D., Li, Y.: Two-level degradation hybrid algorithm for multiple traveling salesman problem. Application Research of Computers 28(8), 2876–2879 (2010)
8. Rizzoli, A.E., Montemanni, R., Lucibello, E., Gambardella, L.M.: Ant Colony Optimization for real world vehicle routing problems. Swarm Intelligence 1(2), 135–151 (2007)
9. Ding, C., Cheng, Y., He, M.: Two level genetic algorithm for clustered Travelling Salesman Problem with Application in large scale TSPs. Tsinghua Science and Technology 12(4), 459–465 (2007)
10. Schabauer, H., Schikuta, E., Weishaupl, T.: Solving Very Large Travelling Salesman Problems by SOM parallelization on cluster architecture. In: Proceedings of the Sixth International Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT), pp. 954–958 (2005)
11. Meer, K.: Simulated Annealing Versus Metropolis for a TSP instance. Information Processing Letters 104(6), 216–219 (2007)
12. Wang, D., Wu, X.-B., Mao, X.-C., Liu, W.-J.: Accurate solving hybrid algorithm for small scale TSP. Systems Engineering and Electronics 30(9), 1693–1696 (2008)
13. Traveling Salesman Problems Library, `http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/`

On the Performance Metrics of Multiobjective Optimization*

Shi Cheng[1,2], Yuhui Shi[2], and Quande Qin[3]

[1] Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool, UK
[2] Department of Electrical & Electronic Engineering, Xi'an Jiaotong-Liverpool University, Suzhou, China
shi.cheng@liverpool.ac.uk, yuhui.shi@xjtlu.edu.cn
[3] College of Management, Shenzhen University, Shenzhen, China

Abstract. Multiobjective Optimization (MOO) refers to optimization problems that involve two or more objectives. Unlike in the single objective optimization, a set of solutions representing the tradeoff among the different objects rather than an unique optimal solution is sought in MOO. How to measure the goodness of solutions and the performance of algorithms is important in MOO. In this paper, we first review the performance metrics of multiobjective optimization and then classify variants of performance metrics into three categories: set based metrics, reference point based metrics, and the true Pareto front/set based metrics. The properties and drawbacks of different metrics are discussed and analyzed. From the analysis of different metrics, an algorithm's properties can be revealed and more effective algorithms can be designed to solve MOO problems.

Keywords: Multiobjective Optimization, Performance Metrics, Pareto Front/Set, Reference Point.

1 Introduction

An optimization problem in $\mathbb{R}^n$, or simply an optimization problem, is a mapping $f : \mathbb{R}^n \to \mathbb{R}^k$, where $\mathbb{R}^n$ is termed as decision space (or parameter space, problem space), and $\mathbb{R}^k$ is termed as objective space. Optimization problems can be divided into two categories based on the value of k. If $k = 1$, the problems are called Single Objective Problems (SOPs); if $k > 1$, problems are called Multiobjective Problems (MOPs), and specially, problems are called Many Objective Problems when k is large than 2 or 3 [1].

One of the main differences between SOPs and MOPs is that MOPs constitute a multidimensional objective space. In addition, a set of solutions representing the tradeoff among the different objectives rather than an unique optimal solution is sought in Multiobjective optimization (MOO). How to measure the goodness of solutions and the performance of algorithms is important in MOO.

* The authors' work is partially supported by National Natural Science Foundation of China under grant No.60975080.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 504–512, 2012.

Although many articles have discussed metrics on multiobjective optimization [5,8,14], there is no one metric that can overwhelm others. It is necessary to have more analyses and discussions. In this paper, we classify variants of performance metrics into three categories: set based metrics, reference point based metrics, and the true Pareto front/set based metrics. The properties and drawbacks of different metrics are discussed and analyzed. From the analyses of the metrics, more effective algorithms can be designed to solve multiobjective problems.

This paper is organized as follows. Section 2 reviews the basic definitions of multiobjective optimization. Section 3 introduces set based metrics, which include outperformance relations, $\mathcal{C}$ measure, and $\mathcal{M}_3$ measure. Section 4 introduces a reference point based metrics, which include $\mathcal{S}$ measure (hypervolume) and $\mathcal{D}$ measure. Section 5 introduces the true Pareto front/set based metrics, which include inverted generational distance metric, hypervolume difference metric and spacing Δ metric. Section 6 concludes with some remarks and future research directions.

2 Multiobjective Optimization

A general *multiobjective optimization problem* can be described as a vector function $\mathbf{f}$ that maps a tuple of n parameters (decision variables) to a tuple of m objectives. Without loss of generality, minimization is assumed throughout this paper.

$$\begin{aligned} \text{minimize} \quad & \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_m(\mathbf{x})) \\ \text{subject to} \quad & \mathbf{x} = (x_1, x_2, \cdots, x_n) \in \mathbf{X} \\ & \mathbf{y} = (y_1, y_2, \cdots, y_m) \in \mathbf{Y} \end{aligned}$$

where $\mathbf{x}$ is called the *decision vector*, $\mathbf{X}$ is the *decision space*, $\mathbf{y}$ is the *objective vector*, and $\mathbf{Y}$ is the *objective space*, and $\mathbf{f} : \mathbf{X} \to \mathbf{Y}$ consists of m real-valued objective functions.

Let $\mathbf{u} = (u_1, \cdots, u_m)$, $\mathbf{v} = (v_1, \cdots, v_m) \in \mathbf{Y}$, be two vectors, $\mathbf{u}$ is said to dominate $\mathbf{v}$ (denoted as $\mathbf{u} \preceq \mathbf{v}$), if $u_i \leq v_i, \forall i = 1, \cdots, m$, and $\mathbf{u} \neq \mathbf{v}$. A point $\mathbf{x}^* \in \mathbf{X}$ is called Pareto optimal if there is no $\mathbf{x} \in \mathbf{X}$ such that $\mathbf{f}(\mathbf{x})$ dominates $\mathbf{f}(\mathbf{x}^*)$. The set of all the Pareto optimal points is called the *Pareto set* (denoted as PS). The set of all the Pareto objective vectors, $PF = \{f(x) \in X | x \in PS\}$, is called the *Pareto front* (denoted as PF).

In a multiobjective optimization problem, we aim to find the set of optimal tradeoff solutions known as the Pareto optimal set. Pareto optimality is defined with respect to the concept of nondominated points in the objective space.

The optimization goal of an MOP consists of three objectives: (1) The distance of the resulting nondominated solutions to the true optimal Pareto front should be minimized; (2) A good (in most cases uniform) distribution of the obtained solutions is desirable; (3) The spread of the obtained nondominated solutions should be maximized, i.e., for each objective a wide range of values should be covered by the nondominated solutions.

3 Set Based Metrics

In multiobjective optimization, a set of solutions representing the tradeoff among the different objectives rather than an unique optimal solution as sought. It's a straightforward way to measure solutions on set based metrics. These metrics are a kind of quality measures, which are difficult to measure the goodness of solutions.

3.1 Outperformance Relations

Three kinds of outperformance relations are introduced in [4] to express the relations between two sets of internally nondominated objective vectors. The relations are as follow:

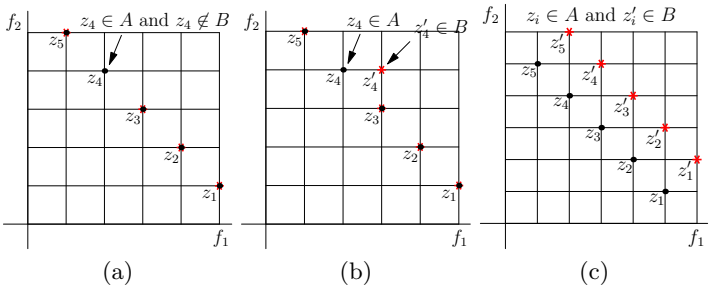


Fig. 1. Examples of outperformance relations, $\bullet \in \mathbf{A}$, and $* \in \mathbf{B}$: (a) $\mathbf{A}$ weak outperformance $\mathbf{B}$, (b) $\mathbf{A}$ strong outperformance $\mathbf{B}$, (c) $\mathbf{A}$ complete outperformance $\mathbf{B}$

- Weak outperformance: $\mathbf{ND}(\mathbf{A} \cup \mathbf{B}) = \mathbf{A}$ and $\mathbf{A} \neq \mathbf{B}$. $\mathbf{A}$ *weakly outperforms* $\mathbf{B}$ if all solutions in $\mathbf{B}$ are *contained* in $\mathbf{A}$ and there is at least one solution in $\mathbf{A}$ that is not contained in $\mathbf{B}$, e.g. Fig. 1 (a).
- Strong outperformance: $\mathbf{ND}(\mathbf{A} \cup \mathbf{B}) = \mathbf{A}$ and $\mathbf{B} \backslash \mathbf{ND}(\mathbf{A} \cap \mathbf{B}) \neq \emptyset$. $\mathbf{A}$ *strongly outperforms* $\mathbf{B}$ if all solutions in $\mathbf{B}$ are equal to or dominated by solutions in $\mathbf{A}$ and there exists at least one solution in $\mathbf{B}$ that is dominated by solutions in $\mathbf{A}$, e.g. Fig. 1 (b).
- Complete outperformance: $\mathbf{B} \cap \mathbf{ND}(\mathbf{A} \cup \mathbf{B}) = \emptyset$. $\mathbf{A}$ *completely outperforms* $\mathbf{B}$ if each solution in $\mathbf{B}$ is dominated by solutions in $\mathbf{A}$, e.g. Fig. 1 (c).

where $\mathbf{ND}()$ denotes the set includes all nondominated solutions.

3.2 $\mathcal{C}$ Measure

The $\mathcal{C}$ measure indicates the coverage of two sets [13]. This measure compares two sets of solutions and calculates the proportion of solutions in the second set for which there are solutions at least as good in every objective in the first set.

The definition of $\mathcal{C}$ measure is as follows: Let $\mathbf{A}, \mathbf{B} \subseteq \mathbf{X}$ be two sets of decision vectors. The function $\mathcal{C}$ maps the ordered pair $(\mathbf{A}, \mathbf{B})$ to the interval $[0, 1]$:

$$\mathcal{C}(\mathbf{A}, \mathbf{B}) := |\{\mathbf{b} \in \mathbf{B} \,|\, \exists \mathbf{a} \in \mathbf{A} : \mathbf{a} \preceq \mathbf{b}\}|/|B| \tag{1}$$

The value $\mathcal{C}(\mathbf{A}, \mathbf{B}) = 1$ means that all decision vectors in $\mathbf{B}$ are at least weakly dominated by $\mathbf{A}$. The opposite, $\mathcal{C}(\mathbf{A}, \mathbf{B}) = 0$, represents the situation when none of the points in $\mathbf{B}$ are weakly dominated by $\mathbf{A}$. Note that always both directions have to be considered, since $\mathcal{C}(\mathbf{A}, \mathbf{B})$ is not necessarily equal to $1 - \mathcal{C}(\mathbf{B}, \mathbf{A})$.

The $\mathcal{C}$ measure has some drawbacks: (1) It cannot measure the *subset* relation. In Fig. 2. (a), Set $\mathbf{A}$ *includes* Set $\mathbf{B}$, however, values of $\mathcal{C}$ measure are both zero; (2) If solutions in set $\mathbf{A}$ not dominated by solutions in set $\mathbf{B}$, while vice versa, the value of $\mathcal{C}$ measure is zero, e.g., Fig. 2. (b); (3) The magnitude of solutions is not considered. In Fig. 2. (c), the result of $\mathcal{C}$ measure is not obeyed the intuition.

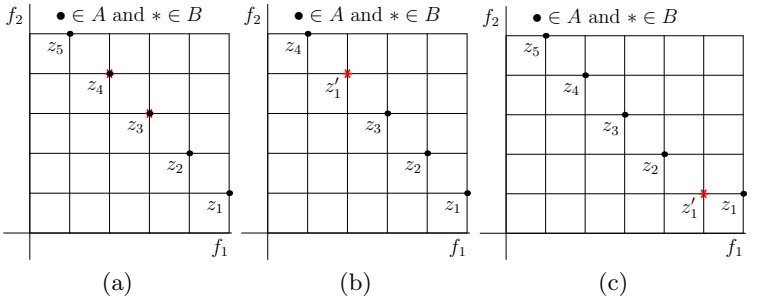


Fig. 2. Drawbacks of $\mathcal{C}$ measure: (a) Set $\mathbf{B}$ is a subset of set $\mathbf{A}$: $\mathcal{C}(\mathbf{A}, \mathbf{B}) = 0$, $\mathcal{C}(\mathbf{B}, \mathbf{A}) = 0$; (b) $\mathbf{A} \not\prec \mathbf{B}$ and $\mathbf{B} \not\prec \mathbf{A}$: $\mathcal{C}(\mathbf{A}, \mathbf{B}) = 0$, $\mathcal{C}(\mathbf{B}, \mathbf{A}) = 0$; (c) Different number of element in each set: $\mathcal{C}(\mathbf{A}, \mathbf{B}) = 0$, $\mathcal{C}(\mathbf{B}, \mathbf{A}) = 1/5$

The above metrics are a kind of quality measure. It shows the relations of two sets, however, in most cases, two solutions both have part of non-dominated solutions. The set based metrics are difficult to utilize in that situation.

3.3 Function $\mathcal{M}_3$

The function $\mathcal{M}_3$ is a spread metric, which measures the spread of the solutions set $\mathbf{A}$ in decision space or the spread of the obtained nondominated solutions $\mathbf{U}$ in objective space [13].

$$\mathcal{M}_3(\mathbf{A}) = \sqrt{\sum_{i=1}^{n} \max\{\|a_i - b_i\| \; |\mathbf{a}, \mathbf{b} \in \mathbf{A}\}} \tag{2}$$

$$\mathcal{M}_3^*(\mathbf{U}) = \sqrt{\sum_{i=1}^{n} \max\{\|u_i - v_i\| \; |\mathbf{u}, \mathbf{v} \in \mathbf{U}\}} \tag{3}$$

The function $\mathcal{M}_3$ ignores the magnitude of solutions.

4 Reference Point Based Metrics

The reference points based metrics are mostly used in multiobjective optimization. Through these metrics, the goodness of solutions is measured by a single

scalar. These metrics are easy in concept and efficient in calculation, however, these metrics are sensitive to the choice of the reference point, and a solution in different part of Pareto front plays different role in the scalar calculation.

4.1 $\mathcal{S}$ Measure (Hypervolume)

A favored metric is *hypervolume*, also known as the $\mathcal{S}$ measure [13] or Lebesgue measure. The hypervolume is a measure of how much of the objective space is weakly dominated by a given nondominated set. i.e., it measures the size of the portion of objective space that is dominated by these solutions collectively.

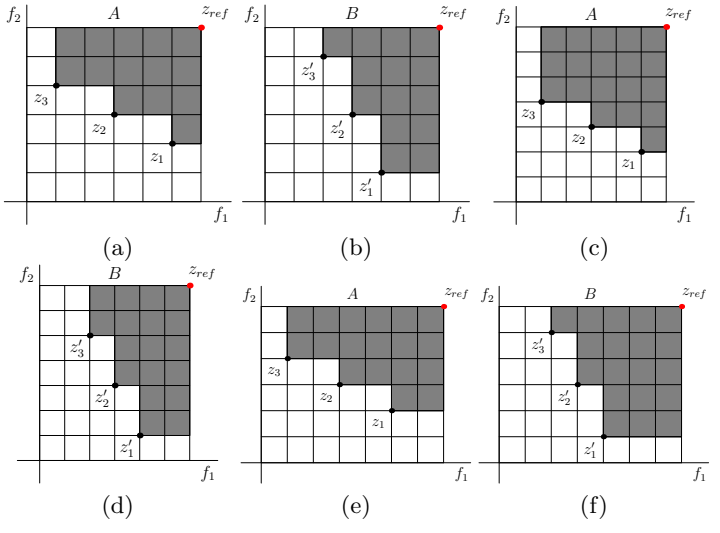


Fig. 3. The relative value of the $\mathcal{S}$ metric depends upon an arbitrary choice of reference point z_{ref}. Two nondominated sets are shown, A and B, in Fig. (a) and (b) $\mathcal{S}(A) = \mathcal{S}(B)$, in Fig. (c) and (d) $\mathcal{S}(A) > \mathcal{S}(B)$, and in Fig. (e) and (f) $\mathcal{S}(A) < \mathcal{S}(B)$. The same sets have a different ordering in $\mathcal{S}$ caused by a different choice of z^{ref}.

Generally, hypervolume is favored because it captures in a single scalar both the closeness of the solutions to the optimal set and, to some extent, the spread of the solutions across objective space. Hypervolume also has nicer mathematical properties than many other metrics; although it is difficult to calculate the accurate value of hypervolume, many fast algorithms are proposed to get an approximate scalar [10,11]. Also, it has been proved that hypervolume is maximized if and only if the set of solutions contains only Pareto optima.

Hypervolume has some nonideal properties:

– It is sensitive to the choice of reference point. Fig. 3 displays that the same sets have a different ordering in $\mathcal{S}$ caused by a different choice of z^{ref} [5,6].

– Extreme points play an important role than points in the middle of the Pareto front. For example, in Fig. 3 (a), z_3 is more important than z_2, and in Fig. 3 (b). z'_1 is worth more than z'_2.
– Hypervolume is expensive to calculate, an approach needs to be designed to approximate it within a reasonable error [11].

4.2 $\mathcal{D}$ Measure

The $\mathcal{D}$ measure indicates the coverage difference of two sets [13]. This measure combines the C measure and hypervolume measure.

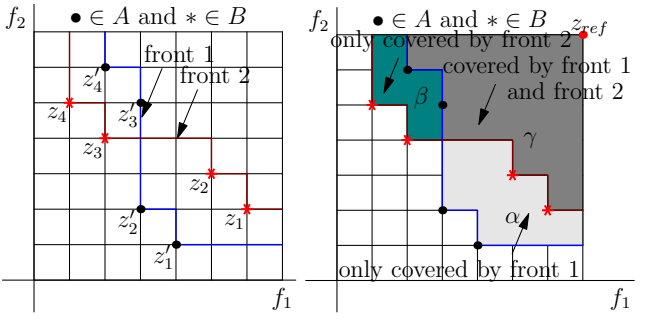


Fig. 4. The comparison between $\mathcal{C}$ measure and $\mathcal{D}$ measure. (a) the $\mathcal{C}$ measure: $\mathcal{C}(\mathbf{A}, \mathbf{B}) = \mathcal{C}(\mathbf{B}, \mathbf{A}) = 1/2$; (b) The $\mathcal{D}$ measure: $\mathcal{D}(\mathbf{A}, \mathbf{B}) = \alpha$, $\mathcal{D}(\mathbf{B}, \mathbf{A}) = \beta$, and $\mathcal{D}(\mathbf{A}, \mathbf{B}) > \mathcal{D}(\mathbf{B}, \mathbf{A})$.

The definition of $\mathcal{D}$ measure is as follows: Let $\mathbf{A}, \mathbf{B} \subseteq \mathbf{X}$ be two sets of decision vectors. The function $\mathcal{D}$ is defined by

$$\mathcal{D}(\mathbf{A}, \mathbf{B}) := \mathcal{S}(\mathbf{A} + \mathbf{B}) - \mathcal{S}(\mathbf{B}) \tag{4}$$

and gives the size of the space weakly dominated by $\mathbf{A}$ but not weakly dominated by $\mathbf{B}$ (regarding the objective space).

As shown in Fig. 4, (a) is for $\mathcal{C}$ measure, (b) is for $\mathcal{D}$ measure. There is the area of size α that is covered by front 1 but not by front 2; and area of size β that is covered by front 2 but not by front 1. The dark-shaded area (of size γ) is covered by both front in common. It holds that $\mathcal{D}(\mathbf{A}, \mathbf{B}) = \alpha$, and $\mathcal{D}(\mathbf{B}, \mathbf{A}) = \beta$.

In this example, $\mathcal{D}(\mathbf{B}, \mathbf{A}) > \mathcal{D}(\mathbf{A}, \mathbf{B})$ which reflects the quality difference between the two fronts in contrast to the $\mathcal{C}$ metric. In addition, the $\mathcal{D}$ measure gives information about whether either set entirely dominates the other set, e.g., $\mathcal{D}(\mathbf{A}, \mathbf{B}) = 0$ and $\mathcal{D}(\mathbf{B}, \mathbf{A}) > 0$ means that $\mathbf{A}$ is dominated by $\mathbf{B}$.

The $\mathcal{D}$ measure is based on the hypervolume calculation. It is sensitive to the choice of reference point. Fig. 5 displays that the same sets have a different ordering in $\mathcal{D}$ caused by a different choice of z^{ref}.

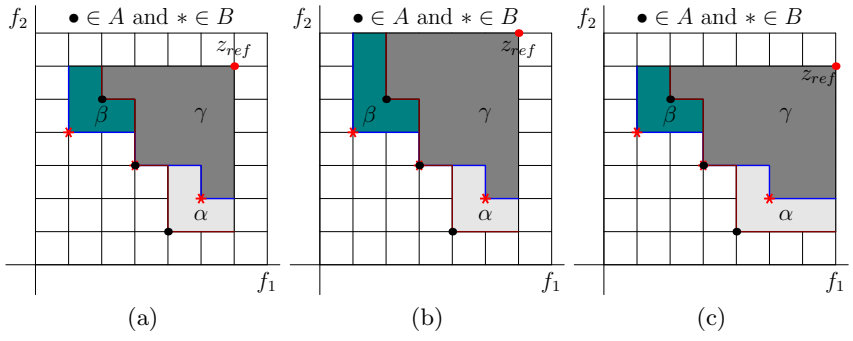


Fig. 5. The relative value of the $\mathcal{D}$ metric depends upon an arbitrary choice of reference point z^{ref}. In Fig.(a) $\mathcal{D}(A,B) = \mathcal{D}(B,A)$, (b) $\mathcal{D}(A,B) < \mathcal{D}(B,A)$, (c) $\mathcal{D}(A,B) > \mathcal{D}(B,A)$.

5 True Pareto Front/Set Based Metrics

True Pareto front based metrics compares the distribution of Pareto front found by the search algorithm and the true Pareto front. This kind of metrics is only utilized on benchmark problems, because the true Pareto front is unknown for real-world problems. However, utilizing these metrics, the search efficiency of different algorithms can be compared.

5.1 Inverted Generational Distance (IGD)

One frequently used metric is Inverted generational distance (IGD) [7, 12] also known as reverse proximity indicator (RPI) [2, 9], or the convergence metric γ [3]. It measures the extent of convergence to a known set of Pareto-optimal solutions.

The definition of this metric is as follows: Let P^* be a set of uniformly distributed Pareto-optimal points in the PF (or PS). Let $\mathbf{P}$ be an approximation to the PF (or the PS). The IGD metric is defined as follows:

$$IGD(\mathbf{P}^*, \mathbf{P}) = \sum_{v \in \mathbf{P}^*} d(v, \mathbf{P}) / |\mathbf{P}^*| \tag{5}$$

where $d(v, \mathbf{P})$ is the minimum Euclidean distance between v and all of the points in the set $\mathbf{P}$; and $|\mathbf{P}^*|$ is the cardinality of $\mathbf{P}^*$. In this metric, the number of solutions in $\mathbf{P}$ should be large enough to obtain an accurate result.

The IGD metric can be utilized both in solution space and objective space. In objective space, $\mathbf{P}^*$ is a set of points in the PF and $d(v, \mathbf{P})$ is the minimum distance between fitness values of solutions and the Pareto front. While in decision space, $\mathbf{P}^*$ is a set of points in the PS and $d(v, \mathbf{P})$ is the minimum distance between solutions and the Pareto set.

5.2 Hypervolume Difference (I_H^-) Metric

The Hypervolume difference I_H^- metric is defined as

$$I_H^-(\mathbf{P}^*, \mathbf{P}) = I_H(\mathbf{P}^*) - I_H(\mathbf{P}) \tag{6}$$

where $I_H(\mathbf{P}^*)$ is the hypervolume between the true Pareto front $\mathbf{P}^*$ and a reference point, and $I_H(\mathbf{P})$ is the hypervolume between the obtained Pareto front $\mathbf{P}$ and the same reference point. The hypervolume difference measure is also based on the hypervolume calculation. The result may be different by the choice of reference point.

Both the IGD metric and the I_H^- metric measure convergence and diversity. To have low IGD and I_H^- values, $\mathbf{P}$ must be close to the PF (or PS) and cannot miss any part of the whole PF (or PS) [12].

5.3 Spacing Δ Metric

The spacing Δ metric measures the extent of spread achieved among the obtained solutions [3]. The following metrics is utilized to calculate the nonuniformity in the distribution:

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{|\mathbf{P}|-1} |d_i - \bar{d}|}{d_f + d_l + (|\mathbf{P}| - 1)\bar{d}} \tag{7}$$

where d_i is the Euclidean distance between consecutive solutions in the obtained nondominated set of solutions $\mathbf{P}$, d_f and d_l are the distances between the extreme solutions in true Pareto front and the boundary solutions of $\mathbf{P}$. $\bar{d}$ is the average of all distance d_i, $i \in [1, |\mathbf{P}| - 1]$.

6 Conclusions

Multiobjective Optimization refers to optimization problems that involve two or more objectives, and a set of solutions is obtained instead of one. How to measure the goodness of solutions and the performance of algorithms is important in multiobjective Optimization.

In this paper, we reviewed variants of performance metrics and classified them into three categories: set based metric, reference point based metric, and the true Pareto front/set based metric. The properties and drawbacks of different metrics are discussed and analyzed.

A proper metric should be chosen under different situations, and on the contrary, an algorithm's ability can be measured by different metrics. An algorithm's properties can be revealed through different metrics analysis on different problems, then different algorithms can be utilized in an appropriate situation. From the analysis of different metrics, an algorithm's properties can be revealed and more effective algorithms can be designed to solve MOO problems.

References

1. Adra, S.F., Fleming, P.J.: Diversity management in evolutionary many-objective optimization. IEEE Transactions on Evolutionary Computation 15(2), 183–195 (2011)
2. Bosman, P.A.N., Thierens, D.: The balance between proximity and diversity in multiobjective evolutionary algorithms. IEEE Transactions on Evolutionary Computation 7(2), 174–188 (2003)
3. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2), 182–197 (2002)
4. Hansen, M.P., Jaszkiewicz, A.: Evaluating the quality of approximations to the non-dominated set. Tech. Rep. MM-REP-1998-7, Technical University of Denmark (March 1998)
5. Knowles, J.D., Corne, D.: On metrics for comparing non-dominated sets. In: Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002), pp. 711–716 (2002)
6. Knowles, J.D., Corne, D.: Properties of an adaptive archiving algorithm for storing nondominated vectors. IEEE Transactions on Evolutionary Computation 7(2), 100–116 (2003)
7. Li, H., Zhang, Q.: Multiobjective Optimization Problems With Complicated Pareto Sets, MOEA/D and NSGA-II. IEEE Transactions on Evolutionary Computation 13(2), 284–302 (2009)
8. Okabe, T., Jin, Y., Sendhoff, B.: A critical survey of performance indices for multi-objective optimisation. In: Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003), vol. 2, pp. 878–885 (December 2003)
9. Soylu, B., Köksalan, M.: A favorable weight-based evolutionary algorithm for multiple criteria problems. IEEE Transactions on Evolutionary Computation 14(2), 191–205 (2010)
10. While, L., Bradstreet, L., Barone, L.: A fast way of calculating exact hypervolumes. IEEE Transactions on Evolutionary Computation 16(1), 86–95 (2012)
11. While, R.L., Hingston, P., Barone, L., Huband, S.: A faster algorithm for calculating hypervolume. IEEE Transactions on Evolutionary Computation 10(1), 29–38 (2006)
12. Zhou, A., Zhang, Q., Jin, Y.: Approximating the set of Pareto-optimal solutions in both the decision and objective spaces by an estimation of distribution algorithm. IEEE Transactions on Evolutionary Computation 13(5), 1167–1189 (2009)
13. Zitzler, E.: Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications. Ph.D. thesis, Swiss Federal Institute of Technology Zurich (ETH) (November 1999)
14. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., da Fonseca, V.G.: Performance assessment of multiobjective optimizers: An analysis and review. IEEE Transactions on Evolutionary Computation 7(2), 117–132 (2003)

Brain Storm Optimization Algorithm for Multi-objective Optimization Problems

Jingqian Xue[1], Yali Wu[1], Yuhui Shi[2], and Shi Cheng[2]

[1] School of Automation and Information Engineering,
Xi'an University of Technology, Shaanxi, China
[2] Dept. of Eletrical & Electronic Engineering,
Xi'an Jiaotong-Liverpool University, Suzhou, China
jingqian.xue@hotmail.com, yliwu@xaut.edu.cn,
yuhui.shi@xjtlu.edu.cn, shi.cheng@liverpool.ac.uk

Abstract. In this paper, a novel multi-objective optimization algorithm based on the brainstorming process is proposed(MOBSO). In addition to the operations used in the traditional multi-objective optimization algorithm, a clustering strategy is adopted in the objective space. Two typical mutation operators, Gaussian mutation and Cauchy mutation, are utilized in the generation process independently and their performances are compared. A group of multi-objective problems with different characteristics were tested to validate the effectiveness of the proposed algorithm. Experimental results show that MOBSO is a very promising algorithm for solving multi-objective optimization problems.

Keywords: Brain Strom Algorithm, Multi-objective Optimization, Clustering Strategy, Mutation Operator.

1 Introduction

In recent years, multi-objective optimization problems have gained much attention. the optimum solution for a multi-objective optimization problem is not unique but a set of solutions. The solutions in the set are equally important, that is, no solution is better than any other one with regards to all objectives.

Many kinds of evolutionary computation methods, such as genetic algorithm (GA) [1], evolutionary algorithm (EA) [2], particle swarm optimization (PSO) [3], cultural algorithms (CA) [4], ant colony optimization (ACO) [5], differential evolution (DE) [6], bacterial foraging optimization (BFO) [7], *etc*., have been modified to solve multi-objective problems. Researches indicated that most of these algorithms can improve the convergence and distribution of the Pareto-front more or less.

In swarm intelligence algorithm, the individuals, such as birds in PSO, ants in ACO, bacteria in BFO, *etc.*, moving cooperatively and collectively toward the better and better areas in the solution space, represent only simple objects. Human beings are the most intelligent in the world. Being inspired by this human idea generation process, Shi [8] proposed a novel optimization algorithm - brain storm optimization algorithm

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 513–519, 2012.

(BSO). In this paper, the BSO is further developed to solve multi-objective optimization problems.

The remaining paper is organized as follows. Section 2 describes multi-objective BSO (MOBSO) in detail. Section 3 contains the simulation results and discussion. Section 4 provides the conclusions and some possible paths for future research.

2 Multi-objective Brain Storm Optimization Algorithm

BSO algorithm is designed based on the brainstorming process [9]. In the brainstorming process, the idea generation obeys the Osborn's original four rules. The people in the brainstorming group will need to be open-minded as much as possible and therefore generate more diverse ideas. The procedure of BSO algorithm is shown in [8].

The proposed MOBSO algorithm contains six parts, three of which are specific to the BSO. They are clustering strategy, generation process, and updating global archive. The other three are common to other evolutionary (or swarm intelligence) algorithms.

2.1 Clustering Strategy

Using clustering strategy in the objective space is one of the main novelties of the approach. we use the k-means cluster algorithm [10] to cluster the population into k clusters based on each objective. The process is shown in Algorithm 1. The **Archive_set** contains the non-dominated solutions**,** the **Elite_set** and the **Normal_set** are two temporary sets which are obtained by the clustering in each iteration.

Algorithm 1. Progress of clustering strategy

1. Initialize **Elite_set** = ϕ, **Normal_set** = ϕ;
2. Evaluate the population and update the **Archive_set** according to the Pareto dominance;
3. Initialize the cluster centers based on the fitness value of M objective;
4. For each objective f_m: cluster the population into k clusters, and choose one cluster with the best fitness value as the Elite_cluster$_m$;
5. For each individual: if the individual is in any of Elite_cluster$_m$, then add the individual into the **Elite_set**;else, add the individual into the **Normal_set**.

2.2 Generation Process

After the clustering step, the new individuals will be generated according to the choosing process, dispersal step, mutation operator and selection operator. The procedure is shown in Algorithm 2.

Algorithm 2. Generation procedure

if rand() < *P1*

if rand() < *P2*

if rand() < *P3*

randomly choose an individual from the **Elite_set** as the $\mathbf{X}_{selected}$;

else, randomly choose the $\mathbf{X}_{selected}$ from the **Normal_set**;

end if

else, choose an individual as the $\mathbf{X}_{selected}$ from the **Archive_set**, randomly;

end if

else, dispersal step: randomly generate an individual as the $\mathbf{X}_{selected}$;

end if

where p is the number of individuals, *P1*, *P2*, *P3* are the pre-determined probability values; $\mathbf{X}_{selected}$ is the individual selected to generate the new individual and $\mathbf{X}_{new}$ is the new individual generated by the $\mathbf{X}_{selected}$.

In the dispersal step, a randomly generated individual is chosen as the $\mathbf{X}_{selected}$, the Mutation operator: get the $\mathbf{X}_{new}$ by applying the mutation operator to the $\mathbf{X}_{selected}$; Selection operator: select the Pareto optimal one from $(\mathbf{X}_{selected}, \mathbf{X}_{new})$ as the new individual in next generation.

2.3 Mutation Operator

Mutation generates new solutions from the current ones. Gaussian mutation is usually adopted in the classical evolutionary algorithms. The step of the generation can be represented as follows:

$$x_{new}^{d} = x_{selected}^{d} + \xi * \mathrm{N}(\mu, \sigma) \tag{1}$$

$$\xi = \mathrm{logsig}((0.5 * max\_iternation - current\_iteration)/K) * \mathrm{rand}()$$

where $x_{selected}^{d}$ is the d-th dimension of the individual selected to generate new individual; x_{new}^{d} is the d-th dimension of the individual newly generated; $\mathrm{N}(\mu, \sigma)$ is the Gaussian random function with mean μ and σ ; ξ is a coefficient that weight the contribution of the Gaussian mutation; logsig() is a logarithmic sigmoid transfer function, *max_iteration* and *current_iteration* are the maximum iteration and the current

iteration number, *K* is for changing logsig() function's slope, and rand() is a random value within (0,1).

Another important mutation operator is Cauchy mutation. Cauchy mutation has a higher probability of making longer jumps than Gaussian mutation due to its long flat tails [11]. The results in [11] show that Cauchy mutation is an efficient search operator for a large class of multimodal function optimization problems. In this paper, the Cauchy mutation operator is also utilized to generate new individual as follow:

$$x_{new}^{d} = x_{selected}^{d} + \xi * \mathrm{C}(\mu^{'}, \sigma^{'}) \tag{2}$$

where $\mathrm{C}(\mu^{'}, \sigma^{'})$ is the Cauchy random function with mean $\mu^{'}$ and variance $\sigma^{'}$; x_{new}^{d}, $x_{selected}^{d}$ and ξ are defined as in 2.1. In this paper, both mutations will be employed and compared.

2.4 Selection Operator

Selection operator is used to decide whether any newly generated solution should survive to the next generation. The selection is based on Pareto dominance. For the selected individual $\mathbf{X}_{selected}$ and the mutated individual $\mathbf{X}_{new}$, the selection rules are as follows: if $\mathbf{X}_{new}$ dominates $\mathbf{X}_{selected}$, then $\mathbf{X}_{new}$ survive; if $\mathbf{X}_{selected}$ dominated $\mathbf{X}_{new}$, then $\mathbf{X}_{selected}$ survive; if $\mathbf{X}_{new}$ and $\mathbf{X}_{selected}$ are not dominated by each other, then randomly choose one from $\mathbf{X}_{new}$ and $\mathbf{X}_{selected}$ as the new individual.

2.5 Global Archive

In each generation, the Global Archive is updated by the new non-dominated solutions. The best solution is preserved into the Global Archive the size of which is limited within the size *Max_Archive*. To update the Global Archive, each new non-dominated solution obtained in current iteration will be compared with all members in the Archive. The crowded-comparison operator [12] is adopted to guide the points toward a uniformly spread-out Pareto-optimal front.

3 Experiments and Discussions

3.1 Test Problems and Parameter Setting

In order to evaluate the performance of MOBSO, five benchmark test problems are used [13]: ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6. Each problem has two objectives and no constraints. The parameter settings for all of the problems are listed in Table 1 below. All of the algorithms are implemented in MATLAB using a real-number representation for decision variables. For each experiment, 30 independent runs were conducted to collect the statistical results.

In this paper, we use the metric Υ and metric Δ [13] to measure the performance of the MOBSO algorithm. The metric Υ measures the closeness of solutions to the true Pareto front.

$$\Upsilon = \frac{\sum_{i=1}^{|\mathbf{P}|} d(\mathbf{P}_i, \mathbf{TF})}{|\mathbf{P}|} \tag{3}$$

where 500 uniformly spaced solutions from the true Pareto-optimal front are selected to form the true Pareto front set **TF**; **P** is the Pareto front that has been found; d is the minimum Euclidean distance of $\mathbf{P}_i$ and to all of the points in the set **TF**; and |**P**| is the size of the set **P** .The metric Δ measures the nonuniformity in the distribution.

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{|\mathbf{P}|-1} \left| d_i - \overline{d} \right|}{d_f + d_l + (|\mathbf{P}| - 1)\overline{d}} \tag{4}$$

where, the Euclidean distances d_f and d_l are calculated the same as in [13]; d_i is the Euclidean distance between the i-th pair of consecutive solutions in the set **P**; $\overline{d}$ is the average of all distances d_i .

Table 1. Parametric setup of MOBSO

d	P	P1	P2	P3	k	K	μ	σ	μ'	σ'	Max_Archive	Max_iteration
5(10)	200	0.8	0.6	0.9	5	40	0	1	0	1	100	1000(2000)

3.2 Discussion of the Results

For all the test problems, the metric Υ and metric Δ have been calculated and recorded.

Table 2. The best value and mean value of the convergence metric Υ for Run 30

Function		ZDT1		ZDT2		ZDT3		ZDT4		ZDT6	
d	value	G	C	G	C	G	C	G	C	G	C
5	best	0.0041	**0.0037**	0.0027	0.0030	0.0022	**0.0019**	0.0022	**0.0015**	0.0054	**0.0050**
	mean	0.0132	0.0138	0.0117	**0.0075**	0.0099	**0.0051**	0.2318	**0.1118**	0.0067	0.0079
10	best	0.0166	**0.0134**	0.0196	**0.0128**	0.0116	**0.0064**	0.7075	**0.2490**	0.0157	**0.0111**
	mean	0.0295	**0.0184**	0.0403	**0.0193**	0.0175	**0.0089**	1.9838	**1.0964**	0.0212	**0.0131**

Table 3. The best value and mean value of the diversity metric Δ for Run 30

Function		ZDT1		ZDT2		ZDT3		ZDT4		ZDT6	
d	value	G	C	G	C	G	C	G	C	G	C
5	best	0.6967	**0.6573**	0.6868	**0.6277**	0.7462	0.9254	0.7771	0.7820	0.8509	**0.8019**
	mean	0.8391	**0.8143**	0.8302	**0.7634**	0.8651	0.9511	0.9190	**0.8851**	0.8919	**0.8612**
10	best	0.7057	**0.7029**	0.7372	**0.7124**	0.7666	0.7954	0.8534	0.9354	0.8981	**0.8529**
	mean	0.8010	**0.7790**	0.8159	**0.7921**	0.8404	0.8491	0.9264	0.9668	0.9247	**0.9118**

(In Table 2 and Table 3, d means the dimension, G and C means the MOBSO with Gaussian and Cauchy mutation, respectively)

Table 2 shows the best and mean values of the convergence metric ϒ obtained using MOBSO-G (MOBSO with Gaussian mutation) and MOBSO-C (MOBSO with Cauchy mutation). The diversity metric Δ about the test problems are listed in Table 3.

The results given in Table 2 indicate that the convergence of the MOBSO-C is slightly better than the MOBSO-G in test functions with low dimensional. However, for the function with high dimensional solution space, the MOBSO-C always overwhelms the MOBSO-G on all test functions. While the Table 3 shows that the MOBSO-C has a better diversity on the test functions ZDT1, ZDT2 and ZDT6. The data in Table 3 also indicates that the convergence of the MOBSO declined when the dimension of the solution space extended from five to ten. Because test function ZDT4 has totally 21^9 local segments [13], MOBSO has a little difficulty in converging toward the true Pareto front for the function with high dimensional solution spaces.

Combining the results in Table 2-3, we can conclude that MOBSO can be a promising algorithm for solving multi-objective optimization problems.

4 Conclusions

In this paper, we developed a novel multi-objective optimization algorithm based on the brainstorming process. The using of the clustering strategy guides individuals to move toward the better and better areas. The two different mutation operators have been utilized to generate new individuals and two performance metrics have been used to compare the MOBSOs with the above two different mutations. Simulation results illustrated that both MOBSO-G and MOBSO-C can be a good optimizer for solving multi-objective optimization problems. Adaptive and mixing mutations based on niching techniques should also be investigated. Another interesting area is to exploit the MOBSO for solving multi-objective optimization problems with constraints and for solving many-objective problems.

Acknowledgement. This paper is partially supported by partially supported by Natural Science Foundation of Shaanxi Province under Grant Number 2010JQ8006 and Science Research Programs of Education Department of Shaanxi Province under Grant

Number 2010JK711, and partially supported by National Natural Science Foundation of China under Grant Number 60975080 and Suzhou Science and Technology Project under Grant Number SYJG0919.

References

1. Horn, J., Nafpliotis, N., Goldberg, D.E.: A Niched Pareto Genetic Algorithm for Multiobjective Optimization. In: IEEE World Congress on Computational Intelligence, Evolutionary Computation, pp. 82–87 (1994)
2. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the Strength Pareto Evolutionary Algorithm. In: Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, Athens, Greece, pp. 95–100 (2001)
3. Kennedy, J., Eberhart, R., Shi, Y.: Swarm Intelligence. Morgan Kaufmann Publisher (2001)
4. Reynolds, R., Liu, D.: Multi-Objective Cultural Algorithms. In: Proceedings of 2011 Congress on Evolutionary Computation (CEC 2011), pp. 1233–1241 (2011)
5. Dorigo, M., Stützle, T.: Ant Colony Optimization. The MIT Press (2004)
6. Becerra, R.L., Coello Coello, C.A.: Solving Hard Multiobjective Optimization Problems Using ε-Constraint with Cultured Differential Evolution (2006)
7. Passion, K.M.: Bacterial Foraging Optimization. International Journal of Swarm Intelligence Research 1(1), 1–16 (2010)
8. Shi, Y.: Brain Storm Optimization Algorithm. In: Tan, Y., Shi, Y., Chai, Y., Wang, G. (eds.) ICSI 2011, Part I. LNCS, vol. 6728, pp. 303–309. Springer, Heidelberg (2011)
9. Smith, R.: The 7 Levels of Change, 2nd edn. Tapeslry Press (2002)
10. Jain, A.K.: Data clustering: 50 years beyond K-means. Pattern Recognition Letters (31), 651–666 (2010)
11. Yao, X., Liu, Y., Lin, G.: Evolutionary Programming Made Faster. IEEE Transactions on Evolutionary Computation 3(2), 82–102 (1999)
12. Zitzler, E., Deb, K.A., Thiele, L.: Comparison of multiobjective evolutionary algorithms: Empirical results. Evolutionary Computation 8(2), 173–195 (2000)
13. Deb, K.A., Pratap, A., et al.: A Fast and Elitist Multiobjective Genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2), 182–197 (2002)

Modified Multi-Objective Particle Swarm Optimization Algorithm for Multi-objective Optimization Problems

Ying Qiao

Research Institute of Information and System Science, Beifang University of Nationalities, YinChuan, 750021, China
qiaodoctor@126.com

Abstract. Multi-objective particle swarm optimization (MOPSO) is an optimization technique inspired by bird flocking, which has been steadily gaining attention from the research community because of its high convergence speed. However, faced with multi-objective problems, adaptations are needed. Deeper researches must be conducted on its key steps, such as guide selection, in order to improve its efficiency in this context. This paper proposes an modified multi-objective particle swarm optimizer named MMOPSO, for dealing with multi-objective problems. we introduce some ideas concerning the guide selection for each particle. The proposed algorithm is compared against four multi-objective evolutionary approaches based on particle swarm optimization on four benchmark problems. The numerical results show the effectiveness of the proposed MMOPSO algorithm.

Keywords: multi-objective optimization, particle swarm optimization, modified operator, guide selection.

1 Introduction

Particle Swarm Optimization (PSO) is a heuristic search technique that simulates the movements of a flock of birds which aim to find food, which was proposed by Kennedy and Eberhart in 1995[1,2]. The relative simplicity of PSO and the fact that is a population-based technique have made it a natural candidate to be extended for multi-objective optimization. Such as Coello Coello CA, Pulido GT and Lechuga MS proposed MOPSO, the algorithm introduced external populations of adaptive network system, which require variation for particle and particle scope, variation scale is proportional to evolution algebra[3]. Tsai et al. proposed an improved multi-objective particle swarm optimization algorithm[4]. Mostaghim S and Teich proposed Sigma method that decided gbest for each particle and introduced disturbance factor[5]. Ono, S.,Nakayama, S. proposes an algorithm using multi-objective Particle Swarm Optimization (MOPSO) for finding robust solutions against small perturbations of design variables[6].Dun-wei Gong, Jian-hua Zhang present a global path planning approach based on multi-objective particle swarm optimization[7]. Q-K Pan, L Wang

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 520–527, 2012.

and Qian B. presents a novel multi-objective particle swarm optimization (MOPSO) algorithm for solving no-wait flow shop scheduling problems with makespan and maximum tardiness criteria. in the algorithm, particles are represented as job permutations and updated directly in the discrete domain[8]. S.Z.Zhao, P.N.Sunganthan proposed two-lbests based multi-objective particle swarm optimizer[9]. This approach emphasizes the global best (gbest) or local best (lbest) of every particle in state-of-the-art multi-objective particle swarm optimization (MOPSO) implementations is selected from the non-dominated solutions in the external archive.

2 Basic Concept of MOP

The multi-objective optimization problem can be mathematically described as:

$$\min_{\vec{x}\in R^D} \quad \vec{f}(\vec{x}) \tag{1}$$

where $\vec{x} = (x_1, x_2, \ldots, x_D)$ is the D decision variables, and $\vec{f}(\vec{x})=(f_1, f_2, \cdots, f_M)$ are the M objectives to be minimized.

There are two basic concepts which are often used in multi-objective optimization:

Definition 1(Pareto dominate): A vector $\vec{u} = (u_1, u_2, \ldots, u_D)$ is said to dominate $\vec{v} = (v_1, v_2, \ldots, v_D)$ (denoted by $\vec{u} \succ \vec{v}$) if and only if

$$(\forall i \in \{1, \cdots, D\}, u_i \le v_i) \wedge (\exists i_0 \in \{1, \cdots, D\}, u_{i_0} \le v_{i_0})$$

Definition 2(Pareto optimal solution): A point $\vec{x^*} \in R^D$ is Pareto optimal if there is not another $\vec{x} \in R^D$ satisfies with $\vec{f}(\vec{x}) \succ \vec{f}(\vec{x^*})$.

3 Standard Particle Swarm Algorithm

Particle swarm algorithm optimization is a stochastic, population-based and global evolutionary algorithm proposed by Kennedy and Eberhart in 1995. With the standard particle swarm optimization, each particle of the swarm adjusts its trajectory according to its own flying experience and the flying experiences of other particles within its topological neighborhood in a D-dimensional space S. The velocity and position of particle i are represented as $\vec{v} = (v_{i1}, v_{i2}, \cdots, v_{iD})$ and $\vec{x_i} = (x_{i1}, x_{i2}, \cdots, x_{iD})$ respectively. Its best historical position is recorded as $\vec{p_i} = (p_{i1}, p_{i2}, \cdots, p_{iD})$, which is also called p_{best}. The best historical position that

the entire swarm has passed is denoted as g_{best}. The velocity and position of particle i on dimension j(j= 1,2,. . . ,D) in iteration t + 1 are updated as follows:

$$v_{ij}^{t+1} = wv_{ij}^{t} + c_1 r_1 (pbest_{ij}^{t} - x_{ij}^{t}) + c_2 r_2 (gbest_{ij}^{t} - x_{ij}^{t}) \tag{2}$$

$$x_{ij}^{t+1} = x_{ij}^{t} + v_{ij}^{t+1} \tag{3}$$

Where c_1 and c_2 denote constant, which is called the acceleration coefficients, r_1 and r_2 are elements from two uniform random sequences in the range of [0,1], w is the inertia weight which decent by linear decrease.

4 Modified Multi-Objective Particle Swarm Optimization Algorithm

4.1 Modified Operator

According to the searching behavior of PSO, the gbest will be an important clue in leading particles to the global optimal solution, but it is unavoidable that the solution would fall into the local minimum while particles are trying to find better solutions. In order to allow the solution exploration in the area to produce more potential solutions and to explore un-searched solution space, we introduces a modified operation, modified operator is given as below:

$$\begin{aligned}
&If \quad r < 0.5 \\
&\quad v_{ij}^{t+1} = wv_{ij}^{t} + c_1 r_1 (pbest_{ij}^{t} - x_{ij}^{t}) + c_2 r_2 (gbest_{ij}^{t} - x_{ij}^{t}) \\
&\quad x_{ij}^{t+1} = x_{ij}^{t} + v_{ij}^{t+1} \\
&Else \\
&\quad x_{ij}^{t+1} = p \times opti^{t}(rr, j) + (1-p) \times (opti^{t}(rr, j) - x^{t}(i, j)) \\
&End
\end{aligned} \tag{4}$$

where the random number *r* is a uniform distribution form zero to one, the same of *p* , $opti^{t}(rr)$ is the member chosen randomly from the external repository.

4.2 Guide Selection

In single-objective problems there is only one existent gbest. In multi-objective problems, more than one conflicting objectives must all need be optimized. The number of non-dominated solutions which are located on/near the Pareto front will be

more than one. Therefore, each non-dominated solution can be the gbest and provides its position information to current particles. According to particle searching behavior in multi-objective problems, the pbest of a particle will usually be its current position.

Therefore, the pbest is useless for guiding particles toward to find new solution in most situations, the same as gbest.we will propose a method to solve the useless guiding problem. and is described as follow:

$$\begin{aligned} &If \quad r < 0.5 \\ &\quad pbest_{ij}^{t+1} = x_{ij}^{t} \\ &Else \\ &\quad pbest_{ij}^{t+1} = x^{t}(r_1, j) + rand \times (x^{t}(r_2, j) - x^{t}(r_3, j)) \\ &End \end{aligned} \tag{5}$$

$$\begin{aligned} &If \ r < 0.5 \\ &\quad gbest_{ij}^{t+1} = x_{ij}^{t} \\ &Else \\ &\quad gbest_{ij}^{t+1} = opti^{t}(r_4, j) + rand \times (opti^{t}(r_5, j) - opti^{t}(r_6, j)) \\ &End \end{aligned} \tag{6}$$

where the random number r is a uniform distribution form zero to one, the same of *rand* , $x^{t}(r_1), x^{t}(r_2), x^{t}(r_3)$ are the members chosen randomly from evolutionary population. $opti^{t}(r_4)$, $opti^{t}(r_5)$, $opti^{t}(r_6)$ are the members chosen randomly from the external repository.

4.3 The Pseudo Code of MMOPSO

Begin

Initial particle's velocity, position, global best particle(gbest), past best particle(pbest), external repository.

While stopping criterion is not met

For each Particle

1. Update particle's position and according to (4).
2. Update the external repository using dominate.
3. Assign pbest to each particle using the members in the evolutionary population according to (5).
4. Assign gbest to each particle using the members in the external repository according to (6).

End For

End While

End

5 Experimental Results

5.1 Tests Problems

Different sets of classical test problems suggested in the MOEA literature are used to estimate the performance of the MMPSO. In this paper, we choose four tests problems: ZDT1, ZDT2, ZDT3 and ZDT6, which is defined in [10].

5.2 Performance Measures

To validate our approach, we used the methodology normally adopted in the evolutionary multi-objective optimization literature. we use two metrics that have been used in these studies. They represent both convergence metric and diversity metric.

(i) Convergence metric(γ)

This metric is defined as [10]:

$$\gamma = \frac{\sum_{i=1}^{n} d_i}{n} \tag{7}$$

where n is the number of non-dominated vector found by the algorithm being analyzed and d_i is the Euclidean distance between the obtained non-dominated front Q and the nearest member in the true Pareto front P. It measures the distance between the Q and P.

(ii) Diversity metric (Δ)

It measures the extent of spread achieved among the obtained solutions and is defined as:

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{s-1} | d_i - \bar{d} |}{d_f + d_l + (s-1)\bar{d}} \tag{8}$$

where $\bar{d} = \frac{\sum_{i=1}^{s-1} d_i}{s-1}$,the parameter d_f and d_i are the Euclidean distances between the extreme solutions and the boundary solutions of the obtained non-dominated set. The parameter $\bar{d}$ is the average of all distances d_i, $i = 1, 2, \ldots, (s-1)$, assuming that there are s solutions on the best non-dominated front.

5.3 Comparison and Discussion

In order to verify MMOPSO performance of the algorithms, which numerical experiments are compared with MOPSO[3], OMOPSO[11], EMMOPSO[12], SMOPSO[13], NSPSO[14]. To match the settings of the algorithms used for comparison, the population size is set 100 and the algorithm is run for 200 generations, the maximum size of external elitist archive is set 100. Results on four test functions, in relation to the convergence metric and diversity metric, are presented in Table I-IV, the mean and variance of the values are averaged over 30 runs. As we can see, MMOPSO is able to make better both on the convergence metric and diversity metric in all problems.

Table 1. Statistics of results on ZDT1

Algorithm	Convergence ± Variance	Diversity ± Variance
MOPSO	0.098 ± 6.17e-04	0.66 ± 7.23e-03
OMOPSO	0.069 ± 4.12e-05	0.59 ± 6.36e-03
EMMOPSO	0.005 ± 6.82e-07	0.39 ± 1.57e-04
SMOPSO	0.089 ± 5.61e-04	0.64 ± 3.31e-04
NSPSO	0.139 ± 4.17e-03	0.68 ± 6.23e-03
MMOPSO	0.001 ± 1.48e-08	0.22 ± 2.24e-04

Table 2. Statistics of results on ZDT2

Algorithm	Convergence ± Variance	Diversity ± Variance
MOPSO	0.273 ± 5.77e-02	0.87 ± 4.75e-02
OMOPSO	0.007 ± 4.63e-04	0.59 ± 5.63e-03
EMMOPSO	0.005 ± 6.83e-08	0.27 ± 5.77e-06
SMOPSO	0.076 ± 5.94e-04	0.59 ± 4.64e-03
NSPSO	0.099 ± 5.21e-03	0.63 ± 6.18e-03
MMOPSO	0.001 ± 7.46e-08	0.21 ± 7.88e-04

Table 3. Statistics of results on ZDT3

Algorithm	Convergence ± Variance	Diversity ± Variance
MOPSO	0.189 ± 3.51e-03	0.51 ± 1.16e-03
OMOPSO	0.159 ± 1.45e-03	0.59 ± 2.56e-03
EMMOPSO	0.0073 ± 6.80e-06	0.57 ± 1.29e-03
SMOPSO	0.174 ± 1.59e-03	0.54 ± 5.12e-03
NSPSO	0.231 ± 1.97e-02	0.51 ± 8.65e-03
MMOPSO	0.005 ± 4.39e-08	0.45 ± 1.78e-04

Table 4. Statistics of results on ZDT6

Algorithm	Convergence ± Variance	Diversity ± Variance
MOPSO	0.447 ± 5.28e-02	0.87 ± 6.32e-02
OMOPSO	0.010 ± 5.76e-06	0.69 ± 7.25e-05
EMMOPSO	0.006 ± 4.66e-07	0.64 ± 7.52e-05
SMOPSO	0.115 ± 6.37e-02	0.65 ± 5.78e-02
NSPSO	0.215 ± 4.37e-02	0.68 ± 5.48e-02
MMOPSO	0.002 ± 6.86e-08	0.202 ± 1.61e-04

6 Conclusion

In order to enhance wide-ranged exploration ability and explore un-searched space ability of particle swarm optimization ,this paper will introduce modified operator and guide selection into the particle swarm optimization. Experimental results show that MMOPSO is an effective multi-objective particle swarm optimization.

Acknowledgment. The work is supported by The National Natural Science Foundation of China (60962006) and the Foundation of Beifang University of Nationalities (2010Y036).

References

1. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: IEEE International Conference on Neural Networks, pp. 1942–1948 (1995)
2. Shi, Y., Eberhart, R.: A modified particle swarm optimizer. In: Proceeding of the IEEE World Congress on Computational Intelligence, pp. 69–73 (1998)
3. Coello, C.C., Pulido, G.T., Lechuga, M.S.: Handling multiple objectives with particle swarm optimization. IEEE Trans. on Evolutionary Computations 8(3), 256–279 (2004)
4. Shang, J.T., Sun, T.Y., Liu, C.C.: An improved multi-objective particle swarm optimizer for multi-objective problems. Expert Systems with Applications 2(18), 1–15 (2010)
5. Mostaghim, S., Teich: Strategies for finding good local guides in Multi-Objective Particle Swarm Optimization (MOPSO). In: Proceedings 2003 IEEE Swarm Intelligence Symp., Indianapolis, pp. 26–33 (2003)
6. Ono, S., Nakayama, S.: Multi-Objective Particle Swarm Optimization for robust optimization and its hybridization with gradient search. In: IEEE International Conference on Evolutionary Computations, pp. 1629–1636 (2009)
7. Gong, D.W., Zhang, J.H., Zhang, Y.: Multi-objective Particle Swarm Optimization for Robot Path Planning in Environment with Danger Sources. Journal of Computers 6(8), 1554–1561 (2011)
8. Pan, Q.K., Wang, L., Qian, B.: A novel multi-objective particle swarm optimization algorithm for no-wait flow shop scheduling problems. Journal of Engineering Manufacture 222(4), 519–539 (2008)

9. Zhao, S.Z., Sunganthan, P.N.: Two-lbests based multi-objective particle swarm optimizer. Engineering Optimization 43(1), 1–17 (2011)
10. Deb, K., Pratap, A.: A fast elitist non-dominated sorting genetic algorithm for multi-objective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2), 182–197 (2002)
11. Sierra, M.R., Coello, C.A.C.: Improving PSO-Based Multi-objective Optimization Using Crowding, Mutation and ε-Dominance. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) EMO 2005. LNCS, vol. 3410, pp. 505–519. Springer, Heidelberg (2005)
12. Reddy, M.J., Kumar, D.N.: An efficient multi-objective optimization algorithm based on swarm intelligence for engineering design. Engineering Optimization 39(1), 49–68 (2007)
13. Mostaghim, M.S., Teich, J.: Strategies for finding good local guides in multi-objective particle swarm optimization. In: IEEE Swarm Intelligence Symposium, Indianapolis, pp. 26–33 (2003)
14. Li, X.: A Non-Dominated Sorting Particle Swarm Optimizer for Multi-Objective Optimization. In: Cantú-Paz, E., Foster, J.A., Deb, K., Davis, L., Roy, R., O'Reilly, U.-M., Beyer, H.-G., Kendall, G., Wilson, S.W., Harman, M., Wegener, J., Dasgupta, D., Potter, M.A., Schultz, A., Dowsland, K.A., Jonoska, N., Miller, J., Standish, R.K. (eds.) GECCO 2003. LNCS, vol. 2723, pp. 37–48. Springer, Heidelberg (2003)

A Multi-objective Mapping Strategy for Application Specific Emesh Network-on-Chip (NoC)

Bixia Zhang[1,2], Huaxi Gu[1,2], Sulei Tian[3], and Bin Li[3]

[1] State Key Laboratory of ISN, Xidian University, 710071 Xi'an, China
[2] Science and Technology on Information Transmission and Dissemination in Communication Networks Laboratory
[3] The 54th Institute of CETC, 050081 Shijiazhuang, China
zhangbixiaok@163.com

Abstract. This paper proposes a new optimization model for mapping IP cores onto a new Network-on-Chip (NoC) topology, Emesh. Since competition for one port of the router in Emesh is violent, network competition is considered in our model, in addition to energy and communication cost. To solve this optimization model, the authors present a new application specific multi-objective mapping algorithm onto Emesh topology based on the idea of the crossover and mutation of genetic algorithm. The experimental results show that, compared to traditional heuristic genetic algorithm, the proposed algorithm has lower energy consumption, lower communication cost and less network competition.

Keywords: Network-on-Chip (NoC), Emesh, mapping, energy consumption, network competition, communication cost.

1 Introduction

With the advances in integrated circuits and semiconductor technology, hundreds of Intellectual Property (IP) cores are allowed to be put on a single chip. These IP cores can be general-purpose processors, coprocessors, DSPs, application specific hardware, memory blocks, I/O blocks, etc. The use of standard hardwired buses to interconnect these cores is not scalable. To solve this problem, Network-on-Chip (NoC) has been proposed and used for interconnecting the cores [1] [2]. The use of on-chip interconnection network has several advantages, including better structure, performance and modularity. Therefore, the network on chip is going to be a promising architecture for future complex SoCs.

The 2D-mesh topology has become the most commonly used choice in NoCs owing to its regularity. However, the diameter of network will be increased rapidly with the network size. Therefore, Saneei etc. introduce Cluster-Mesh (CM) into NoC that reduces the power consumption and the area overhead. However, this structure is easy to be blocked because a router in the CM topology connects 4 IP cores at most. Therefore, in this paper, we employ a better structure which is called Emesh.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 528–536, 2012.

In NoC design, mapping a specific application to the NoC platform is an important step, which significantly affects the system power consumption, latency, and etc. NoC mapping is an NP-complete problem. It is hard to obtain the optimal solution by exhausting search due to the complexity of time and space. There are many important metrics that impact network performance, like power consumption, delay, and throughput etc. Moreover, some of these metrics are more important for certain applications than others and it is easier to consider single metric. Therefore, researchers tried to optimize the NoC architecture for the most important metric, assuming other metrics as constraints [3] [4] [5]. However, NoC-based systems usually have conflicting metrics. Therefore, the challenging problem is to optimize the on-chip networks for more than one metric. In this paper, at most 4 IP cores may be connected to a router. Therefore, it will cause more serious competition in the network, in other words, we need to consider the problem of network competition. Moreover, in order to reduce system overhead, the traffic transmitted by the routers should be as little as possible, that is to say, the communication cost needs to be optimum. As a result, we target a three-objective optimization problem, including energy consumption, communication cost and network competition.

This paper is organized as follows. The multi-objective optimization problem is introduced in Section 2. Section 3 presents our proposed mapping algorithm. Section 4 presents experimental results for two applications. Finally, we draw our conclusions and give ideas for future work in Section 5.

2 Related Work

There are various mapping algorithms which have been proposed to map an application onto different topologies. Different topologies have different advantages and problems. Regarding the application specific mapping optimization, the mapping and routing allocation for the tile based architectures have been addressed in [6]. In [7], a branch and bound algorithm has been proposed that maps cores onto a tile based NoC architecture satisfying the bandwidth constraints and minimizing the total energy consumption. Lei and Kumar [8] present an approach that uses genetic algorithms to map an application onto a mesh-based NoC architecture. The algorithm finds a mapping of the vertices of the task graph on the available cores so as to minimize the execution time. De Micheli addresses the problem under the bandwidth constraint with the aim of minimizing communication delay by exploiting the possibility of splitting traffic among various paths in [9]. In [10], a multi-objective genetic algorithm (GAMAP) has been proposed that solves the problem of topological mapping of IPs/cores in a mesh-based NoC architecture. This approach explores the mapping space and finds the Pareto mappings that optimize performance and power consumption. And the experiments confirm the efficiency, accuracy and scalability of the approach. In [11], a multi-objective genetic algorithm (MGAP) has been proposed that solves the mapping problem in two systematic steps. These two steps are task assignment and core-tile mapping respectively. According to simulation results, the proposed algorithm saves more than 15% link bandwidth and 15%-20% (on average) of energy

consumption as compared to PMAP and PBB. In order to minimize energy and maximum bandwidth requirements, Rabindra etc propose a multi-objective genetic algorithm [12]. This approach considers "many-many" mapping between switch and cores instead of "one-one" mapping. In [13], a new mapping algorithm is proposed based on Artifical Bee Colony model to solve the problem of energy aware mapping optimization in NoC design. The comparison of the proposed algorithm with Genetic Algorithm and Max-Min Ant System based mapping algorithm shows that the new algorithm has lower energy consumption and faster convergence rate.

3 Problem Definition

A Core Graph (CG) is a directed graph, $G(V,E)$, where each vertex $v_i \in V$ represents an IP core, each directed arc $e_{i,j} \in E$ represents communication between the IP core v_i and v_j, and the weight of edge $w_{i,j} \in W$ signifies the volume of data flowing through the edge.

An Architecture Characterization Graph (ACG) is a directed graph, $G'(N,R,P)$, where each vertex $n_i \in N$ represents a tile in the architecture, each node $R_i \in \mathscr{R}$ represents the i-th router in the architecture, r_i^k represents the k-th port of the i-th router, where $k \in \{0,1,2,3\}$ and 0, 1, 2, 3 represent the North, East, South, West port of the router, respectively, and each directed arc $p_{i,j} \in P$ represents the routing path from the tile n_i to n_j.

3.1 Architecture

The network under consideration is composed of $m \times n$ tiles and $(m-1)\times(n-1)$ routers interconnected according to the Emesh topology. Fig. 1 shows an Emesh NoC

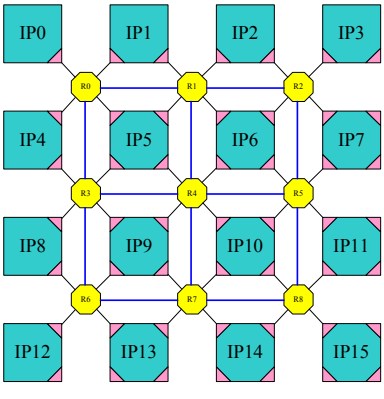


Fig. 1. 4×4 Emesh NoC architecture

architecture which has 16 tiles and 9 routers (nodes). Each IP core can be located in a tile. The router is connected to the four neighboring routers and its local IP core(s) via channels. A 8×8 crossbar switch is used as the switching fabric in the router. In this architecture, an IP core can be connected to a router around it. Therefore, 4 IP cores may be connected to the same router.

3.2 Energy Model

We use the energy consumption model proposed in [14]. The average energy consumption for sending one bit of data from tile n_i to can be calculated as follows

$$E_{bit}^{n_i,n_j} = n_{hops} \times E_{Sbit} + (n_{hops} - 1) \times E_{Lbit} \tag{1}$$

where E_{Sbit} and E_{Lbit} represent the energy consumed by the switch inside the switching fabric and on the links between routers, respectively, and n_{hops} is the number of routers the bit traverses from tile n_i to n_j.

Then, the average energy consumption for total network is

$$E = \sum\nolimits_{\forall e_{i,j}} \left(w_{v_i,v_j} \times E_{bit}^{map(v_i),map(v_j)} \right) \tag{2}$$

where $map(v_i)$ represents the mapping result of IP core v_i.

3.3 Communication Cost Model

In order to improve system performance and reduce system overhead, the traffic transmitted by routers should be as little as possible. Hence, the communication cost between IP core v_i and v_j is given by

$$t_{i,j} = \sum_{R_k \in p_{i,j}} f(R_k) \times w_{i,j} \tag{3}$$

where

$$f(R_k) = \begin{cases} 1 & if(R_k \neq s(p_{i,j})) \\ 0 & else \end{cases} \tag{4}$$

where $s(p_{i,j})$ is the source router of path $p_{i,j}$.

Then, the total communication cost of the network is given by

$$T = \sum_{\forall e_{i,j} \in E} \left(t_{i,j} \right) \tag{5}$$

3.4 Network Competition Model

In Emesh topology, since the number of IP cores a router connected to may be up to 4, the problem of network competition is that 4 IP cores may compete for the same output port. Hence, the competition of the k-th port of the i-th router is given by

$$g(r_i^k) = c(r_i^k) \times e^{c(r_i^k)} \tag{6}$$

where $c(r_i^k)$ represents the number of input ports which compete for current port.

Then, the total competition of the network is given by

$$C = \sum_i \sum_{k=0}^{3} g(r_i^k) \tag{7}$$

3.5 Optimization Model

With the model mentioned above, the optimal mapping problem can be described as

$$\min\{\alpha_0 E' + \alpha_1 T' + \alpha_2 C'\} \tag{8}$$

s.t.

$$\forall v_i \in V, map(v_i) \in N. \tag{9}$$

$$\forall v_i \neq v_j \in V, map(v_i) \neq map(v_j) \in N. \tag{10}$$

where E', T' and C' are the normalized energy, communication cost and network competition costs. The normalization is necessary due to the potential large difference between the absolute values of the three cost components. The parameters α_0, α_1 and α_2 are weight factors. Conditions (9) and (10) mean that each IP core should be mapped onto one tile and no tile can host more than one IP core.

4 Emesh Based Mapping Algorithm

The traditional heuristic approaches are likely to trap into local optimal solutions, we present a new application specific multi-objective mapping algorithm (MOMA) based on Emesh topology, a new structure, for NoC design. This algorithm draws on the idea of the crossover and mutation of genetic algorithm.

Define the population size and the number of IP cores as N_{pop} and N_{core}, respective- ly. Then, the following pseudo code will describe the proposed algorithm in detail:

```
const COUNT; // limiting parameter of inner iterations
        max_iter; // maximum outside iterations
begin
  iterout = 1;
  repeat
    generate a population randomly whose size is 2N_pop;
    choose the better N_pop individuals to make up the
   initial population; //optimize the initial population
    cnt = 1; // the generation of new population
    iterin = 1;
```

```
    repeat
      if iterin is odd
        i = 1;
        repeat
          generate rand01 randomly // 0≤rand01≤(Ncore -1)
          exchange the rand01th IP core of ith and (i+1)th
          individuals // generate 2 new individuals
          i = i + 2;
        until i = Npop /2 -1 // first part of population
        i = Npop /2 + 1;
        repeat
          generate rand02 randomly // 0≤rand02≤(Ncore -1)
          change the rand02th IP core information of the
          ith individual // generate a new individual
          i = i + 1;
        until i = Npop // last part of population
      if iterin is even
        i = 1;
        repeat
          generate rand01 randomly // 0≤rand01≤(Ncore -1)
          change the rand02th IP core information of the
          ith individual // generate a new individual
          i = i + 1;
        until i = Npop /2 // first part of population
        i = Npop /2 + 1;
        repeat
          generate rand02 randomly // 0≤rand01≤(Ncore -1)
          exchange the rand01th IP core of ith and (i+1)th
          individuals // generate 2 new individuals
          i = i + 2;
        until i = Npop -1 // last part of population
      apply roulette wheel selection to select
      individuals // generate the next generation
    iterin = iterin + 1;
      if find a better mapping result
        cnt  = 1;
      else cnt  = cnt + 1;
        until cnt  = COUNT // exit inner iteration
    iterout = iterout + 1;
  until iterout = max_iter
end.
```

5 Experiments and Results

In order to show the performance of the proposed mapping algorithm, we use Genetic algorithm (GA) [7] and Multi-Objective Mapping Algorithm (MOMA) based mapping

to different benchmarks, respectively. The simulations are running with VC++6.0 in windows XP OS.

As is shown in Fig. 2, they are two application specific communication core graphs used in most of simulations in application specific NoC design. Fig. 2(a) is a Video Objective Plane Decoder (VOPD) application core graph will be mapped onto 4×4 Emesh topology [5]. The MPEG-4 decoder application core graph is shown in Fig. 2(b) with 12 IPs [5] which will be mapped onto 3×4 Emesh topology.

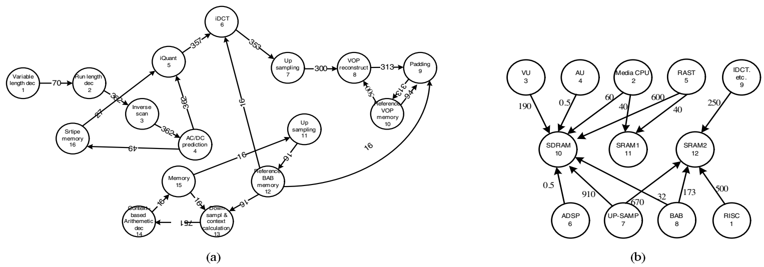
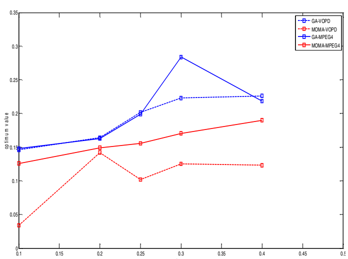


Fig. 2. The benchmark of application specific traffic: (a) VOPD communication core graph (Mb/s); (b) MPEG-4 decoder communication core graph (Mb/s)



(α_0, α_1, α_2)		(0.1,0.1,0.8)			(0.2,0.2,0.6)		
E(mJ),T(Mb/s)		E	T	C	E	T	C
VOPD	GA	8549	2559	104	7519	1957	104
	MOMA	6507	1405	0	7762	1616	104
MPEG-4	GA	6512	1998	104	5439	1344	120
	MOMA	5332	1279	104	5332	1279	104

Fig. 3. The performance of different methods in VOPD and MPEG-4 communication graph

Fig. 3 shows the optimal solution derived by GA and MOMA based mapping respectively. From Fig 3, the multi-objective optimal mapping results of MOMA save 76.7% in VOPD and 39.9% in MPEG-4 than GA based mapping at most. The corresponding energy, communication cost and network competition are also shown in Fig 3. The mapping results are varied as the weight factors. In the case of α_0=0.1, α_1=0.1 and α_2=0.8, respectively, the energy, communication cost and network competition of MOMA based mapping are 6507 (mJ), 1405 (Mb/s) and 0 in VOPD, and 8549 (mJ), 2559 (Mb/s) and 104 of GA based mapping. According to the calculation analysis, MOMA based mapping has lower energy consumption, lower communication cost and less network competition than GA based mapping.

6 Conclusions and Future Work

In this paper, we proposed a new multi-objective mapping algorithm (MOMA) for application specific Emesh NoC design. This algorithm draws on the idea of the crossover and mutation of genetic algorithm. According to the simulation results, MOMA is an effective mapping algorithm. In the future, more work needs to be done in order to improve network performance, considering the constrained problems.

Acknowledgment. This work is supported partly by the National Science Foundation of China under Grant No. 61070046, No. 60803038, the special fund from State Key Lab (No.ISN1104001), the Fundamental Research Funds for the Central Universities under Grant No.K50510010010, the 111 Project under Grant No. B08038, the fund from Science and Technology on Information Transmission and Dissemination in Communication Networks Laboratory under Grant No. ITDU-11009.

References

1. Benini, L., Micheli, G.D.: Networks on Chips: A New SoC Paradigm. IEEE Computer Society 35(1), 70–78 (2002)
2. Kumar, S., Foster, I., Kesselman, C.: A network on chip architecture and design methodology. In: IEEE Computer Society Annual Symposium on VLSI (ISVLSI), pp. 117–124 (2002)
3. Srinivasan, K., Chatha, K., Konjevod, G.: Linear-programming-based techniques for synthesis of network-on-chip architectures. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 14(4), 407–420 (2006)
4. Leary, G., Srinivasan, K., Mehta, K., Chatha, K.S.: Design of network-on-chip architectures with a genetic algorithm-based technique. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 17(5), 674–687 (2009)
5. Dumitriu, V., Khan, G.: Throughput-oriented NoC topology generation and analysis for high performance SoCs. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 17(10), 1433–1446 (2009)
6. Umit, Y.O., Hu, J.: Key research problems in NoC design: A Holistic Perspective. In: International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS), pp. 69–74 (2005)
7. Tang, L., Peng, H., Yang, Y.: A two-step genetic algorithm for mapping task graphs to a network on chip architecture. In: Euromicro Symposium on Digital System Design, pp. 180–187 (2003)
8. Hu, J., Marcuiescu, R.: Energy-Aware Mapping for Tile-based NoC Architectures under Performance Constraints. In: Asia and South Pacific Design Automation (ASP-DAC), pp. 233–239 (2003)
9. Murali, S., Micheli, G.D.: Bandwidth-constrained mapping of cores onto NoC architectures. In: Design, Automation and Test in Europe Conference and Exhibition (DATE), vol. 2, pp. 896–901 (2004)
10. Giuseppe, A., Vincenzo, C., Maurizio, P.: A Multi-Objective Genetic Approach to Mapping Problem on Network-on-Chip. Journal of Universal Computer Science 12(4), 370–394 (2006)

11. Rabindra, K.J., Gopal, K.S.: A Multi-Objective Evolutionary Algorithm Based Optimization Model for Network-on-Chip Synthesis. In: International Conference on Information Technology New Generation (ITNG), pp. 977–982 (2007)
12. Rabindra, K.J., Prabhat, K.M.: Design Space Exploration of Network-on-Chip. International Journal of Computing and ICT Research 2(1), 17–25 (2008)
13. Deng, Z., Gu, H., Feng, H., Shu, B.: Artificial Bee Colony Based Mapping for Application Specific Network-on-Chip Design. In: Tan, Y., Shi, Y., Chai, Y., Wang, G. (eds.) ICSI 2011, Part I. LNCS, vol. 6728, pp. 285–292. Springer, Heidelberg (2011)
14. Hu, J., Marculescu, R.: Exploiting the routing flexibility for energy/performance aware mapping of regular NoC architectures. In: Design, Automation and Test in Europe Conference and Exhibition (DATE), pp. 688–693 (2003)

Binary Nearest Neighbor Classification of Predicting Pareto Dominance in Multi-objective Optimization

Guanqi Guo, Cheng Yin, Taishan Yan, and Wenbin Li

Hunan Institute of Science and Technology, Yueyang, Hunan, 414006, China
{gq.guo,yincheng886337,yantaishan163,wenbin_lii}@163.com

Abstract. A method of predicting Pareto dominance in multi-objective optimization using binary nearest neighbor classification (BNNC) is proposed. It encodes real value feature variables into binary bit strings with the same length. The similarity of two feature variables is directly measured by weighted sum of the binary bits. The analysis shows that when the orders of magnitude for various feature variables differ from each other, the similarity measured by scaled feature variables is able to more uniformly reflect the contribution of each feature variable to Pareto dominance relationship, and BNNC has computational complexity of $O(N)$. Experiments results show that, in addition to remarkably increasing classification accuracy rate, it is more efficient and robust than the canonical nearest neighbor rule and Bayesian classification when used to classify those problems with unbalanced class proportions and feature vectors no less than 2 dimensions.

Keywords: multi-objective optimization, Pareto dominance, pattern classification, binary nearest neighbor classification.

1 Introduction

In recent years, there is an increasing interest in applying evolutionary algorithms to solve multi-objective optimization problems [1], [2]. A number of well-known multi-objective optimization evolutionary algorithms (MOEAs) have been proposed, such as NSGA-II [3], SPEA-II [4], PAES [5], MOPSO [6]. The common characteristic is that these algorithms determine the Pareto dominance of two candidate solutions by computing and comparing their objective vectors, and then iteratively identifies all the non-dominated solutions in the evolutionary populations. As the number of evolutionary generation is increasing, the frequently updated non-dominated set asymptotically converges to the Pareto front.

However, for the complicated structure design optimization problems [7], [8], evaluation of the objective vectors or constraint functions is no more than computing values of simple functions. It is a time consuming procedure often taking hours or days. It is simply referred to the curse of computation cost. For the overwhelming expense evaluating objective vectors, the current popular MOEAs are almost not competent to large scale complicated multi-objective optimization tasks.

To reduce the computation expense evaluating objective vectors, GUO [9] proposed a method of predicting Pareto dominance using pattern recognition. For a

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 537–545, 2012.

multi-objective optimization problem, the method combines decision vectors of two candidate solutions into a feature vector determining their Pareto dominance. The Pareto dominance relation non-dominated, dominated and incomparable of two candidate solutions is predicted by pattern classification algorithm. No need of modeling fitness estimation and fitness inherit for objective functions of a MO problem, the predicted Pareto dominance can be used in any MO algorithms based on Pareto dominance concept, thus providing an efficient approach for relieving the curse of computation cost in solving complicated MO problems.

Based on the assumption that class-conditional probability follows normal distributions, GUO preliminarily implemented a kind of Bayesian classifiers by minimizing the classification error rate and minimizing the average risk respectively. The classifier obtained acceptable prediction accuracy on SCH [10] function. However, as the complexity of the test problems increases, the shortcoming of Bayesian classifier is evident. In most complex MO problems, the distribution of class-conditional probability is unknown a prior, the assumption of normal distribution consequently causes a number of predicting errors.

This paper proposes a method of using binary nearest neighbor classification to predict Pareto dominance. The method is used in test problems with multi-dimensional objective space and unbalanced class proportion. The rest of the paper is organized as follows. Section 2 gives the Pareto optimality concept briefly. Section 3 proposes and analyzes the binary nearest neighbor classification in detail. Section 4 presents and compares the experimental data. And the conclusions are summarized in Section 5.

2 Outlines of Pareto Optimality

A general multi-objective optimization problem can be described as a vector function **F** that maps a space of *n*-dimensional decision vectors to a space of *m*-dimensional objective vectors. Formally:

$$\begin{aligned} &\min\ \mathbf{y} = \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_m(\mathbf{x}))\ , \\ &s.t.\ \ \mathbf{x} = (x_1, x_2, \cdots, x_n) \in X\ , \\ &\qquad \mathbf{y} = (y_1, y_2, \cdots, y_m) \in Y\ . \end{aligned} \tag{1}$$

Where, **x** denotes decision vector, *X* decision space, **y** objective vector, and *Y* objective space.

Since the objectives in (1) are often conflicting with each other and incommensurable, it is impossible to optimize all objectives simultaneously. Therefore, there only exists a set of Pareto-optimal solutions.

Pareto Dominance: For every vector $\mathbf{u}=(u_1, u_2,\ldots,u_m) \in Y$, $\mathbf{v}=(v_1, v_2,\ldots,v_m) \in Y$, iff $\forall i \in \{1,2,...,m\}: u_i \le v_i \wedge \exists j \in \{1,2,...,m\}: u_j < v_j$, **u** is called to dominate **v,** denoted as $\mathbf{u} \prec \mathbf{v}$; or **v** dominated by **u**, denoted as $\mathbf{u} \prec \mathbf{v}$. Otherwise, **u** and **v** are incomparable, denoted as **u~v**.

Pareto Optimum: $\mathbf{x} \in X$ is referred as to a Pareto optimal solution (namely Pareto non-dominated solution or non-inferior solution), iff $\neg\exists \mathbf{x}' \in X,\ \mathbf{v} = \mathbf{F}(\mathbf{x}') \prec \mathbf{u} = \mathbf{F}(\mathbf{x})$.

Pareto Optimal Set: The set of all Pareto optimum in the decision space X is referred as to Pareto optimal set, and the set of the corresponding objective vectors is referred as to Pareto front or Pareto optimal surface.

3 Binary Nearest Neighbor Classification of Pareto Dominance

3.1 Prediction Model of Pareto Dominance

According to Pareto-optimality concept, Pareto dominance relation of two candidate solutions (**u**, **v**) is classified into three categories: $\mathbf{u} \prec \mathbf{v}$, $\mathbf{u} \succ \mathbf{v}$, $\mathbf{u} \sim \mathbf{v}$. Where, for the sake of convenience, **u** and **v** respectively represent n-dimensional decision vectors of the candidate solutions. Regarding a couple (**u**, **v**) as a $2n$-dimensional feature vector of a pattern, the Pareto dominance relation can be regarded as the category of the pattern, namely Pareto dominance class. Each category is labeled with class ω_i, where i=1~3. ω_1, ω_2 and ω_3 respectively represent Pareto dominance relation $\mathbf{u} \prec \mathbf{v}$, $\mathbf{u} \succ \mathbf{v}$, $\mathbf{u} \sim \mathbf{v}$.

For a MO problem, assuming two candidate solution sets with size p, q respectively are generated randomly in the decision variable domain, each sample $s(\mathbf{x}_1, \mathbf{x}_2)$ of Pareto dominance class is constructed as the way that $\mathbf{x}_1$ is selected randomly from the p solutions and $\mathbf{x}_2$ from the q solutions. Then $p+q$ candidate solutions can be be used to construct a Pareto dominance sample set with size $p \cdot q$. The class label ω_i^j of the sample $s_j(\mathbf{x}_1^j, \mathbf{x}_2^j)$ is determined by computing and comparing the objective vectors of the candidate solutions $\mathbf{x}_1^j$ and $\mathbf{x}_2^j$, where i=1~3, j=1~ $p \cdot q$. Given a sample set $S = \{(s_j, \omega_i^j)\}$ with appropriate size, by using the classifier structure shown in Fig.1 and utilizing an efficient learning algorithm, it is possible to predict the Pareto dominance class of any decision variable **u** and **v**.

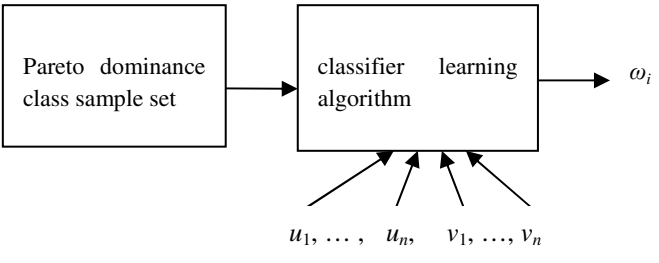


Fig. 1. Pareto dominance classifier structure: each sample consists of a decision vector couple (**u**, **v**) and a known class label ω_i. **u**, **v** are n-dimensional decision vectors of two candidate solutions respectively

3.2 Binary Nearest Neighbor Classification

For a MO problem, each dimension of the decision vector is encoded as a binary string with length l bits $x_i \in \{0,1\}^l$, where i=1~n. The value of the encoded binary string $\boldsymbol{x}_i$ is calculated by

$$x_i = a_i + \frac{(b_i - a_i) \times \sum_{k=1}^{l} x_{ik} \times 2^{k-1}}{2^l - 1} \quad . \tag{2}$$

Where, $[a_i, b_i]$ is the domain of x_i.

Given two binary strings x_i and y_i with length l, the distance $d_i(x_i, y_i)$ between the strings x_i and y_i is calculated by

$$d_i(x_i, y_i) = \sum_{k=1}^{l} (x_{ik} \oplus y_{ik}) \times 2^{k-1} \quad . \tag{3}$$

Where, $\oplus$ is the logical exclusive or operator. Then the similarity $\delta(\mathbf{x}, \mathbf{y})$ of two decision vectors **x** and **y** is measured by

$$\delta(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} d_i(x_i, y_i), \; i = 1 \sim n \; . \tag{4}$$

Given decision vector set $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, the Pareto dominance class label $C(\mathbf{u}, \mathbf{v})$ of arbitrary decision vector couple (**u**, **v**) is assigned by the binary nearest neighbor classification algorithm described as follows:

Step1. Find out $\mathbf{x}_i$, $i = \arg\min_{i=1\sim N} \delta(\mathbf{u}, \mathbf{x}_i)$;

Step2. Find out $\mathbf{x}_j$, $j = \arg\min_{j=1\sim N} \delta(\mathbf{v}, \mathbf{x}_j)$;

Step3. Let $C(\mathbf{u}, \mathbf{v}) = C(\mathbf{x}_i, \mathbf{x}_j)$, where $C(\mathbf{x}_i, \mathbf{x}_j) \in \{\omega_1, \omega_2, \omega_3\}$.

The reason of adopting encoded binary string in BNNC is that the weighted sum of binary bits reflects the similarity of two feature variables more accurately than Euclidian distance of real value space, especially for problems in which the domains of feature variables differs in orders of magnitude.

Taking the following problem for consideration:

$$\begin{aligned} &\min f_1(x_1, x_2) = x_1, \;\; f_2(x_1, x_2) = \frac{1 + x_2}{x_1}; \\ &0 \le x_1 \le 0.15, \; 0 \le x_2 \le 15 \; . \end{aligned} \tag{5}$$

The values of feature variables of two groups of samples (**a**, **b**, **c**) and (**x**, **y**, **z**) and the corresponding encoded binary strings are shown in Table 1. The Pareto dominance class label of the couple (**c**, **z**) will be determined by prediction.

Table 1. The samples and their encoded binary strings

Selected samples	Feature vectors		The encoded binary strings	
	x_1	x_2	x_1	x_2
a	0.01	10	0001	1010
b	0.12	3	1100	0011
c	0.15	7	1111	0111
x	0.01	11	0001	1011
y	0.11	4	1011	0100
z	0.07	11	0111	1011

By real-value nearest neighbor classification (RNNC) rule, the similarity δ_r depends on the Euclidean distances between samples. A few simple computation gives $\delta_r(\mathbf{a},\mathbf{c}) = 3.003$, $\delta_r(\mathbf{b},\mathbf{c}) = 4.000$, $\delta_r(\mathbf{x},\mathbf{z}) = 0.060$ and $\delta_r(\mathbf{y},\mathbf{z}) = 7.000$. It shows that **c** is closer to **a**, and **z** is closer to **x**. Then $\mathbf{c} \prec \mathbf{z}$ is obtained from $C(\mathbf{c},\mathbf{z}) = C(\mathbf{a},\mathbf{x})$, where $C(\mathbf{a},\mathbf{x})$ is obtained by evaluating and comparing the objective vectors of **a** and **x**. In fact, the evaluated Pareto dominance relation is **c~z**.

However, the similarity measured by the weighted sum of binary bits are $\delta_b(\mathbf{a},\mathbf{c}) = 27$, $\delta_b(\mathbf{b},\mathbf{c}) = 7$, $\delta_b(\mathbf{x},\mathbf{z}) = 6$ and $\delta_b(\mathbf{y},\mathbf{z}) = 27$. According to BNNC algorithm, the actual Pareto dominance relationship **c~z** can be derived from $C(\mathbf{c},\mathbf{z}) = C(\mathbf{b},\mathbf{x})$.

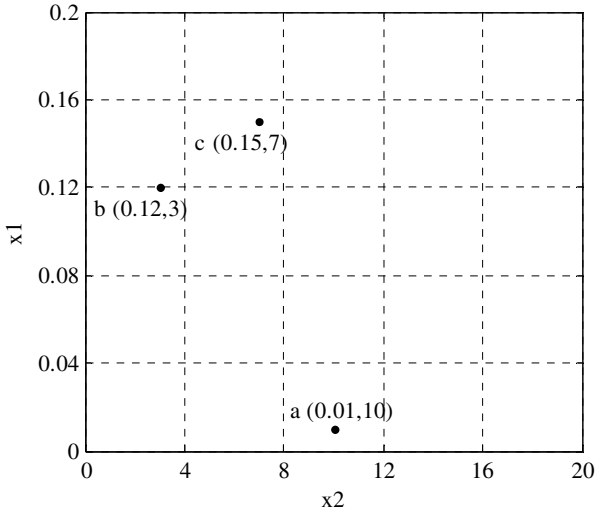


Fig. 2. The locations of sample **a**, **b** and **c** in the transformed coordinate system, where the height of domain of x_1 is enlarged to the same as the width of x_2. It is the similar case as x_1 and x_2 are encoded to binary strings with the same length.

In RNNC, Euclidean distance is calculated on original coordinate scale, the value is inclined to be dominated by the feature variable x_2 with greater interval [0,15], the contribution of the feature variable x_1 with smaller interval [0,0.15] is almost neglected on extreme cases. Consequently, the similarity measured by Euclidean distance gives the result that **a** is closer to **c** than **b**.

However, regarding the similarity of Pareto dominance, the contribution of each feature variables to the similarity computation must be identically considered. The intention can be implemented by using transformed coordinate scales as adopted in BNNC. In the transformed coordinate system, the fact **b** is closer to **c** than **a** is evident as visualized in Fig.2. Therefore, the weighted sum of binary string used in BNNC more appropriately represents the similarity of feature space.

3.3 Time Complexity

For Pareto dominance prediction of MO problems, when the canonical nearest neighbor rule[11] is used, the candidate solution sample set of size N is constructed to candidate solution couple sample set of size N^2. Therefore, the time complexity of finding out the nearest neighbor for any observed sample ($\mathbf{x}_1$, $\mathbf{x}_2$) is $O(N^2)$. But it is no need to construct candidate solution couple sample set to implement BNNC. For any observed sample $\mathbf{x}_1$ and $\mathbf{x}_2$, BNNC directly finds out the nearest neighbors from the candidate solution sample set of size N respectively. It is clear that the total time complexity is $O(N)$.

4 Experimental Results

We used BNNC and RNNC to predict the Pareto dominance for the problem in (5). The only difference is that RNNC uses real-value representation of feature variables but BNNC binary bit strings. In experiments, x_1 takes the fixed domain [0,0.1], but x_2 takes three kinds of domains [0,5], [0,50] and [0, 500].

Table 2. The average accuracy rates of BNNC and RNNC classifying the 1600 observed data over 100 sample sets. Each sample set consists of 10000 randomly generate candidate solutions. The rows correspond to the three different domain combination of feature variable x_1 and x_2.

Average class proportion in sample data (%)			Average accuracy of BNNC predicting observed data (%)			Average accuracy of RNNC predicting observed data (%)		
ω_1	ω_2	ω_3	ω_1	ω_2	ω_3	ω_1	ω_2	ω_3
11.12	11.39	77.49	71.51	70.84	93.40	61.30	70.16	90.15
13.85	14.33	71.82	76.07	78.76	91.53	50.63	42.54	78.80
14.07	14.96	70.97	80.51	78.26	89.46	38.41	39.08	73.92

For each group domain of $x1$ and $x2$, 10000 sample candidate solutions are randomly generated. The Pareto dominance relationships between samples are determined by evaluating and comparing the objective vectors. Each sample set is used to classifying the same set of randomly generated 1600 observed data. The average class proportion

over 100 random sample sets and the average accuracy rate classifying the 1600 observed data are listed in Table 2.

It shows that the classification accuracy of RNNC decreases as the difference between domains of feature variables increases. But BNNC is not sensitive to the intervals of feature variables besides the overall stronger competence in classification accuracy.

For testing the robustness of BNNC, we performed additional classification tests on problems as listed in (6)~(8), which are the popular benchmark [10], [12], [13] in testing multi-objective optimization algorithms.

$$\min \ f_1(x) = x^2, \ f_2(x) = (x-2)^2; \quad -5 \le x \le 5 \ . \tag{6}$$

$$\begin{aligned} \min \ & f_1(x_1, x_2) = [1 + (A_1 - B_1)^2 + (A_2 - B_2)^2], \\ & f_2(x_1, x_2) = [(x_1 + 3)^2 + (x_2 + 1)^2]; \\ & A_1 = 0.5\sin 1 - 2\cos 1 + \sin 2 - 1.5\cos 2, \\ & A_2 = 1.5\sin 1 - 2\cos 1 + 2\sin 2 - 0.5\cos 2, \\ & B_1 = 0.5\sin x_1 - 2\cos x_1 + \sin x_2 - 1.5\cos x_2, \\ & B_2 = 1.5\sin x_1 - \cos x_1 + 2\sin x_2 - 0.5\cos x_2; \\ & -\pi \le x_1, x_2 \le \pi \ . \end{aligned} \tag{7}$$

$$\begin{aligned} \min \ & f_1(x_1, x_2) = x_1^2 + (x_2 - 1)^2, \\ & f_2(x_1, x_2) = x_1^2 + (x_2 + 1)^2 + 10, \\ & f_3(x_1, x_2) = (x_1 - 1)^2 + x_2^2 + 2; \\ & -2 \le x_1, x_2 \le 2 \ . \end{aligned} \tag{8}$$

In the experiments, BNNC and Bayesian classifier based on the criterion minimizing the error rate and the assumption that class condition probability density follows normal distribution, are compared. As to the size of the sample set and the observed data, we adopted the same testing conditions as described above. For each algorithm and problem, the classifying trial is performed 100 runs on varied sample set.

The average class proportion of the sample sets and the average classification accuracy rate of the two methods are shown in Table 3. The resulting data in first to third row respectively corresponds to problem (6)~(8). It indicates that the classification accuracy of Bayesian classifier evidently depends on the dimension of the feature vectors, especially the unbalanced degree of class proportion in samples. For the samples of problem (6) with 2 objectives has relatively balanced class distribution,

both methods are able to get satisfying classification result, but BNNC averagely obtains higher prediction accuracy rate approximate to 98% for each class.

For the class proportions in the samples of the problem (7) and (8) with 2 feature variables are extremely unbalanced, the Bayesian classifier are almost unable to recognize the classes with miner proportion, no matter how we adjust the algorithm parameters and the size of samples. But BNNC is able to obtain acceptable classification accuracy rate about 66~75% for miner classes.

Table 3. The average class proportions over 100 sample sets, and the average classification accuracy rates of BNNC and Bayesian classifier. The rows correspond to problems (6)~(8) respectively.

Sample classes (%)			BNNC (%)			Bayesian (%)		
ω_1	ω_2	ω_3	ω_1	ω_2	ω_3	ω_1	ω_2	ω_3
34.06	33.52	32.42	97.48	97.67	97.91	82.31	86.75	91.95
14.76	15.22	70.03	66.96	70.46	87.45	20.67	22.22	94.76
15.05	15.37	69.58	74.55	72.46	90.42	33.95	30.40	98.79

5 Conclusions

Multi-objective optimization based on Pareto optimality needs to identify Pareto dominance among the candidate solutions. For extremely complicated optimization problems, population-based evolutionary algorithm may be confronted with the curse of computation cost for evaluating and comparing a large number of objective vectors. This study proposes a kind of method predicting Pareto dominance among the candidates by binary nearest neighbor classification. The algorithm encodes each feature variable into a binary string with fixed length. The similarity of two candidate solutions is measured by weighted sum of binary bits. The analysis and experiments shows that the method is able to predict Pareto dominance efficiently. In addition to get the higher classification accuracy, it is more robust than real value nearest neighbor and Bayesian classification when used in MO problems, in which the orders of magnitude of feature variable domains differ from each other and the class proportions of the samples are unbalanced.

Acknowledgments. The authors gratefully acknowledge the support from the National Natural Science Foundation of China under grant No.60975049 and 30971570, and the Natural Science Foundation of Hunan Province under grant key project No.11JJ2037.

References

1. Deb, K.: Multi-objective optimization using evolutionary algorithms: an introduction. KanGAL Report Number 2011003, Indian Institute of Technology Kanpur (2011)
2. Zhou, A., et al.: Multiobjective evolutionary algorithm: A survey of the state of the art. In: Swarm and Evolutionary Computation, vol. 1, pp. 32–49 (2011)

3. Deb, K., et al.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2), 182–197 (2002)
4. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the strength pareto evolutionary algorithm. In: Giannakoglou, K., Tsahalis, D.T., Périaux, J., Papailiou, K.D., Fogarty, T. (eds.) Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, pp. 95–100. Springer, Berlin (2002)
5. Knowles, J.D., Corne, D.W.: The Pareto archived evolutionary strategy: A new baseline algorithm for Pareto multiobjective optimization. In: Proc. of CEC 1999, vol. 1, pp. 98–105. IEEE Press, Piscataway (1999)
6. Coello, A.C.C., Pulido, G.T., Lechuga, M.S.: Handling multiple objectives with particle swarm optimization. IEEE Transactions on Evolutionary Computation 8(3), 256–279 (2004)
7. Li, X., Li, W.: Problems in complex engineering system optimization design and alternative solution. Chinese Journal of Mechanical Engineering 42(6), 156–160 (2006) (in Chinese)
8. Nain, P.K.S., Deb, K.: A multi-objective search and optimization procedure with successive approximate models. KanGAL Report 2004012, Indian Institute of Technology Kanpur (2004)
9. Guo, G., Li, W., Yang, B., Li, W., Yin, C.: Predicting Pareto Dominance in Multi-objective Optimization Using Pattern Recognition. In: 2012 International Conference on Intelligent Systems Design and Engineering Applications, Sanya, Hainan, China, January 6-7 (2012)
10. Schaffer, J.D.: Multiple objective optimization with vector evaluated genetic algorithms. In: Grefensttete, J.J. (ed.) Proceeding of the First International Conference on Genetic Algorithms, pp. 93–100. Lawrence Erlbaum, Hillsdale (1987)
11. Theodoridis, S., Koutroumbas, K.: Pattern Recognition, 4th edn. China Machine Press (2010)
12. Poloni, C.: Hybrid GA for multi-objective acrodynamic shape optimization. In: Winter, G., Periaux, J., Galan, M., Cuesta, P. (eds.) Genetic Algorithms in Engineering and Computer Science, pp. 397–414. Wiley, New York (1997)
13. Viennet, R.: Multi-criteria optimization using a genetic algorithm for determining a Pareto set. International Journal of Systems Science 27(2), 255–260 (1996)

Multi-objective Evolutionary Algorithm Based on Layer Strategy

Sen Zhao[1,2], Zhifeng Hao[1,3], Shusen Liu[4], Weidi Xu[4], and Han Huang[4]

[1] College of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China
[2] Department of Computer Science, JiNan University, Guangzhou, China
[3] School of Computer, Guangdong University of Technology, Guangzhou 510006, China
[4] School of Software Engineering, South China University of Technology, Guangzhou 510006, China

Abstract. In view of the unsatisfactory search performance of binary crossing operator as well as the elitist-preserving approach's influence on the population's diversity, an algorithm of multi-objective based on layer strategy and self-adaptive crossing distribution index is put forward on the basis of research and analysis on NSGA-Ⅱ algorithm. The algorithm will be applied to the ZDT series test functions. The experiment results show that the improved algorithm maintains the diversity and distribution of population. Compared with NSGA-Ⅱ, the Pareto front we get is much closer to the true Pareto optimal front.

Keywords: Multi-objective Evolutionary Algorithm, NSGA-Ⅱ, Pareto optimal Front, Layer Strategy.

1 Introduction

In the practice of science and engineering, many problems have multi-objective nature. They are expected to meet several different objectives, which often conflict with each other. Such problems are called multi-objective optimization problem (MOP). Unlike single objective optimization problem (SOP), the solution to MOP is an optimal solution set instead of a unique answer. All the elements in the solution set are known as Pareto optimal solution, which cannot be compared with each other concerning all objects.

Unlike SOP, multi-objective optimization solution aims at achieving two goals:

(1) Approaching to the Pareto optimal front;
(2) Maintaining the diversity of the population.

In order to achieve these two goals, researchers have proposed many methods to solve MOP, one of the methods is using the classical optimization methods, which apply the counter measure weighing principle to the relative importance of target to handle

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 546–553, 2012.

MOP before a single objective optimization can be formed. The solution to MOP has some defects: it is very sensitive to the shape of the Pareto optimal front, and it cannot handle the front of the recess; moreover the heuristic knowledge concerning application background is frequently unavailable, thus preventing the implementation of optimization. Another method is to use evolutionary algorithms and other random search algorithms. A large number of studies have shown that the evolutionary algorithm is very suitable for solving the MOP because during the evolution populations may parallel search many objectives, the Pareto optimal solutions will be finally achieved through the community evolution.

Evolutionary algorithms based on multi-objective optimization methods have got more and more attention. Having done lots of researches and proposed a variety of treatment strategies, domestic and foreign scholars have formed a number of effective multi-objective evolutionary algorithms (MOEA): such as Corne's PESA [1] and PESA-II [2], Knowles and Corne's PAES [3], Horn et al.'s NPGA [4], Zitzler and Thiele proposed SPEA [5] and SPEA2 [6], and Deb's NSGA-II [7]. These algorithms have their own advantages, but also have flaws. In the process of MOEA development and application, it is still necessary to constantly improve and refine existing multi-objective evolutionary algorithm and invent new optimization methods in order to provide more valuable research methods and tools for evolutionary algorithm to solve optimization problems.

Among the MOEA, having both a good distribution and the fast convergence rate, Deb's NSGA-II is one of the best multi-objective optimization algorithms so far, which has been widely cited by scholars from domestic and abroad. But the NSGA-II algorithm also has its own disadvantages. For example, the adaption of simulated binary crossover operator results in poor search performance, and its elitist-preserving approach affects population diversity, and the remaining infeasible solutions caused by the imperfect crowding mechanisms reduce the efficiency of evolution. Aiming at the above issues, this paper does improving study of NSGA-Ⅱ algorithm in order to increase the search capabilities and maintain the population diversity much better.

2 Analysis and Improvement of Algorithm

2.1 Analysis and Improvement of Crossover Operator

What NSGA-Ⅱ uses is a simulated binary crossover algorithm (SBX)[8] which simulates the working principle of single-point crossover operator on binary strings. Two parent chromosomes produce two children chromosomes through crossover operation, and leave the parent mode of information about the chromosomes protected in the offspring. There is a cross-distribution function in SBX operator. The cross-distribution function contains a parameter that can be any non-negative real number: cross-distribution index η^c. The index influence on the production of offspring solution has the following characteristics: If a large value of η^c is chosen, the resulting

offspring solutions are close to the parent solutions; On the other hand, for a small value of η^c, solutions are likely to stay away from parents.

According to the feature of η^c , fixed η^c will be no longer used in the process of evolution , instead we determine η^c in view of the current generation of the evolution. In the initial period of evolution, smaller η^c is used for dispersion search, which helps explore unknown space information and maintain the diversity of solutions. In the process of evolution, the individual solution tends to converge, increasing η^c gradually and adopting small-scale centralized search to improve the convergence rate. As a result, using time-related logistic function designs gradually change cross distribution index η^c.

Logistic function has the following characteristics: when time t is very small, it grows exponentially, and when t increases, the growth rate declines; until a certain value is approached.

In this function, the η^c value is calculated using the following equation:

$$\eta^c = \frac{3}{1+e^{-0.01t/2}} \tag{1}$$

where t is the current evolution generation and ηc gradually increases from 2 to 3 during the evolutionary process.

2.2 Analysis and Improvement of Parent Individual Choice

It's well known that the quality of the parent individual plays an important role in generating high-quality offspring individuals. In early search, excellent individuals are few, a lot of repeated individuals among parent individuals are generated by binary tournament selection method. It is more likely to be stuck in local optimum, and be an obstacle to the maintenance of individual diversity. So, it is necessary to limit the number of the individuals' copy to ensure that the selected parent individuals are different from each other.

2.3 Analysis and Improvement of Elitist Preserving Mechanism

Although elitist preserving mechanism is helpful to maintain excellent individuals and improve the overall evolution level of population, there are still shortcomings in the NSGA-II elitist preserving approach. When the number of individuals in non-dominated front whose order value is 1 exceeds initial population size, individuals selected by the elitist preserving mechanism are all in this front, and then the dominant solution (inferior solution) cannot play the role fully, which possibly prevents some partial areas from being searched in Pareto optimal front. So it is not conducive to the protection of individual diversity. In order to make individuals in the current Pareto front spread to the whole Pareto front and distribute as evenly as

possible, a new elitist preserving mechanism is proposed. The new method adopts layered scanning the destination space to achieve the elite individual choice.

Specific methods are as follows: in the destination space, selecting an objective function as an indicator, dividing the target space into several sub-spaces evenly, and individual quantity in each selected sub-space is calculated using the following equation:

$$m = \lceil N/i \rceil \tag{2}$$

where N is the number of the selected population, i is the number of divided space , m is the individual quantity in each selected sub-space.

According to dominance relations, a series of non-dominated Pareto solution set in each sub-space is obtained in the order of F_1, F_2, ...and so on. The level of F_1 is the highest. If the individual quantity in F_1 is larger than m, the individual in F_1 needs the crowding sort method. Individual with large crowding distance gets preferentially access to next generation P_{t+1}. Otherwise, if the individual quantity in F_1 is less than m, all members will be chosen into the next generation P_{t+1}. Remaining members of P_{t+1} will be selected among F_2, F_3,...until individual quantity approaches to m.

If the number of individuals in a sub-space is less than m, the surplus individual will be selected from the neighboring upper sub-space. The number of surplus individual is equal to m minus the individual quantity which has already been selected in current layer. If all selected individual quantity is less than population size, the random method will be adopted to select individual from the initial data set in order to achieve population size.

Layer strategy is different from the original strategy. In original strategy order value of individual in population are continual digits starting from 1, but with layer strategy, order value of individual in sub-space may not be continuous, and the smallest order value is not necessarily 1. For example, in the sub-space, only 2, 4 of the order value emerge, instead of a continual digit 1,2,3,4. So the strategy of individual selection is: first to read the value and then achieve individual choice in the same order value individuals.

3 Simulation Results

3.1 Test Functions and Performance Measures

In this section, five typical multi-objective test functions [9] of ZDT series are selected to compare the performance of the improved version algorithm proposed in this paper. These multi-objective optimization optimal fronts are convex, concave, continuous, discontinuous, uniform and non-uniform. All test functions have two objective functions. None of these functions has any constraint.

In the experiment, parameters are set: population size of 100, a real number encoding, the crossover probability 0.9, mutation probability of 1 / n (n is the number of decision variables), the evolution of generation 1000.

To demonstrate the effectiveness of the algorithm and compare with algorithms from the other literatures, this paper will adopt performance metrics of convergence and diversity proposed by Deb and others to evaluate the performance of the algorithm [7].

(1) Convergence: measured by calculating the average minimum distance between the obtained solution and the nearest member of the Pareto optimal front. This metric is defined as :

$$\gamma = \frac{1}{N}\sum_{p\in P}\min\{\|p - p^*\|, p^* \in P^*\} \tag{3}$$

where N is the number of non-dominated solutions, P* is the Pareto optimal front and P is obtained non-dominated solutions.

(2) Diversity: proposed as follows:

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1}\|d_i - \overline{d}\|}{d_f + d_l + (N-)\overline{d}} \tag{4}$$

where d_i is the Euclidean distance of adjacent solutions in Pareto solutions, $\overline{d}$ is the mean value of these distance, d_f and d_l are the Euclidean distances between the extreme solutions and the boundary solutions of the obtained non-dominated set .

Δ measures whether getting a set of solutions that evenly spread the entire Pareto optimal region. A lower value implies a better variety.

3.2 Experiment Results and Analysis

Fig.1 to 5 show typical simulation results of the test functions ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6. All non-dominated solutions obtained from the improved algorithm and the Pareto optimal front are shown in the same figure. This figure demonstrates the abilities of algorithm in converging to the true front and in finding diverse solutions in the front.

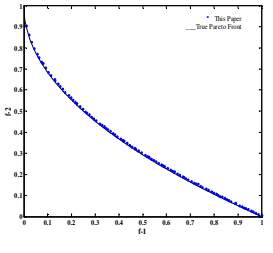


Fig. 1. ZDT1

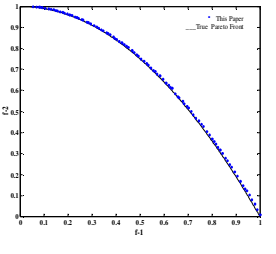


Fig. 2. ZDT2

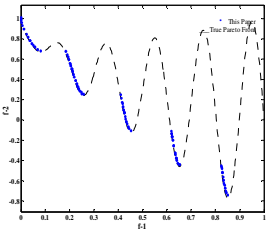


Fig. 3. ZDT3

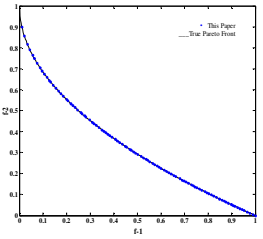


Fig. 4. ZDT4

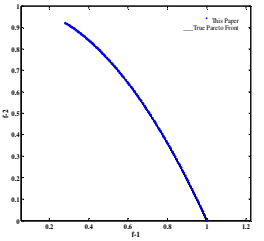


Fig. 5. ZDT6

As it is shown from the figure, the obtained non-dominated solutions and the true Pareto front almost coincide, and the solution is also more evenly distributed on ZDT1, ZDT3, ZDT6, but distribution of the solutions is not very uniform in some local areas. On ZDT2, this problem has a non-convex Pareto optimal front, where the solution is fairly distributed evenly, but in some partial areas obtained non-dominated solutions slightly deviated from the true Pareto optimal front. On ZDT4, the obtained non-dominated solutions not only are quite close to the true Pareto optimal front, but also have a good diversity and distribution.

In order to show how layer strategy enhances the performance of the algorithm, we perform simulation experiments on five typical multi-objective test functions of the ZDT series and run each configuration 10 times in each problem independently. The obtained results of convergence and diversity are shown respectively in Table 1 and Table 2. In each row of these tables, the upper cell contains the mean and the lower one contains the variance. Moreover, the first column represent the number of level by the setting L=1, L=N/5, L=N/2 or L=N.

Table 1. Mean and Variance of the convergence metric γ

The Number of Layer(L)	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
L=1	**0.0041**	**0.0089**	**0.0060**	**1.0095**	**0.0041**
	0.0000	**0.0000**	**0.0000**	**0.1881**	**0.0000**
L=N/5	**0.0047**	**0.0050**	**0.2730**	**0.9124**	**0.0048**
	0.0000	**0.0000**	**0.1094**	**0.0688**	**0.0000**
L=N/2	**0.0056**	**0.0067**	**1.4065**	**0.8309**	**0.0044**
	0.0000	**0.0000**	**0.0200**	**0.1412**	**0.0000**
L=N	**0.0084**	**0.0108**	**1.6502**	**0.3811**	**0.0046**
	0.0000	**0.0000**	**0.0012**	**0.0520**	**0.0000**

Table 2. Mean and Variance of the diversity metric Δ

The Number of Layer(L)	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
L=1	**0.7048**	**0.7705**	**1.2899**	**0.8877**	**0.8899**
	0.0015	**0.0139**	**0.0028**	**0.0011**	**0.0002**
L=N/5	**0.6511**	**0.6422**	**1.2348**	**0.8840**	**0.8325**
	0.0011	**0.0000**	**0.0095**	**0.0007**	**0.0001**
L=N/2	**0.5855**	**0.5491**	**1.1209**	**0.8372**	**0.7992**
	0.0028	**0.0023**	**0.0036**	**0.0079**	**0.0005**
L=N	**0.3690**	**0.3300**	**0.7931**	**0.6041**	**0.6371**
	0.0001	**0.0002**	**0.0000**	**0.0184**	**0.0000**

As indicated from the tables, the layer strategy got the lowest mean for both metrics in all problems. Furthermore, in 4 of 5 problems with the increase of the number of layer, convergence is gradually becoming larger and larger, while the diversity has gradually become smaller. But for ZDT4, convergence and diversity are both gradually becoming smaller with the increase of layer.

4 Conclusions

This paper has designed an adaptive cross distributed index and improved NSGA-Ⅱ elitist preserving approach. A multi-objective evolutionary algorithm is proposed based on adaptive crossover and layer strategy. Having run the improved algorithm on five different problems, we compared the results with NSGA-Ⅱ. The experimental results have shown that the improved algorithm outperformed NSGA-Ⅱ in convergence to the Pareto optimal front and in diversity of the final non-dominated solutions. However, during the experiment we have found that it is also necessary to discuss further about how to effectively balance the convergence and diversity to make the algorithm more practical.

Acknowledgments. This work is supported by National Natural Science Foundation of China (61003066, 61070033), Doctoral Program of the Ministry of Education (20090172120035), Guangdong Province Science and Technology Project(2010B010600025), The Fundamental Research Funds for the Central Universities, SCUT(2012ZM0083) and The Pearl River Science technology Star Project (2012-07).

References

1. Corne, D.W., Knowles, J.D., Oates, M.J.: The Pareto Envelope Based Selection Algorithm for Multiobjective Optimization. In: Deb, K., Rudolph, G., Lutton, E., Merelo, J.J., Schoenauer, M., Schwefel, H.-P., Yao, X. (eds.) PPSN VI 2000. LNCS, vol. 1917, pp. 839–848. Springer, Heidelberg (2000)
2. Corne, D.W., Jerram, N.R., Knowles, J.D., Oates, M.J.: PESA-II: Region_Based selection in evolutionary multiobjective optimization. In: Proceedings of Genetic and Evolutionary Computation Conference, GECCO 2001, pp. 283–290 (2001)
3. Knowles, J.D., Corne, D.W.: Approximating the non-dominated front using the Pareto archived evolution strategy. Evolutionary Computation 8(2), 149–172 (2000)
4. Horn, J., Nafpliotis, N., Goldberg, D.E.: A niched Paretogenetic algorithm for multiobjective optimization. In: Proceedings of 1st IEEE Congress on Evolutionary Computation, pp. 82–87 (1994)
5. Zitzler, E., Thiele, L.: Multi-Objective evolutionary algorithms: A comparative case study and the strength Pareto approach. IEEE Transactions on Evolutionary Computation 3(4), 257–271 (1999)

6. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35,CH-8092 Zurich, Switzerland, Technical Report 103, Computer Engineering and Networks Laboratory (TIK) (May 2001)
7. Deb, K., Pratap, A., Agrawal, S., Meyarivan, T.: A Fast Elitist Multiobjective Genetic Algorithms: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2), 182–197 (2002)
8. Deb, K., Agrawal, R.B.: Simulated binary crossover for continuous search space. Complex System 9(2), 115–148 (1995)
9. Zitzler, E., Deb, K., Thiele, L.: Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. Evolutionary Computation 8(2), 173–195 (2000)
10. Xie, T., Chen, H.-W., Kang, L.-S.: Evolutionary Algorithms of Multi-Objective Optimization Problems. Chinese Journal of Computers 26(8), 997–1003 (2003)
11. Zheng, J.-H., Jiang, H., Kuang, D., Shi, Z.-Z.: An Approach of Constructing Multi-Objective Pareto Optimal Solutions Using Arena's Principle. Journal of Software 18(6), 1287–1297 (2007)
12. Lei, D.-M., Wu, Z.-M.: Crowding Measure Based Multi-Objective Evolutionary Algorithm. Chinese Journal of Computers 28(8) (2005)
13. Gong, M.-G., Jiao, L.-C., Yang, D.-D., Ma, W.-P.: Research on Evolutionary Multi-Objective Optimization Algorithms. Journal of Software 20(2), 271–289 (2009)

Priority Based Multi Robot Task Assignment

Rahul Goyal, Tushar Sharma, and Ritu Tiwari

Department of IT, ABV-IIITM, Gwalior
rahul2.1989@gmail.com

Abstract. In this paper, we designed and developed three new model related to task assignment in robotics. The aim is to allocate all the robots to all the available tasks such that all tasks are finished with minimum total cost and minimum time taken. The first model is completely based on priority but waits for robots and allocates task only to the most suitable robots (highest bid robots). The second model completes the task in such a way that it allocates best robots that are currently available to the task without waiting for the most suitable robots to be free. The third and final model uses heuristics based approach with auction algorithm to identify the non-performing slow robots and eliminate them from the list of available robots. This helps in increasing the efficiency of the whole system and helps to reduce the total cost of performance for the system. The work aims at providing a best suitable algorithm for completing all the tasks with minimum overhead and maintaining a specific order for the completion of the tasks. Further it also rejects the slow working robots so that the total time taken for completion can be reduced.

Keywords: Priority, Multi Robot, Task Assignment, Heuristics.

1 Introduction

The problem of task allocation in multi-robot systems has received significant amount of interest in the research community. Researchers build, cooperate and design the multi-robot systems with great complexities but one question always remains in their mind i.e. “Which robot should be selected for a task?”[1]

Multi robot systems are one of the most complex and hard to interpret systems in today’s AI systems. It consists of many parameters to optimize and is generally multi-objective so as to provide best result. Some of the complexities in the multi-robot system are:

Area Exploration: First and foremost part of any practical implementation of multi robot system is to explore the area or the grid in which we are working. We have to find where are the obstacles, which parts are designated location (goal points) and what are the limits for the system. It is in itself an area of research for exploring the grid via multiple robots.

Task Assignment: After the area is explored we have to examine the number of tasks and number of robots in the grid so that we could allocate the robots to the task [4][5]. It

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 554–563, 2012.

is also one of the most thoughtful areas of research because we have to optimally select the sequence and priority of the task to be completed via multiple robots. It is done by considering various aspects of conflict resolution, minimization of the total cost, types of robots, etc [9][13].

Path Planning: After the area exploration and task assignment, we have to find the best suitable path from current position of the robot to the destined position of the task. It is done keeping in mind that the robots do not collide into the obstacles or into other robots and finding the minimum distance between the two positions [3]. Coordination and collaboration: After the completion of all the above three steps, the final step of coordination and collaboration comes into picture. In this, robots have to align themselves to complete the task they are assigned to with min resource utilization and max efficiency. It is however one of the least explored areas in robotics as it is very difficult to collaborate multiple robots to do a single job at once [21][22][23].

2 Gaps in the Existing Literature s

The available literature guides us through various available techniques for task assignment used in modern AI research viz. set-precedence [1], bidding [11], dynamic perception [14][12], etc. But all these solutions are best up to a certain limit and have one common flaw in their architecture i.e. to identify the difference between the robots ability and importance of a task. Currently, all the techniques available use some random variable or some mathematical model to formulate a relationship between the robots and tasks, but fail to examine the importance of the tasks and which robot is most suitable for a given task. We tried to solve this problem using priority concept with heuristics and successfully implemented our model using auction algorithm.

3 Problem Statement

Suppose there are ‘n’ robots (r1, r2,..,rn) and m tasks (t1,t2,...,tm) in a grid of size (max_size_w, max_size_h) and for each task, ‘x’ robots are required (x1,x2,...,xm) to complete it successfully with each task having an assigned priority ‘p’ (p1,p2,...,pm). We further sub divide ‘n’ robots into sub-groups ‘n1’,’n2’… which will perform the tasks in a simultaneously such that available robots is less than or equal to the number of robots required for next task j. Each of the robots is assigned a cost factor ‘C’ which denotes the total cost for performing the task (in this case, it is the linear sum of the distance travelled) and a bid value ‘B’ which denotes the ability of robot to perform the task (in this case, it is the mathematical relationship between the distance of the roboti and taskj). Now the objective of the problem is to complete all the tasks assigned to the robots with min total cost Cij and min time (excluding the waiting time) where,

4 Methodology

The following flowchart gives a brief idea as to how our algorithm assigns task to various robots:

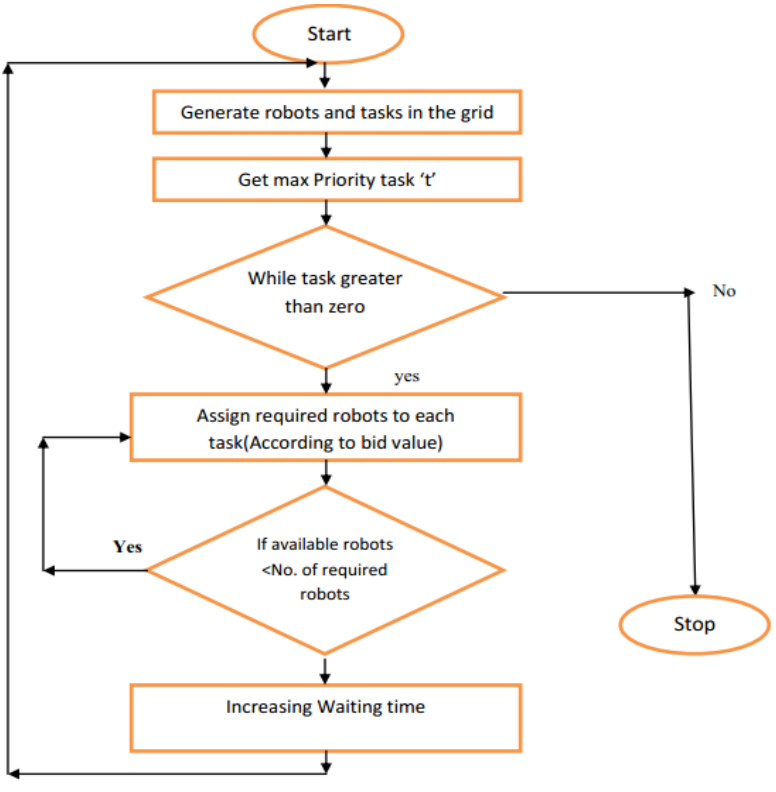


Fig. 1. Select a task and assigning required robots that can executes the task

Task assignment is a continuous process of selecting a task 'j' from a set of available tasks and then assigning robots 'x_j' robots to complete the task. It is a rigorous process to examine and monitor the changes in the current system and counter it with new formulation and identification for an important task. Task assignment is one of the key factors in order to determine the total cost for completion of whole project because the order in which the tasks are to be performed if not done properly constitutes to an extra much larger burden than any other part of robotics. Let us assume the tentative cost for performing the $task_j$ by a $robot_i$ is C_{ij} that is defined as the cost to reach that task and some attribute to identify and quantify the cost incurred in performing the task. Now, the second part is to calculate the bid for auctioning that could be defined as the expectancy of a $robot_i$ to perform the $task_j$.

Cost: Cost method is defined as the expected number of blocks to be travelled by the $robot_i$ to the $task_j$. It is assumed that the cost to travel a single block is unity and translation is only in horizontal and vertical directions [6][7].

Function 2.1 $Cost\_Method\ (\ Robot_i, Task_j\ )$

$C_{ij} = |\ Robot[i].pos_i - Task[j].pos_j\ |;$

$0 <= i < n; 0 <= j < m$

Bid: Bid function is defined as an expectancy of $robot_i$ to opt for the $task_j$. Consider a scenario of auctions, the clients ($robot_i$) bid on a specific item ($task_j$). If a $robot_i$ bid on a $task_j$, it does not guarantee that the $task_j$ is to be assigned to the $robot_i$. However, it provides an opportunity for the $robot_i$ to complete the $task_j$.

Mathematical formulation of bidding is as follows:

Function 2.2 `Bid_Method(Robot[i].pos`$_i$`,Task[j].posj)`

$$bid_{ij} = \begin{cases} \dfrac{1}{e^{|x_i - x_j| + |y_i - y_j|}} & x_i \; != x_j \text{ AND } y_i \; != y_j \\ \dfrac{1}{e^{|x_i - x_j|}} + \dfrac{1}{e^{|y_i - y_j|}} \Big/ 2 & x_i = x_j \text{ OR } y_i = y_j \\ & x_i = x_j \text{ AND } y_i = y_j \end{cases}$$

Fig. 2. Calculate the bid value

The representation is one of the most suitable formulations as it clearly and easily represents the relation between the distance and the bids.

Heuristics: Heuristics are termed as an experience based techniques that are primarily used for learning, discovery and problem solving [10]. It is used to find a satisfactory solution in minimum.

Time: where brute-force approach is impractical. We undertook heuristics in a different sense to monitor and identify different types of robots and task. It is implemented in order to get an in-depth knowledge about the limits of a particular robot and then help this knowledge as an addition factor in selecting the robot for a particular task [17].

Heuristics are initialized with a value to zero and then updating its value periodically based on the time taken to complete the task by the robot.

5 Algorithm

Assumptions: The first part of any algorithm is to find out the bounds and initialize the variables with respect to those bounds. The independent variables in our approach are:

I. Number of tasks: m
II. Number of robots: n
III. Number of robots for a task: r_j, $0 <= j < m$
IV. Size of the grid: ($size_w$, $size_h$)
V. SPriority of each task: p_j, $0 <= j < m$
VI. Velocity of each robot: vel

Now based on the above variables we calculate our dependent variables viz.:

a) Position of the task: post (x_t, y_t) : 0<= x_t<$size_w$, 0<= yt<$size_h$
b) Position of the robots: posr(xr,yr); 0<= x_r<$size_w$, 0<= yr<$size_h$
c) for all j =0 to m $completed_j$ = 0
d) for all i = 0 to n $busy_i$=0
e) for all i=0 to m
for all j = 0 to n
C_{ij}= cost_method($robot_i$,$task_j$)
f) for all i=0 to m
for all j = 0 to n
B_{ij}= bid method ($robot_i$,$task_j$)

Note that in the heuristics based approach we also initialized the Hij variable to 'zero'.

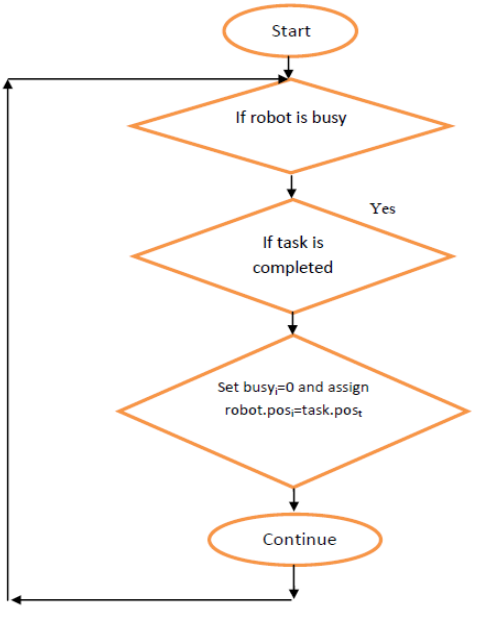


Fig. 3. Thread to explain how exactly the busy robots are set to free

Models: We devised three different models to understand the basics of task allocation using priority in auction algorithm. Let's understand the key points in each of the three models

Model – I: It assigns only the maximum bid robot to the highest priority task.
Advantage: Higher priority task completed first
Disadvantages:

Huge amount of waiting time for completion of all the tasks
Objective function is much larger than expected

Model – II: It assigns maximum bid available robots to the highest priority task.
Advantages:

Reduce waiting time in the system as compared to Model – I.

Reduce the total cost in completion of the task.

Disadvantages:

Do not consider the faulty or incapable robots.

Time lagged during non-performance not considered.

Model – III: It assigns maximum bid and minimum heuristics available robots to the highest priority task.

Advantages:

Robots non-performing or slow in performance would be barred from performing the task by considering heuristics.

Only efficient robots perform the entire task.

Robot Allocation: After the task selection process is complete, we have to allocate robots for the selected task. We have examined various approaches and concluded on two most suitable techniques that are implemented in Model – I and Model – II. In Model – I, task is allocated only to the xe highest bid robot irrespective of their availability i.e. if that robot is not available or performing another task then the system waits for it to be free and doesn't allocate the task to any other available robots Whereas in Model – II the task is allocated to highest bid available robots. Hence clearly we can see that in the second model waiting time is reduced drastically but efficiency to perform the task better is more in Model – I. After close examination of the truthfulness of the above statement, we came to a conclusion that in the Model – III we would be implementing the approach discussed in Model – II.

Objective Function: Objective function 'D' is defined as the total cost incurred by all the robots in performing all the tasks. The cost here is defined as the exact cost of the robot in moving from current position to the destined position and the time taken to perform the task.[15] Objective function gives us an idea of the amount of movement the robots have to do in order to complete all the tasks. This is very much essential when implementing the system in robotics as it gives us an estimated idea of how much movement the robotics arm has to go through in order to complete the tasks. Hence not only improving the system efficiency but also providing an estimate time for quality checking and up gradation of the system.

6 Testing

Maximum Resource Utilization: One of our prime goals is to maximize the resource utilization i.e. not allowing any robots to be sitting idle. This was achieved in Model – II by allocating the tasks to available robots rather than waiting for most suitable (highest bid) robot to be free.

Waiting Time: The waiting time is termed as the time lost in waiting for the robots to be available and the time lost in waiting for the tasks to be completed. We cannot control the waiting time for tasks to be completed because we don't know when the task is to be completed.[19] But the factor that we could control is the waiting time for the robots. In the Model – II where we do not wait for the robots to be available we reduced the waiting time for robots by nearly half.

Thereby completing all the tasks in minimum duration it helps in improving the overall efficiency of the system. The Model – II can be justified in a sense that if task has to wait for a robot that has the highest bid on it, then most certainly it is either performing a task or is malfunctioned. If that robot is performing a task, then the task it is performing must have the priority higher than the current task hence, making it more intriguing that robot should perform that task with more efficiency and a small amount of error can be accounted for in the lower priority tasks.[16] In the second case when it is malfunctioned, it is best not to wait for that robot and allow other robots to perform the tasks because the repairing cost would be too much higher than the total cost of the system.

7 Results

Priority Based MRTA without Heuristics: First we fixed the number of robots and analyzed the pattern by varying number of tasks. It however provides us with random positions of robots and tasks, but the overall value of both the objective function and the time required to complete all tasks (calculated via monitoring system time) lie in the same region. It is to be noted that as the number of tasks increases, the value of objective function increases and reaches to a stable position.

When considering the time required for completing all the tasks, it however starts with low value but increases drastically. This shows that as the number of tasks increases the cost (all robots considered) increases but will be nearly stable for large number of tasks but the time required for completing all the tasks would be increasing in an exponential fashion. Hence, there should be an equal balance when considering large number of tasks.

Table 1. Results of Priority based multi robot task assignment without heuristics considering number of robots as fixed and number of tasks as variable

No. of Robot	No. of Task	Objective Function Value	Time required to complete all tasks (ms)
4	5	14.0	321074
4	10	42.0	267431
4	15	54.0	84697
4	20	56.0	70500

Priority Based MRTA with Heuristics: This approach basically eliminates time taking robots from the task selection process i.e. robots which are not able to complete the task in a specific time updates their Hi value by a factor (which we took unity in our case) and then re-bids to calculate their new B_{ij}.[18] The Table 2 gives us an idea as how the Hi value is useful, the objective function is increasing in nature but the steps of increase is much lower than those of without heuristics. Also, the time required completing all the tasks is much less in heuristics based approach as when comparing with earlier approach. This shows that robots that are more suitable for performing a task should be given the responsibility of completing the task. Hence making it certain that this approach is better than earlier without heuristics one.

Table 2. Results of Priority based multi robot task assignment without heuristics considering number of robots as variable and number of tasks as fixed

No. of Robot	No. of Task	Objective Function Value	Time required to complete all tasks (ms)
5	5	12.0	28109
5	5	16.0	48038
5	5	18.0	19118
5	5	20.0	25426

8 Discussion

Priority Based MRTA: There are many scenarios in real world where a specific task is more important than other tasks on in a way that we have to follow a specific order in completing all the tasks. If we don't want any ordering of tasks in our system then we can have same priority to multiple tasks and the system would select a task at random making it work like a normal process.

Applications of priority based multi robot task assignment: construction business, etc.

Priority Based MRTA with Heuristics: Priority based MRTA with heuristics could be useful in scenarios where robots fail frequently and robot which are not able to perform up to a required expectancy are to be rejected or removed so as to keep minimum efficiency of the system. This approach deals with all the constraints related to robot and its failure rate to perform the tasks. Even if a robot performs the tasks, this approach takes care of the efficiency by which the robot completing the task. Hence making the system aware of the quality and minimizing the total time by removing all non-useful robots.

Applications of priority based multi robot task assignment with heuristics: Research facilities, scientific laboratories, astronomical.

Why better than previous techniques: Following are few of the key points by which our designed technique is better than previous available techniques:

1. Ordering of task considered.
2. Robots are allocated based on bidding and distance between the current position and destined position.
3. Quality of tasks performed by a robot is considered.

All robots are able to perform the tasks and follow divide and rule approach. Hence reducing the overall cost of the system.

9 Conclusion and Future Scope

After the implementation of our algorithm and analyzing the test results we came to the following conclusions.

Model – I is useful only when we need specific robot to complete a task else in other cases it introduces huge amount of waiting time which is harmful for the system.

Model – II is useful when we have to finish all the tasks quickly with an optimum quality. We are not concerned about how well and how fast a robot is performing a task. Rather we are interested in whether the robot is able to perform the task or not.

Model – III is useful when we have to consider not only the robots completing the task but also

This project could be extended much further to help collaborate and coordinate multiple robots such that a task could be divided into segments and each segment is to be completed by different sets of robots. Moreover, optimization techniques like PSO could be used which not only help the system reduce its total cost but also provide an optimal solution for completing all the tasks.

References

1. Luo, L., Chakraborty, N., Katia, S.: Multi-Robot Assignment Algorithm for Tasks with Set Precedence Constraints. In: IEEE International Conference on Robotics, pp. 2526–2533 (2011)
2. Oyama, N., Liu, Z., Gueta, L.B., Ota, J.: Rearrangement Task of Multiple Robots Using Task Assignment Applicable to Different Environments. In: IEEE International Conference on Robotics, pp. 300–305 (2010)
3. Batalin, M.A., Sukhatme, G.S.: Using sensor network for distributed multi-robot task allocation. In: IEEE International Conference on Robotics (2004)
4. Wawerla, J., Vaughan, R.T.: A fast and frugal method for team-task allocation in a multi-robot transportation system. In: IEEE International Conference on Robotics (2010)
5. Kim, Y.H., Kim, B.K.: A multi robot task planning system minimizing the total execution time for hospital service. In: International Conference on Control, Automation and Systems (2010)
6. Fukuda, T., Kubota, N.: Intelligent robotics systems from a single robot to multiple robotics system. In: International Workshop on Robotics (2002)
7. Chen, J., Sun, D.: An experimental study on leader follower coalition method for solving multiple robotic task allocation. In: International Workshop on Robotics (2010)
8. Duvallet, F., Stentz, A.: Imitation learning for task allocation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2010)
9. Zheng, T., Li, J.: Multi-robot task allocation and scheduling based on fish swarm algorithm. In: World Congress on Intelligent Control and Automation (2010)
10. Shi, Z., Wei, J., Wei, X., Tan, K., Wang, Z.: The task allocation model based on reputation for the heterogeneous multi-robot collaboration system. In: World Congress on Intelligent Control and Automation (2010)
11. Kaleci, B., Parlaktuna, O., Ozkan, M.: Market-Based Task Allocation by using Assignment Problem. In: IEEE International Conference on Systems, Man, and Cybernetics (2010)
12. Matarić, M.J., Sukhatme, G.S., Østergaard, E.H.: Multi-Robot Task Allocation in Uncertain Environments. In: International Workshop on Robotics, pp. 255–263 (2002)
13. Gerkey, B.P., Matarić, M.J.: A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. International Journal of Robotics Research 23 (2002)
14. Gerkey, B.P., Matarić, M.J.: A Framework For Studying Multi-Robot Task Allocation. In: Proceedings of the NRL Workshop on Multi-Robot Systems (2003)

15. Matarić, M.J.: Reinforcement Learning in the Multi-Robot Domain. Autonomous Robots, 73–83 (1993)
16. Lo, V.M.: Heuristic algorithms for task assignment in distributed systems. IEEE Transactions on Computers 37(11), 1384–1397 (1988)
17. Efe, K.: Heuristic models of task assignment scheduling in distributed systems. Computer 15(6), 50–56 (1982)
18. Kafil, M., Ahmad, I.: Optimal task assignment in heterogeneous distributed computing systems. IEEE Concurrency 6(3), 42–50 (1998)
19. Burgard, W., Moors, M., Stachniss, C., Schneider, F.E.: Coordinated multi-robot exploration. IEEE Transactions on Robotics 21(3), 376–386 (2005)
20. Shen, C., Tsai, W.: A Graph Matching Approach to Optimal Task Assignment in Distributed Computing Systems Using a Minimax Criterion. IEEE Transactions on Computers C-34(3), 197–203 (1985)
21. Kopidakis, Y., Lamari, M., Zissimopoulos, V.: On the Task Assignment Problem: Two New Efficient Heuristic Algorithms. Journal of Parallel and Distributed Computing 42(1), 21–29 (1997)
22. El-Rewini, H., Lewis, T.G.: Scheduling parallel program tasks onto arbitrary target machines. Journal of Parallel and Distributed Computing 9(2), 138–153 (1990)
23. Tovey, C., Lagoudakis, M., Jain, S., Koenig, S.: The Generation of Bidding Rules for Auction-Based Robot Coordination. Springer, Netherlands (2005)

A Survey of Swarm Robotics System

Zhiguo Shi[1], Jun Tu[1], Qiao Zhang[1], Lei Liu[1], and Junming Wei[2]

[1] School of Computer and Communication Engineering,
University of Science and Technology Beijing
100083 Beijing, P.R.China
[2] ANU College of Engineering and Computer Science,
Australian National University Canberra, 2601, AUS
{szg,jtu,qzhang,liulei}@ustb.edu.cn, wei@anu.edu.au

Abstract. Swarm robotics system has been a particularly active topic of robotics in recent years due to the increasing deepening of research on robotics technology and application. This paper gives a survey of swarm robotics system research from such aspects as theoretical basis and physical research, simulation platform, distributed control information fusion and communications system. Some problems that need to be solved about swarm robotics system research in IOT (Internet of Things) environment are also raised, such as co-adaptation, distributed control and self-organization, resource scheduling management. Finally, the ant colony algorithm and particle swarm optimization are applied to the swarm robotics system.

Keywords: Swarm Robotics, Swarm Intelligence, Distributed Control.

1 Introduction

Modern research on robotics began in 1948 when the Institute of Atomic Commission Argonne of United States developed mechanical master-slave manipulator. The main founder of Robotics Academy in China is Prof. Xinsong J, who gave the system definition and research on various areas of robotics from the controlling point [1]. Single robot is limited in information processing capability and many other aspects, thus the cooperation of multi-robot are needed to complete the task in an efficient way, which results in the birth of multi-robot systems.

Multi-robot researches began in the late 1980s, the previous researches mainly concentrated in three areas [2]: 1) assembly robot system; 2) multi-robot motion planning; 3) multi-robot cooperation framework. In recent years, research on multi-robot systems have make great progress and achieve great success in many areas [3]. Cooperative multi-robot system depends on the individual robot very much, for further enhancing the robustness and scalability of the system, swarm robotics system was born.

The swarm robotics research began in the late 1990s. The project of Swarm-bots [4] chaired by the inventor of the ant colony algorithm Professor Marco Dorigo [5] began in 2000 marked that the studies on swarm robotics came into a new period of

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 564–572, 2012.

development. Swarm robotics Professor Erol Sahin [6], one of the founders, gave a definition of swarm robotics [7]: it studies that how to make lots of robots with relatively simple physical structure show the expected overall behavior through the local interactions among robots and between the robots and the environment. The characteristics of various robot systems are shown in Figure 1.

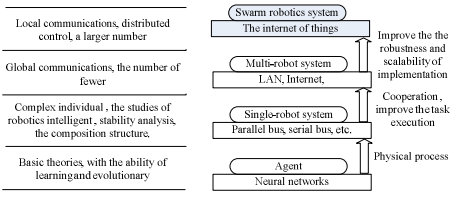


Fig. 1. The characteristics of various robot systems

2 Research Background

The research inspiration of swarm robotics comes from observing the behavior of insects, such as ants, termites, wasps and bees. Swarm robotics system is dynamic self-organization and has good fault tolerance, the task of robot is not pre-allocated and single invalid robot will not affect the operation of the system. Meanwhile, the whole system will not be collapsed by the failure of control center because it adopts distributed controlled. The research of swarm robotics has attracted many researchers for its high robustness and scalability.

The journal "Autonomous Robots" published a special issue of swarm robotics [5] in September 2009, which was edited by the expert of swarm intelligence Marco Dorigo and Erol Sahin. The special issue pointed out a clear standard to distinct the researches similar with swarm robotics, such as Collective Robotics, Distributed Robotics, and Robot Colonies. The standard includes four aspects: 1) large quantities and scalability; 2) roughly the same structure in each robot team; 3) need to rely on each other to complete the task; 4) individual robot has the capability of local sensing and local communications.

Subsequently, the academic Journal "Swarm Intelligence" published a special issue of swarm robotics [6] in 2008 which divided the study into three levels: system design & algorithms, research tools and modeling & analysis.

2.1 Theoretical Basis and Progress of Physical Research

Swarm intelligence theory provides important theoretical basis for swarm robotics research and includes [8]: ACO, PSO, OT and GT, etc.

In 1989, Beni and Wang [9] first proposed the swarm intelligence concept when studied the distributed mobile robot. In the early 1990s, the ASO (Ant Colony Algorithm) first proposed by Macro Dorigo, etc [10] has laid a foundation to the swarm intelligence theory. Ant colony algorithm is a new heuristic algorithm based on bionics

that attracted more and more scholars for its distributed concurrency, positive feedback, robustness, fast convergence, easy access to the global optimum solution, etc. Since 2000 several development and controlling of swarm robotics research plans have been developed, for example the Commission of the European Communities (CEC) emerging technology program to support the swarm robotics study. Representative swarm robotics systems are shown in Figure 2.

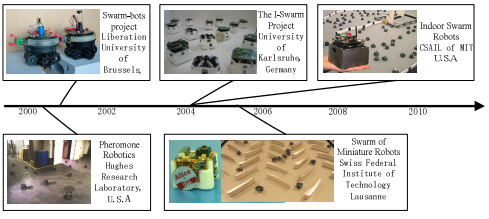


Fig. 2. Representative swarm robotics systems

Domestic research of swarm robotics started in the beginning of this century, and is still at the initial stage. The research content of swarm robotics was proposed by Prof. Tan [11] in 2001 is the earliest research about swarm robotics. Subsequently, many domestic research institutes also started relative research on swarm robotics such as: the research of conformation expression and reconstruction optimization of reconfigurable swarm robot in Shenyang Institute of Automation [12], the research on self-assembly modular swarm robot hosted by the robotics institute of Beijing University of Aeronautics & Astronautics [13], etc.

2.2 The Progress of Simulation Platforms

The research of swarm robot system needs lots of physical robots, which makes it hard to afford for many research institutions. Usually a good simulation platform will achieve a multiplier effect. Some commonly-used simulation platforms [14] are Player/Stage, TeamBots, Gazebo, USARSim, Swarmbot3d, Swarmanoid Simulator, etc. Player/Stage is the representative simulation platform and developed by Robotics Research Laboratory in University of Southern California in 1999, it can provide internal interface and simulation environment for multi-robot systems. See Fig.3.

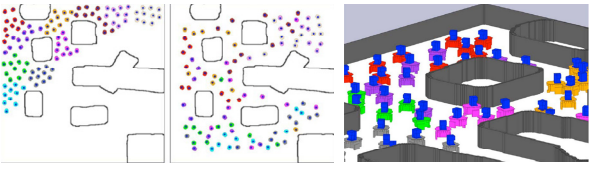


Fig. 3. The simulation scene

2.3 Progress on Distributed Control of Swarm Robotics

Both distributed and centralized approach [15] can be used in the multi-robot control system, but the swarm robotics system generally adopts the distributed control. The research of Distributed control is divided into three areas: movement and formation control, distributed learning, and coordination & task allocation. Among them, the movement and formation control include: method based on behavior and rules, Leader-Follower method, method based on graph theory.

Method based on behavior and rules: providing the robots with some expected behaviors including collision avoidance, obstacle avoidance, and formation maintain, etc. According to the inputs of sensors, the robot responds and output the response vectors as expected response of the behavior.

Leader-Follower method: designing a robot as leader in the group of swarm robot, the rest are followers which track the location and direction of the leader in a certain distance interval and achieve collaboration by sharing status information of the leader robot.

Method based on graph theory: representing the dynamics or kinematics of robot by the graph nodes, the edge between nodes indicates the constraints between the robots and then according to the graph theory and control theory analyzing the stability of the formation which indicated by graph [16]. The main advantages and disadvantages of these four methods are shown in Table 1.

Table 1. Compare between movement and formation control methods

Methods	Advantages	Disadvantages
Behavior-based control method	parallelism, distribution and real-time, good adaptive for many competing objectives	Group behavior cannot be clearly defined, cannot guarantee the stability of formation
Leader-Follower method	The behavior of swarm robot can be easily controlled by the behavior or track of leader	No clear formation feedback, if the leader fails, the entire formation will not maintain
method based on graph theory	Use maps can represent any formation, a mature theory of graph theory as a basis	Largely confined to a small number of robot control, and more complex to achieve

Distributed learning includes [13]: group reinforcement learning and individual reinforcement learning. Group reinforcement learning use combination actions, the action to be taken by all other robots should be taken into account when a robot determine their own actions, which can be seen as a tightly coupled distributed reinforcement learning system.

Independent reinforcement learning can be seen as a loosely coupled distributed reinforcement learning system. For example: in the experiment of pushing and pulling the stick, the number of sticks and robots can be changed by adjusting the robot's

waiting time [17]. The independent reinforcement learning is more suitable for the characters of swarm robotics system.

2.4 The Progress of Research in Swarm Robotics Information Fusion

In the unstructured environments, robot can make the correct decision just by obtaining a variety of information. The core of the development of robots in unstructured environments is the multi-sensor system and information fusion [18]. The research of information fusion technology received more attention from scholars in the early 1980s. In 1984, America set up Data Fusion Subpanel (DFS), in 1988 "Integration, Coordination and Control of Multi-sensor Robot Systems" written by Durrant-Whyte, laid the foundation for the research of multi-sensor information fusion. LUO [19] proposed the four advantages of fusion: redundancy, complementary, timeliness and information cost.

Information fusion technology is the high level of key technology which common concerned by multi-disciplinary and multi-field. The information fusion research focused on man-machine interaction and path planning two aspects.

2.5 Progress of Research on Swarm Robotics Communication System

Communication is a basis for information exchange between robots and achieving collaboration which can strengthen the link between robots so that the robot system use more advanced strategies to coordinate, thus improving the ability to complete complex tasks [20]. Robot communication system mainly includes communication, communication language, and communication network architecture and communication protocol [21].

Communication methods include explicit communication and implicit communication, for swarm robotics system it is generally implicit communication, there is no global rules and way to achieve transfer of information with specific meaning among robots.

SWARMORPH-script [22] is the communication language for swarm robotics in Swarm-bots project which accurately describe the rules of the form growth in the self-assembly process of the robot. In addition, the robot system communication language can also use: Agent Communication Language, Knowledge Query and Manipulation Language.

Currently Robot communication network mainly uses: Wi-Fi network, GPRS communication, Ad hoc networks [21] and wireless sensor networks. The wireless sensor network with the advantages of high robustness and self-organization is applied to network platform of swarm robotics.

3 The Simulation Scene of Swarm Robotics

The self-organizing of ant colony in nature have attracted the attention of entomologist long time ago, Deneubourg [23] et al. developed a study on the foraging behavior of ant

colony by "double bridge experiment". The symmetric double-bridge (the two bridges have same length) A, B will be separated from the nest and food source, ants can moved from the nest to the food source freely, shown in Fig. 4.

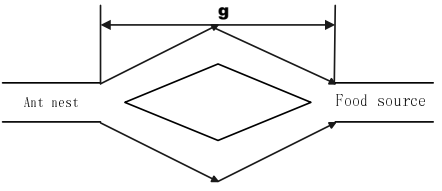


Fig. 4. The symmetric double-bridge model

In the early stage, there is no pheromone in two bridges, every ant will choose bridge A and bridge B at the same probability, so the pheromone left in two bridges is equal. After a period of time, the most ants choose bridge A for some random fluctuations, resulting bridge A attracts more ants with more pheromone left on it. As time goes by, the number of ants who choose bridge A will be more and more, and bridge B just the opposite.

Based on the symmetric double-bridge model, the asymmetric double-bridge experiment has been developed in the Player/Stage, as shown in Fig. 7. There are ten robots in the left and two ways can be chose, the black area represent obstacles, the starting point of robots is the nest and the destination is the food source in the right. One way in the above named bridge A and the other is bridge B, obviously the length of bridge A is longer than bridge B. Taking path length and pheromone two factors into account, the probability of the m-th robot choose bridge A is

$$p_A(m) = \frac{(\frac{A_m + k}{L_A})^h}{(\frac{A_m + k}{L_A})^h + (\frac{B_m + k}{L_B})^h} \tag{1}$$

The probability of the m-th robot choose bridge B is $p_B(m) = 1 - p_A(m)$, L_A and L_B denote the lengthen of bridge A and bridge B. The values of parameter k, h are set to k=20, h=2 [24], and $L_A = 2L_B$.

The algorithm flow is as follows:

Step1: initialize the robot, the robot moves to the path bifurcation junction;

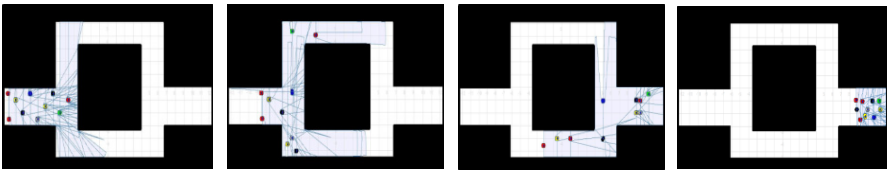
Step2: generate a random number t from 0 to 1;

Step3: calculate the probability that choosing the bridge A p(A), and compared with t, if t>p(A), then choose the bridge B, or choose the bridge A;

Step4: robot reaches the destination.

The result shown in Fig. 5, firstly the robot move from the start region to the path bifurcation junction, and then randomly selected path to move. Without the effect of

pheromone, the probability of robot choose bridge A is similar to the probability of robot choose bridge B, so the number of robot choose bridge A is equal to the number of robot choose the bridge B. When under the joint influence of the pheromone and the length of path, the robot will tend to choose the bridge B until there is no robot chooses the bridge A.



(a) The initial stage (b) The early stage (c) The medium stage (d) The late stage

Fig. 5. Asymmetric double-bridge experiment

It can be seen through the test: asymmetric and symmetric double-bridge experiments have the same mechanism, that is the expansion of the initial fluctuations, the ants will often choose the shortest path, which makes the short branch has more pheromone to induce the nest companions choose the short branch. In other words, through the expansion of the initial fluctuations, the probability of final choice a short branch is growing with the increase of the two branches length.

4 Conclusion

Through ten years of research, the related theory and practice studies of swarm robotics have make much progress, such as self-assembly [25], self-organization [26], task allocation [27] and other aspects already have some of the more in-depth studies. But there is still a big gap between the expectations of swarm robotics, always mainly restricted to Kinematics behavior analysis, Synchronization problems, Collaborative self-organizing, etc.

Acknowledgments. This work is jointly supported by NSFC under Grant No. 60903067, 61170117, Beijing Natural Science Foundation under Grant No. 4122049, Funding Project for Beijing Excellent Talents Training under Grant No. 2011D009006000004, and the Fundamental Research Funds for the Central Universities.

References

1. Xinsong, J.: Control Problem in Robot and Robotics. Robot 12, 1–13 (1990)
2. Arai, T., Pagello, E., Parker, L.: Advances in Multi-Robot Systems. IEEE T. Robotics Autom. 18, 655–661 (2002)

3. Kui, Y., Yuan, L., Li-Xin, F.: Multiple Mobile Robot Systems: A Survey of Recent Work. Acta Automatica Sinica 33, 785–795 (2007)
4. Dorigo, M., Tuci, E., Groß, R., Trianni, V., Labella, T.H., Nouyan, S., Ampatzis, C., Deneubourg, J.-L., Baldassarre, G., Nolfi, S., Mondada, F., Floreano, D., Gambardella, L.M.: The SWARM-BOTS Project. In: Şahin, E., Spears, W.M. (eds.) Swarm Robotics 2004. LNCS, vol. 3342, pp. 31–44. Springer, Heidelberg (2005)
5. Dorigo, M., Sahin, E.: Special Issue: Swarm Robotics. Auton. Robot. 17, 111–113 (2004)
6. Sahin, E., Winfield, A.: Special Issue on Swarm Robotics. Swarm Intelligence 2, 69–72 (2008)
7. Şahin, E.: Swarm Robotics: From Sources of Inspiration to Domains of Application. In: Şahin, E., Spears, W.M. (eds.) Swarm Robotics WS 2004. LNCS, vol. 3342, pp. 10–20. Springer, Heidelberg (2005)
8. Kennedy, J., Eberhart, R.C., Shi, Y.: Swarm Intelligence (photocopy edition). People Post Press, Beijing (2009)
9. Beni, G., Wang, J.: Swarm Intelligence. In: Proceedings of the Seventh Annual Meeting of the Robotics Society of Japan, pp. 425–428 (1989)
10. Colorni, A., Dorigo, M., Maniezzo, V., et al.: Distributed Optimization by Ant Colonies. In: Proceedings of the 1st European Conference on Artificial Life, pp. 134–142 (1991)
11. Min, T., Yong, F., Guohua, X.: Research on Control and Co-operation for Swarm robot Systems. Robot. 23, 178–182 (2001)
12. Minhui, W., Shugen, M., Bin, L., Chaoyue, W.: Configuration expression and remodeling optimization of independent operational and reconfigurable groups of robot. Science in China (Series F: Information Sciences) 39, 269–280 (2009)
13. Hongxing, W., Miao, L., Dezhong, L., Tianmiao, W.: A Novel Self-assembly Modular Swarm Robot: Docking Mechanism Design and Self-assembly Control. Robot. 32, 614–621 (2010)
14. Vaughan, R.: Massively Multi-robot Simulation in Stage. Swarm Intelligence, 189–208 (2008)
15. Zhiguo, S., Zhiliang, W., Jiwei, L.: Developments in heterogeneous multi-robot cooperation systems. CAAI T. Intelli. Systems 4, 377–391 (2009)
16. Desai, J.: A Graph Theoretic Approach for Modeling Mobile Robot Team Formations. Journal of Robotic System 19, 511–525 (2002)
17. Li, L., Martinoli, A., Abu-Mostafa, A.: Learning and Measuring Specialization in Collaborative Swarm Systems. Adapt. Behav. 12, 199–212 (2004)
18. Hall, D., Llinas, H.: An Introduction to Multisensor Data Fusion. Proceedings of the IEEE 85, 6–23 (1997)
19. Luo, R., Kay, M.: Multi-sensor Integration and Fusion in Intelligent Systems. IEEE T. Syst. Man Cy. B. 19, 901–931 (1989)
20. Yuan, L., Kui, Y., Rui, Z.: Performance Analysis of Multi-Robot Communication System Based on Wireless Local Area Networks. Journal of System Simulation 21, 2219–2223 (2009)
21. Xiaoping, R., Zixing, C., Aibing, C.: Current research in multi-mobile robots communication system. Control and Decision 25, 327–332 (2010)
22. Christensen, A., Grady, R., Dorigo, M.: SWARMORPH-script: a Language for Arbitrary Morphology Generation in Self-assembling Robots. Swarm Intelligence 2, 143–165 (2008)
23. Deneubourg, J.L., Goss, S., Pasteels, J.M.: The self-organizing exploratory pattern of the argentine ant. J. Insect Behav. 3, 159–168 (1990)

24. Pasteels, J.M., Deneubourg, J.L., Goss, S.: Self-organization mechanisms in ant societies: Trail recruitment to newly discovered food sources. Experientia Supplementum 54, 155–175 (1987)
25. Gross, R., Bonani, M., Mondada, F., Dorigo, M.: Autonomous Self-Assembly in Swarm-Bots. IEEE T. Robot. 22, 1115–1130 (2006)
26. Trianni, V., Nolfi, S.: Self-Organizing Sync in a Robotic Swarm: A Dynamical System View. IEEE T. Evolut. Comput. 13, 722–741 (2009)
27. Berman, S., Halasz, A., Hsieh, M., Kumar, V.: Optimized Stochastic Policies for Task Allocation in Swarms of Robots. IEEE T. Robot. 25, 927–937 (2009)

Levels of Realism for Cooperative Multi-agent Reinforcement Learning

Bryan Cunningham and Yong Cao

Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, VA 24060, USA
{bcunn06,yongcao}@vt.edu

Abstract. Training agents in a virtual crowd to achieve a task can be accomplished by allowing the agents to learn by trial-and-error and by sharing information with other agents. Since sharing enables agents to potentially reach optimal behavior more quickly, what type of sharing is best to use to achieve the quickest learning times? This paper categorizes sharing into three categories: realistic, unrealistic, and no sharing. Realistic sharing is defined as sharing that takes place amongst agents within close proximity and unrealistic sharing allows agents to share regardless of physical location. This paper demonstrates that all sharing methods converge to similar policies and that the differences between the methods are determined by analyzing the learning rates, communication frequencies, and total run times. Results show that the unrealistic-centralized sharing method – where agents update a common learning module – is the most effective of the sharing methods tested.

Keywords: cooperative learning, multi-agent reinforcement learning, crowd simulation, 2D virtual world, inter-agent communication.

1 Introduction

Single-agent reinforcement learning (RL) has been widely studied over the past few decades [4]. Its extension to multiple agents that share a common environment is called multi-agent reinforcement learning (MARL) [1]. In the recent years MARL has been studied and adapted to work in the crowd simulation domain [8,2]. The overarching goal of crowd simulation is to realistically emulate the outward behaviors of its constituents, or agents, for the purposes of replicating physical actions and their resultant effects in an environment. Its applications include architectural and urban planning, evacuation planning, and video game and movie domains. The reason for adapting MARL to work within the crowd simulation domain is simple: since people learn and consequently adapt to situations using a form of reinforcement learning, why not apply this technique to train simulated computer agents to learn how to behave in a crowd? Naturally, because the agents are surrounded by other agents, it is logical to assume that they will come into contact with one another during simulation. The agents

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 573–582, 2012.

could benefit from communicating what they have learned to others. This paper focuses on the concept of inter-agent sharing within the crowd simulation domain for an evacuation scenario. It seeks to understand the impact that various methods of sharing have on the effectiveness of the agents' learning while exploiting the benefits of using a layered MARL architecture. Effectiveness is defined by resultant navigational behavior and total training time. Ultimately, this research serves as a case study to explore the most efficient ways to scale up to larger crowds in larger environments. Before delving into the specifics of the research, this paper will briefly discuss further reasoning for the necessity of this study followed by the essential background material relevant to understanding MARL, inter-agent sharing, and layered MARL. This paper attempts to explore how the intersection of these three domains and their application to the crowd simulation domain results in a previously unstudied research void.

2 Related Work and Motivation

RL is good at capturing individuality, or diversity, in an agent because each agent learns based on its own experiences within an environment. These experiences shape the decisions made by an agent, causing them to appear as if they have their own unique personality as they navigate through the environment. In the real world, people not only learn from trial-and-error exploration but also from each other through observational and/or verbal communication.

For purposes of this paper, realistic sharing is defined as taking place amongst agents within close proximity to one another. Other literature on sharing methods within the RL domain follow a trend in which inter-agent sharing is done via sharing across all agents, independent of agent location [7,3,9]. These papers use unrealistic methods of sharing and claim to be able to train the agents using fewer numbers of episodes, which is indicative of faster learning rates. It is important to note that faster learning rates do not necessarily imply faster total run times for agent training. For example, one learning rate could be faster than another but have a much higher communication overhead associated with it. During training this communication overhead could prove to dominate the total simulation training time. This type of method would therefore prove to be much more ineffective, computationally-wise, than a method that has a slower learning rate but a smaller communication overhead. This raises questions about the differences in the resultant navigational behavior between learning with realistic, unrealistic, and independent (no sharing) methods. This paper also seeks to identify viable, potentially faster, methods for training agents in the crowd simulation domain.

Tan [7] includes discussions on the communicational overhead associated with unrealistic and independent sharing methods. However, these discussions are limited in that the author analyzed communication overhead from a theoretical perspective only and discussed learning rate separately from communication overhead. As the realistic sharing methods presented in this paper allow agents to share on an inconsistent basis, we cannot perform static, theoretical analysis.

Instead performance tests measure learning rate, actual communication overhead and total running time for each sharing method. Learning rate and communication overhead need to be analyzed in conjunction with one another in order to classify the effectiveness of a method more accurately. Total method run time sufficiently captures the effectiveness of a method.

3 Background Concepts

3.1 Multi-agent Reinforcement Learning

RL is a bottom-up programming methodology that imbues agents with the ability to generalize learned information and extract salient environmental cues online. RL relies on the concept of Markov decision processes (MDP) to model how an agent moves around in the environment. An MDP is a 4-tuple taking the form $(S, A, P^a_{ss'}, R^a_{ss'})$ where S is the state space, A is the action set, P is the transition function where $P^a_{ss'}$ represents the probability of transitioning from state s to state s' via action a, and R is the reward function where $R^a_{ss'}$ represents the expected value of the reward achieved when an agent moves from state s to state s' via action a. As an agent explores its environment, it updates its policy function π that maps each state $s \in S$ and action $a \in A(s)$ to $\pi(s, a)$ which represents the probability of taking action a in state s. Agents define an action-value function for policy π by $Q^\pi(s, a)$ which indicates the expected return given that the agent takes action a in state s and then applies policy π. The agent attempts to maximize the expected total sum of rewards gained over time to converge to an optimal policy π^* (of which there can be multiple). For this paper, we use a simple and popular form of RL called Q-learning: a type of temporal-difference (TD) learning. TD learning is a learning technique in which an agent will update its previously estimated state values using the differences between its current and former values. This effectively propagates more accurate estimates of the state values as learning continues. Q-learning represents an off-policy form of TD control which, for the one-step case used in this paper, takes the following form:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right] \quad . \tag{1}$$

where t is the time step parameter, α is the learning rate parameter, and γ is the discount rate parameter. In this paper, the single-agent form of RL is applied to each agent in a multi-agent environment. Essentially, this means that agent learning takes place simultaneously and agents treat each other as independent, dynamic forces acting within the same environment.

3.2 Layered MARL Architecture

By default, RL and its extension to MARL, use one learning module to capture the learned policy of an agent. In a layered MARL architecture, an agent will use multiple learning modules to capture the learned policy of an agent [5].

Takahashi et al. [6] state that decomposing the control structure into smaller chunks, or learning modules, allows for the module policies to be transportable and applicable to new situations. To understand this, think about how an agent views a state in their state space. States are composed of multiple state parameters – any environmental information that the programmer wants the agent to consider when learning needs to be encoded in a state variable. For instance, a state variable could represent a position on the grid described by Cartesian coordinates or a facing direction described by a cardinal direction. Learning modules can be designed in such a way as to split up and group states' parameters for the purposes of decomposing the problem into separate logical units. These logical units can then be used to train separate learning modules within an agent. The learning modules work together to decide an appropriate action for the agent to take in a given state. This is advantageous because by decomposing the state parameters into logical components, learned behaviors can become more generalized and less dependent on other state parameters that may have no correlation. This helps reduce the total state space necessary to navigate, increases learning efficiency, and allows for sharing at the module level where data is less coupled by design.

3.3 Inter-agent Sharing

People can learn based on consciously or subconsciously observing and/or communicating various types of information based on the situation. Tan and Ribiero et al. [7,3] show that inter-agent sharing of sensation, policies and/or episodes decreases the steps necessary to reach optimal or good convergence points when compared to agents that did not share. This paper explores inter-agent sharing methods and classifies each as either realistic, unrealistic, or independent. With regard to this paper, sharing between two agents is classified as realistic only when the agents are sharing within communication distance of one another. This communication distance represents the range at which an agent is able to physically see or talk to another agent in the environment. Within this range, sharing would imitate realistically how people learn based on observing others or by verbally communicating information they might have learned.

When sharing using MARL, agents share learned information – for instance, in the form of Q-Values – from their policies and incorporate new information being shared with them into their existing policies. Unrealistic sharing indicates the broad range of sharing methods that are not based on reality as we have defined it above. In this case, sharing may not necessarily take place when agents are within one another's communication fields. The sharing method described earlier in which agents share regardless of their location on the map is an example of this. Another example of unrealistic sharing that will be encountered in this paper is centralized sharing. Centralized sharing enables agents to share a single Q-Value table and collectively contribute to and use its learning knowledge to make policy decisions. Note that realistic sharing methods are always localized sharing methods, but localized sharing methods are not exclusively realistic sharing methods.

4 Problem Statement

This paper will address the following questions: How do the sharing methods affect an agent's resultant navigational behavior? How do the sharing methods affect the learning rates of convergence, the communication overhead, and the overall training time for an agent? Can the layered MARL architecture be used to enable agents to successfully navigate both static and dynamic obstacles in the environment, in addition to finding and arriving at a goal location?

5 Approach

5.1 Agents, Environment, and Task

To explore the problems presented, a small-scale environment in which a simple evacuation simulation will take place was created. Agents attempt to evacuate to a common location using the most optimal path while encountering obstacles along the way. Agents traverse a discrete environment consisting of 7 x 7 cells. At each agent's turn, or step, agents are able to move Up, Right, Down, Left, or Stay in the current spot. There are two types of obstacles in the environment: walls (static) and other agents (dynamic). No two agents may be in the same cell as another, so if an agent attempts to move into an occupied cell, it will stay where it is. Similarly, if an agent attempts to leave the map or move into a space blocked by a wall, it 'bounces' off the map edge or wall and stays in place. Agents attempt to navigate to the same goal location and therefore are homogeneously oriented.

Testing occurs on 3 maps that are designed not only to explore the effectiveness of the sharing methods but also to investigate whether layered MARL is a feasible architecture within the crowd simulation domain. To do this, the three maps test dynamic obstacle avoidance, static obstacle avoidance, and a combination of the two, respectively. Map 1 tests dynamic obstacle avoidance because it contains no walls and focuses on agents learning to reach the goal in cell (6,3) while avoiding the other agents. Map 2 tests static obstacle avoidance because it contains a randomized placement of walls and disables agent-agent collisions. Map 3 tests both static and dynamic obstacle avoidance by reusing the same randomized map from Map 2 but turns agent-agent collisions back on. For all three maps, agents start in an assigned location. Agents 1, 2, and 3 start in positions (0,1), (0,3), and (0,5), respectively.

5.2 Sharing Methods

The following descriptions provide detail regarding the four variations of sharing methods that will be used in testing: (1) *Independent (No Sharing)*: Agents do not share any information with one another. (2) *Realistic-Localized Sharing*: For realistic-localized sharing, agents share the Q-values in their policies with other agents when they are within one another's communication fields. An agent's communication field is determined by the communication field size, where the

size corresponds to the depth of cells directly surrounding the agent. A field size of 1, for instance, would indicate that all cells directly around the agent's cell are part of the communication field. A cell C in a communication field must have an unobstructed line-of-sight, free of walls, to the agent the field emanates from; otherwise the agent will not be able to communicate with another agent that may be in C at the time. For both the realistic-localized and unrealistic-localized sharing methods, sharing is performed by using the frequency of state-action visitation to determine which agent has the most experience with that particular state-action pair. When this is determined, the Q-value for that particular state-action pair is synchronized to this *best* Q-value across all agents who are participating in the share. This continues for each possible state-action pair for both of the learning modules every time a share event occurs. Variables that are adjusted for testing this method are communication field size (CFS) and sharing step size (SS), where SS represents the minimum frequency, in terms of steps, with which an agent is allowed to share with another agent. (3) *Unrealistic-Localized Sharing*: Similar to realistic-localized sharing, except that sharing takes place uniformly at defined step sizes by all agents at once. Only the SS variable will be adjusted for testing this method as CFS is not applicable. (4) *Unrealistic-Centralized Sharing*: Agents update a shared, central Q-Value table and no explicit sharing occurs.

5.3 Layered MARL Implementation Details

This paper uses a simple two-module layered architecture: the pathfinder (P) and collision-avoidance (CA) learning modules. The pathfinder learning module's purpose is to find a path from the initial starting position to the goal position. The pathfinder module's set of actions are A_{P}={Up, Right, Down, Left, Stay}. The module's set of states S_{P} represent each cell position (x,y) on the map and therefore $|S_{\mathrm{P}}| = n \times m$ where n is the number of cells in the vertical direction and m is the number of cells in the horizontal direction. The collision-avoidance learning module's purpose is to avoid colliding with obstacles in the environment. The collision-avoidance module's set of actions are A_{CA}={Up, Right, Down, Left, Stay}. The module's set of states S_{CA} are represented by each unique permutation of the 8 cells directly surrounding the agent where each cell can be either empty (= 0) or not empty (= 1) for a total of $2^8 = 256$ states. These two learning modules work together because the pathfinder module determines an action and passes it to the CA module. Based on both the surrounding obstacles and the action suggested by the pathfinder module, the CA module then determines an action to take and instructs the agent to take that action. The pathfinder's module defines rewards as follows: Q_{P}: $(S_{\mathrm{P}} \times A_{\mathrm{P}}) \rightarrow \mathbb{R}$ where all actions that lead to a non-goal state receive a reward of -0.005 and all actions that lead to a goal state receive a reward of 1.0. The CA's module defines rewards as follows: Q_{CA}: $(S_{\mathrm{CA}} \times A_{\mathrm{CA}}) \rightarrow \mathbb{R}$ where an action in agreement with the action A chosen by the pathfinder module receives a reward of 0.005. An action that, oriented with respect to A, points to the side (left or right) receives a reward of -0.005. An action that, oriented with respect to A, points backwards receives a reward of -0.1. An action that, when A was any action but stay, was stay

receives a reward of -0.005. Finally, an action that, when A was stay, was any of the other actions other than stay, receives a reward of -0.005.

5.4 Experimental Setup

In this paper, agents operate using a discrete MDP in an environment conducive to episodes. Each episode is defined with an initial starting state and spans until a terminal, or goal, state is reached for each agent. A series of episodes define a simulation run. 150 simulation runs were tested for each sharing method in order to generate dependable data averages. An agent's learned policy was carried over from one episode to the next within a run, so ideally this results in convergence to an optimal policy as the agent learns more about the state-action space. In order to equally test the sharing methods we used an ϵ-greedy exploitation-exploration method with $\epsilon = 0.05$. The simulation was run until each sharing method converged to the same number of steps per episode – meaning that a path convergence point had been reached. The CFS parameter for realistic-localized sharing will vary and take the values of 1, 2, 3, and 4 where 1 represents a very limited communication field and 4 represents a fairly wide communication field. The SS parameter for both realistic and unrealistic-localized sharing will vary and take the values of 1, 5, 10, and 15. A SS value of 1 corresponds to agents being able to share their policies after a minimum of every step and a SS value of 15 indicates that sharing will occur less often, after a minimum of every 15 steps. The values for the simulation are set as follows: $\alpha_{\mathrm{P}} = 0.9$, $\gamma_{\mathrm{P}} = 0.8$, $\alpha_{\mathrm{CA}} = 0.2$, $\gamma_{\mathrm{CA}} = 0.8$.

6 Results and Contributions

Results were gathered across the 3 maps for the 3 agents. The resulting trends associated with Maps 1 and 2 were the same as the trends associated with Map 3, therefore only the findings from Map 3 will be presented. Similarly, overall data trends amongst agents agreed and only the findings for agent 1 will be shown. Results from the path convergence testing measure learning rate (in average steps per run), communication frequency, and total run time (in seconds) for each sharing method. Communication frequency measures every time a Q-table is shared with another agent – for example, sharing both a pathfinder and a CA Q-table count as two communication units because two Q-tables are shared.

Table 1 provides a detailed table containing results from all variations of the sharing methods used. Recall that faster learning rates (indicated by smaller learning rate numbers) signify that an agent has reached convergence in a fewer number of episodes. As expected, the methods that share the most have the lowest learning rates. Figure 1 illustrates differences in learning rates across a run for the variations of the four unique sharing methods with the best learning rates. The figure shows that the unrealistic-centralized sharing method significantly outperforms the other methods. This is understandable as the frequency of sharing, uniformity of sharing, and quality of sharing all increase as we move

from independent sharing to unrealistic-centralized sharing. To determine the overall computational differences between the sharing methods, especially with the realistic-localized method, we had to consider both learning rate and communication frequency in conjunction, which most clearly translated to a total run time.

Table 1. Learning rate, communication frequency, and total run time for the sharing methods

Sharing Method	Learning Rate (avg. steps)	Communication Frequency	Total Run Time (s)
Independent	95.88	0.00	0.01
Realistic-Localized:			
CFS = 1, SS = 1	61.31	361.13	0.75
CFS = 1, SS = 5	62.68	135.93	0.30
CFS = 1, SS = 10	63.79	98.71	0.22
CFS = 1, SS = 15	65.36	83.31	0.18
CFS = 2, SS = 1	59.87	829.76	1.59
CFS = 2, SS = 5	60.80	259.55	0.54
CFS = 2, SS = 10	62.51	166.60	0.36
CFS = 2, SS = 15	63.95	134.47	0.29
CFS = 3, SS = 1	59.07	1060.96	1.86
CFS = 3, SS = 5	60.33	321.05	0.61
CFS = 3, SS = 10	61.58	202.08	0.38
CFS = 3, SS = 15	62.18	158.24	0.30
CFS = 4, SS = 1	57.03	1166.88	1.99
CFS = 4, SS = 5	58.32	364.44	0.69
CFS = 4, SS = 10	60.55	233.88	0.43
CFS = 4, SS = 15	61.40	177.40	0.33
Unrealistic-Localized:			
SS = 1	49.53	8641.49	7.06
SS = 5	50.96	1776.64	1.48
SS = 10	51.29	852.56	0.72
SS = 15	52.78	577.79	0.50
Unrealistic-Centralized	45.01	0.00	0.01

Performance timing for the total run times provides a general picture of how effective each method is with regard to one another. Figure 2 depicts the differences in time per episode across a run for the variations of the four sharing methods with the fastest total times. Both the independent and unrealistic-centralized sharing methods performed equally well as no sharing takes the same amount of time that implicit sharing does.

The realistic-localized and unrealistic-localized methods perform much more poorly, on the order of 19x and 51x more slowly, respectively. Overall, with respect to both learning rate and total run time, the unrealistic-centralized sharing method is the most effective method tested in this experiment. Realistic sharing

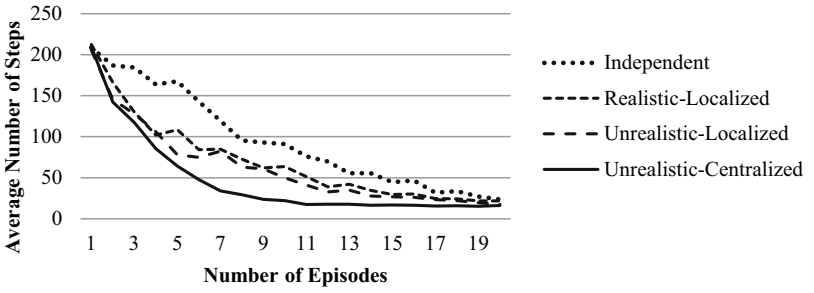


Fig. 1. Average number of steps vs. number of episodes for the sharing methods

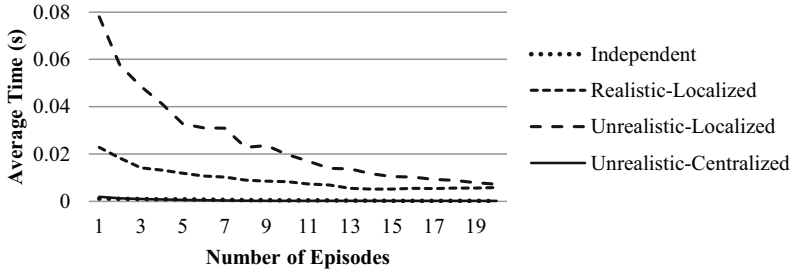


Fig. 2. Average time vs. number of episodes for the sharing methods

limits the rate at which learning can take place because sharing takes place less frequently and less uniformly. Localized sharing in general does not allow for policy information to be as readily absorbed as it is with centralized sharing – not to mention the explicit communication overhead present that overwhelms the training time for the agent. It is important to mention that while centralized sharing is the best choice for the purpose of efficiency, simulations that study communication flows in crowds, for example, must be done using realistic-localized methods. This type of method preserves communication fidelity and ensures accurate inter-agent communication patterns.

7 Conclusions and Future Work

This paper demonstrates that all sharing methods converge to the same policies. The differences between realistic, unrealistic, and independent sharing methods are determined by analyzing the learning rates, communication frequencies, and total run times. The unrealistic-centralized sharing method proved to be the most effective of the sharing methods tested. Finally, testing showed that agents successfully navigated both static and dynamic obstacles in the environment, in addition to finding and arriving at a goal location. This research opens up the possibility to study more detailed problems concerning applying a layered MARL architecture within the crowd simulation domain. For instance, how does

adding more learning modules to each agent affect the complexity of the state-action space? Additionally, how well do the sharing methods presented in this paper port to the GPU in order to support the simulation of more agents in a more finely discretized environment?

Acknowledgements. This work is partially funded by National Science Foundation, IIS 0940723, titled EAGER: Drummer Game: A Massive-Interactive Socially-Enabled Strategy Game".

References

1. Busoniu, L., Babuska, R., De Schutter, B.: A comprehensive survey of multiagent reinforcement learning. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 38(2), 156–172 (2008)
2. Martinez-Gil, F., Lozano, M., Fernández, F.: Multi-agent Reinforcement Learning for Simulating Pedestrian Navigation. In: Vrancx, P., Knudson, M., Grześ, M. (eds.) ALA 2011. LNCS, vol. 7113, pp. 54–69. Springer, Heidelberg (2012)
3. Ribeiro, R., Borges, A., Enembreck, F.: Interaction models for multiagent reinforcement learning. In: 2008 International Conference on Computational Intelligence for Modelling Control Automation, pp. 464–469 (2008)
4. Sutton, R., Barto, A.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
5. Takahashi, Y., Asada, M.: Multi-controller fusion in multi-layered reinforcement learning. In: International Conference on Multisensor Fusion and Integration for Intelligent Systems, pp. 7–12 (2001)
6. Takahashi, Y., Asada, M.: Behavior acquisition by multi-layered reinforcement learning. In: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, pp. 716–721 (1999)
7. Tan, M.: Multi-agent reinforcement learning: Independent vs. cooperative agents. In: Proceedings of the Tenth International Conference on Machine Learning, pp. 330–337. Morgan Kaufmann (1993)
8. Torrey, L.: Crowd simulation via multi-agent reinforcement learning. In: Youngblood, G.M., Bulitko, V. (eds.) Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment. The AAAI Press (2010)
9. Zhang, P., Ma, X., Pan, Z., Li, X., Xie, K.: Multi-Agent Cooperative Reinforcement Learning in 3D Virtual World. In: Tan, Y., Shi, Y., Tan, K.C. (eds.) ICSI 2010, Part I. LNCS, vol. 6145, pp. 731–739. Springer, Heidelberg (2010)

Research of Tourism Service System Base on Multi-Agent Negotiation

Youqun Shi, Cheng Tang, Henggao Wu, and Xinyu Liu

School of Computer Science and Technology
Donghua University, Shanghai, China
yqshi@dhu.edu.cn, tangcheng0407@126.com, whg121908@163.com

Abstract. In a multi-Agent system, agents should keep negotiation and cooperation to get a resolution when dealing with an issue. Consequently, negotiation becomes a key point to run the system successfully. This paper firstly proposed an Agent structure model suitable for this field according to the traits of tourism service system. According to the characteristics of the Agent negotiate mechanism, made some researches about multi-issue problems, utility function, negotiation protocol, negotiation strategy and so on during negotiation process, and then designed a negotiation model for tourism field, and used the test data to show that the availability of this model.

Keywords: Agent structure model, Negotiation, Negotiation model, MAS (multi-Agent system).

1 Introduction

Agent is an autonomous entity which has skills of making decisions independently, cooperating mutually and some intelligence. It has an independent expert knowledge and resources and can independently control its own actions. Just like Shoham[1] said that Agent is a software entity which is able to learn in a specific environment and run by itself, it usually works with other Agent to solve problems. When use agent to achieve some function of a system, a single agent is often unable to complete all needs. At this time, we must rely on a group of Agent work together to achieve their goals.

The field of tourism is a service network consisting of the scenery spots, transportation, hotel, shopping, etc. In today's highly competitive environment, whether the transportation or hotel and so on, visitors who have a lot of choice. Use Agent technology to design the appropriate Agent for each node of the field of tourism to represent their behaviors. Each Agent are autonomous and self-interested, their goal is to pursue their own best benefits. So whether the releaser or the receiver of task, they all hope that through the cooperation between agents to complete tasks and achieve a win-win situation in their respective interests. Negotiation is an effective way to solution the conflict and to achieve cooperation. Negotiation between the Agents is to reach a mutually beneficial agreement on certain issues.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 583–591, 2012.

2 The System Framework Based on the Multi-Agent Negotiation

MAS is to obtain a better overall performance by negotiation between agents. The MAS uses layered hierarchical structure, there is a central negotiation service Agent which is used to collect the information of this layer and the lower layer to undertake the unity decisions, manage multiple lower executions Agent, assign tasks and issue the command.

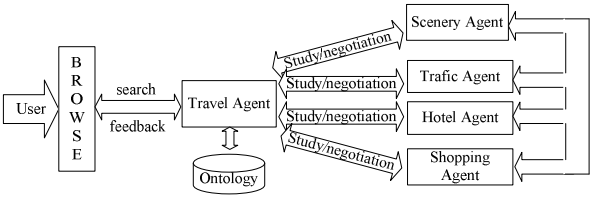


Fig. 1. Framework of multi-Agent intelligent travel information service system

As can be seen from the system framework shown in Figure 1, there is an intermediate Travel Agent, which acts as the role of the Service Agent, it is independent of the various resource-based Agents, and it has its own negotiation strategy and innovation mechanism. Travel Agent's work is mainly to decompose the task (tourists request), and will be distributed these tasks to the below Agent. When Travel Agent receives a new task, firstly, it will search result in the Ontology base, if it can find results then directly return the results to tourists. Otherwise, it will request other Agents to negotiate and cooperate to complete the task by the way of broadcast. In this framework, the Service Agent will be assigned service to the Agent of the corresponding sub-areas. Each agent maintains its own knowledge base, and through it to do reasoning and decision-making. Because of the interdependent relationship between Agents, when an agent encountered can't solve tasks, it will initiative take the way of communication to consult other relevant Agents and will feedback the negotiation results to Travel Agent. Travel Agent will push the learned new knowledge into the Ontology. Once again encountered a similar request, Travel Agent will directly return results to tourists, so as to provide tourists with faster, more accurately and more personal service.

3 Agent Structure Model

In MAS, especially in large complex systems, the first wanted to be solved problem is that the Agent uses what kind of structure is more conductively to communicate and cooperate between agents. Bratman[2] proposed BDI (Belief-Desire-Intention) theory which is recognized as one of the basic theories of DAI. Lots of researchers put forward their own views about Agent-structure model. Among them, Cohen [3], Levesque [3] and Weyns [4], etc. have made outstanding contributions in this area. Combined with previous theoretical perspective and this system characteristic, this

paper mainly studied the mixed-Agent[6] and presented an Agent structure model which is more suitable for the application of this system, as shown in figure 2.

The agent structure model is consisted of the perception modules, strategy modules, decomposition module, execution module and communication modules. The perception module is to receives information from the outside world, and to change perceive content into Agent's own faith; The Strategy module by querying the knowledge base and choosing the appropriate negotiation strategy to better address the negotiation issues; The decomposition module is decomposed the initial target into sub-goals by querying the plan library, and then each sub-goal is assigned to the corresponding Agent; The execution module is to describe the Travel Agent who wants to achieve specific goals (sub-goals) that actions need to be taken; Communication module can be divided into internal communication and external communications, the communication between Agents are through the message transfer schemes of communication model, while we use shared memory (i.e. blackboard structure) to exchange information between the execution modules within Travel Agent.

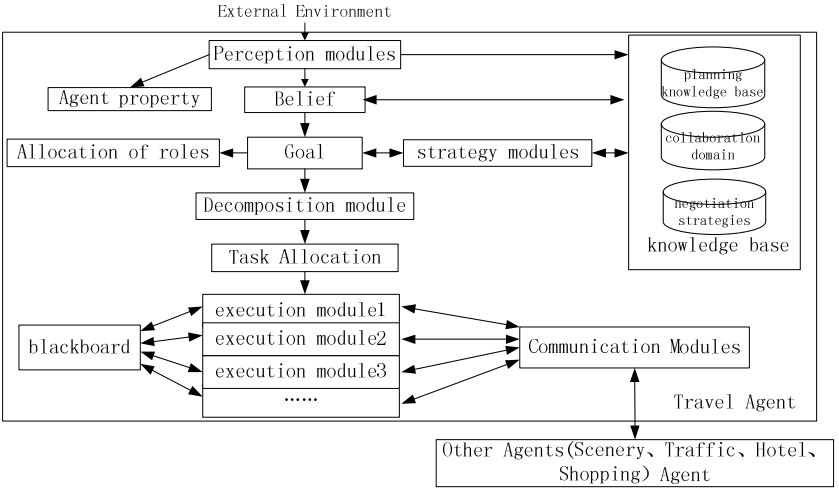


Fig. 2. Agent structure model for the field of tourism

4 Negotiation Model MMN(Multi-Agent Multi-issue Negotiation)

In many cases, Agent needs to interact with other agents constantly and work together to accomplish a task ultimately. The domestic and foreign researchers have proposed some models, at present, the main negotiation model with contract net model (CNM), "blackboard" model and multiple services Agent planning model. The agent in these models will automatically achieve cooperative to solving large complex problems. In order to definite the negotiation model more formally, it firstly need to understand the negotiation process.

4.1 Negotiation Process

A negotiation involves two kinds of Agents: the proposer and the participator, the proposer is a negotiation launched Agent, while the participator is a negotiation involved Agent. T represents the negotiation period, and t represents the negotiation time. The negotiation process can be shown in Figure 3.

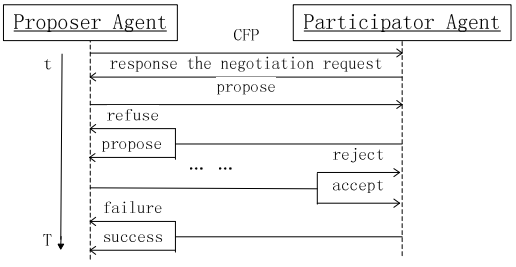


Fig. 3. Negotiation process

(1) Proposer Agent issues the negotiation request to other agent by broadcasting, and sends the CFP (call for proposal) message.
(2) The participator Agent decided to response the negotiation request after analysis.
(3) Both sides negotiate Agents begin to prepare the negotiation work, and t is set 0.
(4) In each round of negotiation, the proposer Agent issued a proposal to the participator Agent.
(5) The participator Agent receives the proposer's proposal, and uses its own utility function to evaluate the proposal.
(6) After evaluation by the utility function, if participator Agent agrees with other's proposal then send the agree message and generate results, turn (8). Otherwise, the participator Agent need to put forward the reverse proposal and send the propose message, turn (7).
(7) The Agent who receive the propose message need to make a decision whether should make concession for the next round of negotiation. If this time $t < T$ turn (4), or turn (9).
(8) Reach an agreement and negotiation is ended.
(9) Negotiation fails, and negotiation is ended.

By the negotiation process, it is known that the interaction process of negotiation may be reactive, also may be repeated interaction. In order to limit the unrestricted negotiation, used time T to limit the time of negotiation.

4.2 Negotiation Model

Defined: Negotiation Model ::= <*Ag, D, K, T, X, V, P, S*>, of which:

(1) *Ag*: the set of agents involved in negotiation, $Ag=\{A_1, A_2, \ldots\}$. Each negotiation involves two kinds of Agent: proposer and participator.

(2) D: the set of negotiation issues, $D=\{D_1, D_2, \ldots\}$, n is the number of issues has been involved in negotiation.

(3) K: the tendency value of negotiation issues, which is reflecting the importance degree of the issue. $K=\{K_i^d \mid i\in Ag,\ d\in D\}$, K_i^d is the Agent A_i's tendency value of the issue d, which reflects the preference degree of Agent A_i on the issue d, the greater the higher the preferences degree. In order to make the negotiation unified management and compared with each other, we can make Agent A_i on all issues tendency value, and the sum is 1. That is $\sum_{d=1}^{n} K_i^d = 1$ (n is the number of the issue).

(4) T: the order of the natural number system clock, $T=\{t_1, t_2, t_3\ldots\}$, the both sides agreed in accordance with the serial execution.

(5) X: the set of proposed value in negotiation. $X=\{x_{m\to n}^{t_i} \mid t_i\in \mathrm{T},\ m\in \mathrm{Ag},\ n\in \mathrm{Ag}\}$, m, n represents the both sides involved in negotiation. $x_{m\to n}^{t_i}$ shows that Agent A_n receive a proposed value which is send from Agent A_m at time t_i.

(6) V: the utility functions on a single issue or multiple issues. Agent will use this function to make an overall assessment with the all proposed value in the negotiation. The utility evaluation mechanism is mainly used to evaluate each other's proposed value and give a support to its feedback.

(7) P: the negotiation protocol. The both sides in negotiation must act the negotiation protocol as guidelines.

(8) S: the negotiation strategy. Negotiation strategy is the main factor to affect whether the negotiation is successful or not. This paper uses time-based thrift negotiation strategy in this negotiation model. In the early negotiation, the proposed value with both sides will changes slowly, and in later period, in order to get an outcome of the negotiations rather than ineffective negotiation, it has taken rapid change. So that the both sides of negotiation can reach an agreement within a specified period of time. Also set a different tendency values to each issue. We use the symbol K_i^d to represent it.

4.2.1 Utility Function

Negotiation mainly refers to the multi-issue negotiation. The both sides of negotiation often need to make a favorable choice in numerous choices. In order to get the maximum utility, the both sides of negotiation need to do an overall evaluation with the proposal. In the study of single-issue negotiation, the utility function is quite simple. P.Faratin[5] proposed the following calculation method.

Formula (1) is a monotonically increasing function, V_i represents the utility value for the issue D_i, $x^{t_i}{}_{b\to a}$ represents the currently proposed value of the issue d, x_{min} and x_{max} respectively represents the issue's minimum and maximum proposed value. When the value of $x^{t_i}{}_{b\to a}$ increasing, the value of V_i will increase too. So, the function can be used to assess the expected proposed value of the participator (hotel).

$$V_i = \frac{x^{t_i}{}_{b\to a} - x_{\min}}{x_{\max} - x^{t_i}{}_{b\to a}} \tag{1}$$

Similarly, we can give a utility function to the proposer agent, such as formula (2), this formula is a monotonically decreasing function, and this function can be used to evaluate the expected proposed value of the proposer (tourists).

$$V_i = \frac{x_{\max} - x^{t_i}_{b \to a}}{x_{\max} - x_{\min}} \tag{2}$$

Said above, the function is the single-issue negotiation utility function. When the negotiations with many issues, we can take advantage of the knowledge of probability and statistics to solve the utility value *V*. The problem of multiple issues can be transformed into a single-issue problem.

$$V = \sum_{d=1}^{n} K_i^d V_d \tag{3}$$

Above this, K_i^d is the Agent A_i's tendency value of the issue *d*, V_d represents the utility value for the issue d, *V* represents the total utility value of agent A on all issues. The large the value of *V*, the proposer Agent is more close to its expectations. The utility value of the utility function is used to determine the next proposal is accepted or rejected.

4.2.2 Negotiation Protocol

Negotiation protocol is a group of interact protocol sets that the negotiation Agent need to fellow in the negotiation process. Any Agent in negotiation will be bound to it. This set of rules includes all possible negotiation participants, negotiation action (for example, proposal, reverse proposal, negotiation termination) and negotiation event handling. Negotiation protocol is shown in figure 4.

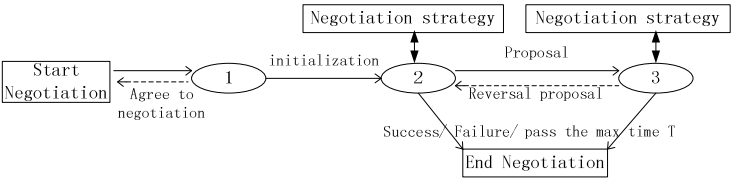


Fig. 4. Negotiation protocol

4.2.3 Negotiation Judgment

When the other's proposal come, the receive Agent need use the formula (4) to determine the received proposed value is accept or reject.

$$P(t_{i+1}, x^{t_i}_{m \to n}) = \begin{cases} reject(t_{i+1} > T) \\ accept(V(x^{t_i}_{m \to n}) >= V(x^{t_{i+1}}_{n \to m})) \\ x^{t_{i+1}}_{n \to m}(\text{revral proposal}) \end{cases} \tag{4}$$

The tendency value of each issue (K_i^d) must be initialized by the user. According to the negotiation strategy, the proposed value of single issue can use the following formula to calculate.

$$\left(x_{m\to n}^{t_i}\right)_d = \left(x_{b\min}\right)_d + \left(\left(x_{b\max}\right)_d - \left(x_{b\min}\right)_d\right)\times\left(\frac{t_i}{T}\right)^n \quad n>=2 \tag{5}$$

$$\left(x_{n\to m}^{t_i}\right)_d = \left(x_{s\max}\right)_d - \left(\left(x_{s\max}\right)_d - \left(x_{s\min}\right)_d\right)\times\left(\frac{t_i}{T}\right)^n \quad n>=2 \tag{6}$$

Above this, $\left(x_{m\to n}^{t_i}\right)_d$, $\left(x_{n\to m}^{t_i}\right)_d$ represents the currently proposed value of the issue d respectively offer by tourists and hotels at time t_i , $(x_{bmin})_d$, $(x_{bmax})_d$, $(x_{smin})_d$, $(x_{smax})_d$ respectively represents the minimum and maximum proposed value of issue *d*, T represents the negotiation period.

5 System Instance

Negotiation in tourism system is mainly used to implement two functions of the line query and hotel checking. The line query is mainly focuses on time (first element) and price (second elements), while the hotel checking mainly considers about the hotel price and hotel quality (overall quality). The system makes Travel Agent by registering after acquiring the data from visitors. Travel Agent will decompose the task into multiple sub-goals by the Agent internal planning, such as scenery information, transportation lines, hotels, etc. Those sub-goals will communicate with other Agent and return the negotiation results to Travel Agent.

It's assumed that Agent A is a Travel Agent. Now, Agent A requirements includes the line query and the hotel checking. There are also some other service Agents including Hotel Agent B, Hotel Agent C, Line Agent E, Line Agent F. Those Agents will communicate with Agent A. The following instance is to show the negotiation with the hotel proposal. The Agent's private information is shown in figure 5 and figure 6.

Agent (Ag)	Xmin ($)	Xmax ($)	T(deadline)	K(tendency value)	Grade (*)
A	60	150	3	2	
B	80	170	3	2	2
C	90	160	3	2	3

Fig. 5. The private information with Agent A, B, C

Grade	Low price ($)	High price($)
1	60	90
2	90	116
3	116	150
4	150	250
…	…	…

Fig. 6. Agent A can accept the hotel price with different quality

Considering an easy discussion, we think that the negotiation contributes to visitors. That is, visitors are the target, and they can decide when to negotiate. We set 2 as a negotiation coefficient for an easy calculating. Agent A thinks 80% about the price of the hotel and 20% about the quality. Agent B stands for a hotel whose quality is 2-star. Agent C stands for 3-star. Figure 7 is the process of Negotiation and bid.

From the negotiation strategy, Agent B and Agent C keep decreasing their proposed values according to formula 6, while Agent A keeps increasing its proposed value according to formula 5. When Agent A received the proposed values of Agent B and C, firstly it will calculate its utility value through formula 2, and then judge whether to reach the agreement through formula 4. If not, Agent A will propose a reversal proposal instead, and then Agent B and C should judge whether this proposed value succeed according to formula 1 and 4. This negotiation process is required to cycling before reaching an agreement. Agent A, B and C will finally get a conclusion in the limited time. Obviously, neither side is profitable until the two almost have the same proposed value. If Agent A has two tentative agreements separately with Agent B and C, it will find the corresponding maximum and minimum according to the hotel level through figure 6. If the proposed value is bigger than the maximum, then rejects, if not, then accept it a priority. In this case, after calculating, Agent A finds it can accept the proposed values in the preliminary agreement. Then it will estimate the two issues - price and quality - separately through the utility function of formula 2. Finally it will get the corresponding overall utility value from the proposed values given by Agent B and C according to the two issue's tendency value and formula 3. Agent A get the total utility value from Agent C is: $80\% * \frac{150-116.25}{150-60} + 20\% * \frac{150-116.25}{150-116} \approx 0.50$, and Agent A get the total utility value from Agent B is: $80\% * \frac{150-115}{150-60} + 20\% * \frac{116-115}{116-90} \approx 0.32$. It's clear that Agent A will reach an agreement with Agent C for getting a higher utility value from Agent C's total utility value. Agent A chooses a higher level hotel by paying a little more. This seems more reasonable for user's behavior habits.

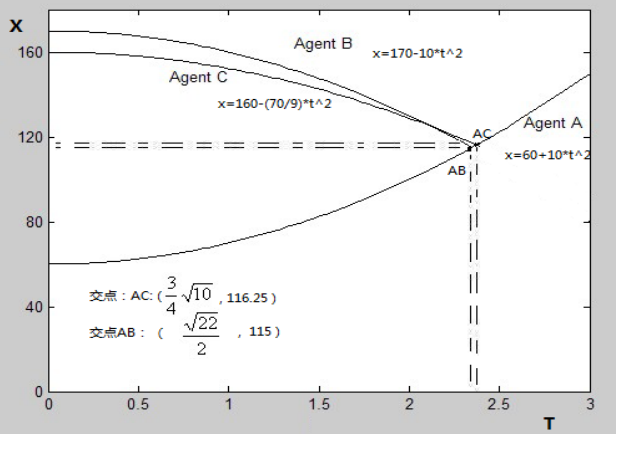


Fig. 7. The process of negotiation and bid

6 Conclusion

MAS is composed by multiple independent Agent, which relies on each other or conflict with each other and it is an important research field in DAI (distributed

artificial intelligence). Negotiation is the key to accomplish the task in MAS. The main work of the paper:

(1) Have deeply researched the MAS technology and Agent negotiation techniques, and introduced the ontology mechanism into Agent system on the basis of the above research.

(2) Constructed an Agent structure model for the tourism service system.

(3) Established a multi-agent multi-issue negotiation model by a detailed research with the negotiation process and negotiation strategies.

(4) Use the JADE platform to implement the tourism service system which is based on Multi-agent and Ontology. According to visitor's request, the system will automatically to select suitable tourism tour route and suitable hotel by negotiation between agents. The experiments prove that the negotiation between agents is able to solve a complex problem.

References

1. Shoham, Y.: Agent oriented programming. Artificial Intelligence 60(1), 51–92 (1993)
2. O'Hare, G.M.P., Jennings, N.R. (eds.): Foundations of distributed artificial intelligence, pp. 505–526. John Wiley & Sons, Inc. (1996)
3. Cohen, P.R., Levesque, H.J.: Intention is choice with commitment. Artificial Intelligence 42(3), 213–261 (1990)
4. Uhrmacher, A.M., Weyns, D.: Multi-Agent Systems: Simulation and Applications. CRC Press, New York (2009)
5. Jennings, N.R., Faratin, P., Lomuscio, A.R., Parsons, S., Sierra, C., Wooldridge, M.: Automated Negotiation: Prospects, Methods and Challenges. Int. J. Group 10(2), 199–215 (2001)
6. Wooldridge, M., Jennings, N.R.: Agent Theories, Architectures, and Languages: A Survey. In: Wooldridge, M.J., Jennings, N.R. (eds.) ECAI 1994. LNCS (LNAI), vol. 890, pp. 1–39. Springer, Heidelberg (1995)
7. Wang, L.-C., Chen, S.-F.: A Multi-Agent Multi-Issue Negotiation Model. Journal of Software 13(08), 1637–1643 (2002) ISSN: 10009825
8. Choi, S.P.M., Liu, J., Chan, S.-P.: A genetic agent-based negotiation system. Computer Networks 37, 195–204 (2001)
9. Faratin, P., Sierra, C., Jennings, N.R.: Negotiation decision functions for autonomous agents. Robotics and Autonomous Systems 24(3-4), 159–182 (1998)
10. Kim, H.S., Cho, J.H.: Supply chain formation using agent negotiation. Decision Support Systems 49, 77–90 (2010)
11. Ren, Z., Anumba, C.J., Ugwu, O.O.: The development of a multi-agent system for construction claims negotiation. Advances in Engineering Software 34(11-12), 683–696 (2003)

Distributed Model Predictive Control of the Multi-agent Systems with Communication Distance Constraints

Shanbi Wei*, Yi Chai, Hongpeng Yin, and Penghua Li

College of Automation, Chongqing University, Chongqing, 400044, P.R. China
wsbmei@yahoo.com.cn

Abstract. This paper addresses a distributed model predictive control (DMPC) scheme for multi-agent systems with communication distance constraints. Firstly, the communication distance constraints are dealt as non-coupling constraints by using the time varying compatibility constraints and the assumed state trajectory. Obviously, the control performance for all system is influenced by the time-varying compatibility constraints. Secondly, the deviation punishment is involved in the local cost function of each agent to penalize the deviation of the computed state trajectory from the assumed state. The value of the time-varying compatibility constraints is set according to the deviation of previous sample time. The closed-loop stability is guaranteed with a large weight for deviation punishment. A numerical example is given to illustrate the effectiveness of the proposed scheme.

Keywords: distributed control, model predictive control, time-varying compatibility constraint.

1 Introduction

Due to the computational advantages and the convenience of communication, distributed MPC(DMPC) is recognized as a nature technique to address trajectory optimization problems for multi-agent systems.

One of the challenges for distributed control is to ensure that local control actions keep consistent with the actions of others agents [1] and assure coupling constraints such as communication distance constraints. [2] proposes a distributed MPC with a fixed compatibility constraint to restrict the deviation. When the bound of this constraint is sufficiently small, the closed-loop system state enter a neighborhood of the objective state. [3] and [7] give an improvement over [2] by adding *deviation punishment term* to penalize the deviation of the computed state trajectory from the assumed state trajectory. Closed-loop exponential stability follows if the weight on the deviation function term is large enough.

* Corresponding author.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 592–601, 2012.

A contribution in this paper is to propose an idea to deal the communication distance constraint by time-varying compatibility constraint and deviation punishment term. At each sample time, the value of compatibility constraint is set as the maximum value of the deviation of the previous sample time. We give the stability condition to guarantee the exponential stability of the global closed-loop system, which is obtained by dividing the centralize stability constraint as the manner of [6]. The effectiveness of the scheme is also demonstrated by a numerical example.

Notations. x_k^i is the value of vector x^i at time k. $x_{k,t}^i$ is the value of vector x^i at a future time $k+t$, predicted at time k. $|x| = [|x_1|, |x_2|, ..., |x_N|]$ is the absolute value for each component of x. For a vector x and positive-definite matrix Q, $\|x\|_Q^2 = x^T Q x$.

2 Problem Statement

Consider a system which is composed of N_v agents. The dynamics of agent i is

$$x_{k+1}^i = f^i(x_k^i, u_k^i), \tag{1}$$

where $u_k^i \in \mathbb{R}^2$, $x_k^i \in \mathbb{R}^4$ and $f^i : \mathbb{R}^4 \times \mathbb{R}^2 \mapsto \mathbb{R}^4$ are the input, state and state transition function of agent i, respectively. $x_k^i = [q_k^i, v_k^i]$. $q_k^i = [q_k^{i,x}, q_k^{i,y}]^{\mathrm{T}}$, $q_k^{i,x}$ and $q_k^{i,y}$ are positions in the horizontal and vertical directions, respectively. $v_k^i = [v_k^{i,x}, v_k^{i,y}]^{\mathrm{T}}$, $v_k^{i,x}$ and $v_k^{i,y}$ are velocities in the horizontal and vertical directions, respectively. The sets of feasible input and state of agent i are denoted as $\mathcal{U}^i \subset \mathbb{R}^2$ and $\mathcal{X}^i \subset \mathbb{R}^4$, respectively, i.e.,

$$u_k^i \in \mathcal{U}^i,\ x_k^i \in \mathcal{X}^i,\ k \geq 0. \tag{2}$$

The communication region of agent i is a disc $\|q - q^i\| \leq \bar{r}_i$, $\bar{r}_i$ is the communication radius of agent i. At each time k, the control objective is to minimize

$$J_k = \sum_{t=0}^{\infty} \left[\parallel x_{k,t} \parallel_Q^2 + \parallel u_{k,t} \parallel_R^2 \right] \tag{3}$$

with respect to $u_{k,t}$, $t \geq 0$, where $x = [(x^1)^{\mathrm{T}}, \cdots, (x^{N_v})^{\mathrm{T}}]^{\mathrm{T}}$, $u = [(u^1)^{\mathrm{T}}, \cdots, (u^{N_v})^{\mathrm{T}}]^{\mathrm{T}}$; $x_{k,t+1}^i = f^i(x_{k,t}^i, u_{k,t}^i)$, $x_{k,0}^i = x_k^i$; $Q = Q^{\mathrm{T}} > 0$, $R = R^{\mathrm{T}} > 0$. $u \in \mathbb{R}^m$, $m = 2 * N_v$ and $x \in \mathbb{R}^n$, $n = 4 * N_v$. Then,

$$x_{k+1} = f(x_k, u_k), \tag{4}$$

where $f = [f^1, f^2, \cdots, f^{N_v}]^{\mathrm{T}}$, $f : \mathbb{R}^n \times \mathbb{R}^m \mapsto \mathbb{R}^n$. (x_e^i, u_e^i) is the equilibrium point of agent i, and (x_e, u_e) is the corresponding equilibrium point of all agents. $\mathcal{X} = \mathcal{X}^1 \times \mathcal{X}^2 \times \cdots \times \mathcal{X}^{N_v}$. $\mathcal{U} = \mathcal{U}^1 \times \mathcal{U}^2 \times \cdots \times \mathcal{U}^{N_v}$. The models for all agents are completely decoupled. The coupling between agents arises due to the situation that they operate in the same environment, and that the "cooperative" objective

is imposed on each agent by the cost function. Hence, there are the coupling cost function and coupling constraints such as commutation distance constraints.

The *control objective* for all system is to cooperatively asymptotically stabilize all agents to an equilibrium point (x_e, u_e) of equation (4) and assure the commutation distance constraints. In this paper we assumed that the $(x_e, u_e) = (0, 0)$, $f(x_e, u_e) = 0$. The corresponding equilibrium point for each agent is $(x_e^i, u_e^i) = (0, 0)$, $f^i(x_e^i, u_e^i) = 0$. Assumption $f^i(0, 0) = 0$ is not restrictive, since if $(x_e^i, u_e^i) \neq (0, 0)$, one can always shift the origin of the system to it.

The resultant control law for minimization of (3) can be implemented in a centralized way with high communication cost. Hence, by means of decomposition, J_k is divided as J_k^i's such that the minimization of (3) is implemented in distributed manner, with

$$J_k^i = \sum_{t=0}^{\infty} \left[\parallel z_{k,t}^i \parallel_{\bar{Q}_i}^2 + \parallel u_{k,t}^i \parallel_{\bar{R}_i}^2 \right], \; J_k = \sum_{i=1}^{N_v} J_k^i, \tag{5}$$

where $z_{k,t}^i = [(x_{k,t}^i)^{\mathrm{T}} \; (x_{k,t}^{-i})^{\mathrm{T}}]^{\mathrm{T}}$; $x_{k,t}^{-i}$ includes the states of the neighbors'. The set of neighbors' of agent i is denoted as $\mathcal{N}_i$. $x_k^{-i} = \{x_k^j | j \in \mathcal{N}_i\}$, $x_k^{-i} \in \mathbb{R}^{n^{-i}}$, $n^{-i} = \sum_{j \in \mathcal{N}_i} 4$. For each agent i, the control objective is to stabilize it to the equilibrium point (x_e^i, u_e^i). $\bar{Q}_i = \bar{Q}_i^{\mathrm{T}} > 0$, $\bar{R}_i = \bar{R}_i^{\mathrm{T}} > 0$. $\bar{Q}_i$ is obtained by dividing Q using the technique of [6]. For the agents have decoupled dynamics, the couplings of control moves for all system are not considered. R is a diagonal matrix and $\bar{R}_i$ is directly obtained. Under the networked environment, it is thus appropriate to allow agents to exchange information only once in each sampling interval [6].

In the receding horizon control manner, a finite-horizon cost function is exploited to approximate J_k^i. According to the (5), the evolution of the control moves with predictive horizon for agent i is based on the estimation of the state trajectories $x_{k,t}^{-i}$, $t \leq N$ of the neighbors', which are substituted by the assumed state trajectories $\hat{x}_{k,t}^{-i}$, $t \leq N$ as [6]. The connectivity of the inter-agent communication network is guaranteed by the communication distance constraints where agent i have to ensure the neighbor agent j ($j \in \mathcal{N}_i$) in its communication region.

$$\| q_{k,t}^j - q_{k,t}^i \| \leq \bar{r}_i, \; \forall t \leq N. \tag{6}$$

$\bar{r}_i$ is the communication radius of agent i. Obviously, if $q_{k,t}^j$ is substituted by $q_{k,t}^j$ within the (6), the connectivity of communication is not assured. Hence, (6) is modified as non-coupling constraint (26) by using the assumed state trajectory and compatible constraint.

Define

$$u_{k,t}^i = F_i(k) x_{k,t}^i, \; \forall t \geq N. \tag{7}$$

$F_i(k)$ is the gain of distributed state feedback controller.

Consider

$$\check{J}_k^i = \sum_{t=0}^{N-1} [\ \| \hat{z}_{k,t}^i \|_{\bar{Q}_i}^2 + \| u_{k,t}^i \|_{\bar{R}_i}^2 + \| x_{k,t}^i - \hat{x}_{k,t}^i \|_{\bar{T}_i}^2\ ] + \sum_{t=N}^{\infty} [\ \| x_{k,t}^i \|_{Q_i}^2 + \| u_{k,t}^i \|_{R_i}^2\ ]. \tag{8}$$

Where $\hat{z}_{k,t}^i = [(x_{k,t}^i)^{\mathrm{T}}\ (\hat{x}_{k,t}^{-i})^{\mathrm{T}}]^{\mathrm{T}}$, $\hat{x}_{k,0}^i = x_k^i$. $\hat{x}^{-i}$ includes the assumed states of the neighbors. $Q_i = Q_i^{\mathrm{T}} > 0$ and $R_i = R_i^{\mathrm{T}} = \bar{R}_i$ satisfy

$$diag\{Q_1, Q_2, \ldots, Q_{N_v}\} \geq Q,\ diag\{R_1, R_2, \ldots, R_{N_v}\} = R. \tag{9}$$

Obviously, Q_i is designed to stabilize the agent i to the local equilibrium point, independently. $\bar{Q}_i$ is designed to stabilize the agent i to the local equilibrium point with neighbor agents, cooperatively. $\bar{T}_i$ is the weight on the deviation punishment term.

At each time k, the optimization problem for distributed MPC is transformed as :

$$\min_{\bar{U}_k^i, F_i(k)}\ \check{J}_k^i,\ s.t.(1), (2), (6), (7), (8). \tag{10}$$

$\bar{U}_k^{*i} = [(u_{k,0}^{*i})^{\mathrm{T}}, (u_{k,1}^{*i})^{\mathrm{T}}, \ldots, (u_{k,N-1}^{*i})^{\mathrm{T}}]^{\mathrm{T}}$, only $u_k^{*i} = u_{k,0}^{*i}$ is implemented, and the problem (10) is solved again at time $k+1$.

3 Feasibility and Stability of Distributed MPC

The stability of distributed MPC by simply applying the procedure as in the centralized MPC will be discussed. The compact and convex terminal set Ω^i is defined by

$$\Omega^i = \{x^i \in \mathbb{R}^4 | (x^i)^{\mathrm{T}} P_i x^i \leq \alpha^i\} \tag{11}$$

where $P_i > 0$, $\alpha^i > 0$ are specified such that Ω^i is a control invariant set. So using the idea of [4] and [5], one simultaneously determines a linear feedback such that Ω^i is a positively invariant under this feedback.

Define the local linearization at the equilibrium point

$$A_i = \frac{\partial f^i}{\partial x^i}(0,0),\ B_i = \frac{\partial f^i}{\partial u^i}(0,0) \tag{12}$$

and assume that (A_i, B_i) is stabilizable. When $x_{k,N+t}^i$, $t \geq 0$ enters into the terminal set Ω^i, the local linear feedback control law is assumed as $u_{k,N+t}^i = F_i(k) x_{k,N+t}^i = K_i x_{k,N+t}^i$. K_i is a constant which is calculated off line as follows.

3.1 Design of the Local Control Law

The following equation follows for achieving closed-loop stability.

$$\|x_{k,N+t+1}^i\|_{P_i}^2 - \|x_{k,N+t}^i\|_{P_i}^2 \leq -\|x_{k,N+t}^i\|_{Q_i}^2 - \|u_{k,N+t}^i\|_{R_i}^2, t \geq 0. \tag{13}$$

Lemma 1. *Suppose there exist* $Q_i > 0$, $R_i > 0$, $P_i > 0$, *which satisfy the Lyapunov-equation*

$$(A_i + B_i K_i)^{\mathrm{T}} P_i (A_i + B_i K_i) - P_i = -\kappa_i P_i - Q_i - K_i^{\mathrm{T}} R_i K_i \tag{14}$$

for some $\kappa_i > 0$. *Then, there exists a constant* $\alpha^i > 0$ *such that* Ω_i *defined in* (11), *satisfies* (13).

Remark 1 . **Lemma 1** is directly obtained by referring to "*Lemma* 1" in [5]. For MPC, the stability margin can be adjusted by turning the value of κ_i according to **Lemma 1**. With regard to DMPC, [2] adjusts the stability margin by tuning the weight in the local cost function.

The control objective is to asymptotically stabilize the closed-loop system, so that $x^i_{k,\infty} = 0$ and $u^i_{k,\infty} = 0$. For $t = 0, \ldots, \infty$, it is easy to obtain

$$\sum_{t=N}^{\infty} \left[\|x^i_{k,t}\|^2_{Q_i} + \|u^i_{k,t}\|^2_{R_i} \right] \leq \|x^i_{k,N}\|^2_{P_i}. \tag{15}$$

Considering both (8) and (15), yields

$$\check{J}^i_k \leq \bar{J}^i_k = \sum_{t=0}^{N-1} \left[\|\hat{z}^i_{k,t}\|^2_{\bar{Q}_i} + \|u^i_{k,t}\|^2_{\bar{R}_i} + \|x^i_{k,t} - \hat{x}^i_{k,t}\|^2_{\bar{T}_i} \right] + \|x^i_{k,N}\|^2_{\bar{P}_i} \tag{16}$$

where $\bar{J}^i_k$ is a finite-horizon cost function, which consists of a finite horizon standard cost, to specify the desired control performance and a terminal cost, to penalize the states at the end of the finite horizon.

The terminal region Ω^i for agent i is designed, so that it is invariant for nonlinear system controlled by a local linear state feedback.

3.2 Compatibility Constraint for Stability

We define $\xi^{-i} = x^{-*i} - \hat{x}^{-i}$, $\xi^i = x^{*i} - \hat{x}^i$, $\bar{Q}_i = \begin{bmatrix} \bar{Q}^1_i & \bar{Q}^{12}_i \\ (\bar{Q}^{12}_i)^{\mathrm{T}} & \bar{Q}^3_i \end{bmatrix}$.

$$C^*_x(k) = \sum_{i=1}^{N_a} \sum_{t=1}^{N-1} \{2(x^{*i}_{k,t})^{\mathrm{T}} \bar{Q}^{12}_i \xi^{-i}_{k,t} + 2(\hat{x}^{-i}_{k,t})^{\mathrm{T}} \bar{Q}^3_i \xi^{-i}_{k,t} + (\xi^{-i}_{k,t})^{\mathrm{T}} \bar{Q}^3_i \xi^{-i}_{k,t}\}, \tag{17}$$

$$C^*_\xi(k) = \sum_{i=1}^{N_a} \sum_{t=1}^{N-1} (\xi^i_{k,t})^{\mathrm{T}} \bar{T}_i \xi^i_{k,t}, \tag{18}$$

Lemma 2. *Suppose* (9) *holds and there exits* $\rho(k)$ *such that, for all* $k > 0$,

$$\begin{aligned} &0 \leq \rho(k) \leq 1, \\ &-\rho(k) \sum_{i=1}^{N_a} \{\|(x^i(k))^{\mathrm{T}}, (\hat{x}^{-i}_k)^{\mathrm{T}}\|^2_{\bar{Q}_i} + \|u^{*i}(0|k)\|^2_{\bar{R}_i}\} + C^*_x(k) - C^*_\xi(k) \leq 0. \end{aligned} \tag{19}$$

Then, by solving the receding-horizon optimization problem

$$\min_{\bar{U}^i(k)} \ \bar{J}_k^i, \ \ s.t.(1), (2), (14), (16), \ \ u_{k,N}^i = K_i x_{k,N}^i, \ \ x_{k,N}^i \in \Omega^i. \tag{20}$$

and implementing $u_{k,0}^{*i}$*, the stability of the global closed-loop system is guaranteed, once a feasible solution at time* $k = 0$ *is found.*

Satisfaction of (19) indicates that all $x_{k,t}^i$ should not deviate too far from their assumed values $\hat{x}_{k,t}^i$. Hence, (19) can be taken as a new version of the compatibility condition. This compatibility condition is derived from a single compatibility condition that collects all the states (whether predicted or assumed) with in the switching horizon and is disassembled to each agent in distributed manner, which results in local compatibility constraint for each agent.

Since $x_{k,t}^*$ for all agent i is coupled with other agents through (19), it is necessary to assign the constraint to each agent so as to satisfy (19) along the optimization.

Denote $\xi_k^i = [\xi_{k,x}^{i,q}, \xi_{k,y}^{i,q}, \xi_{k,x}^{i,v}, \xi_{k,y}^{i,v}]^{\mathrm{T}}$, $\xi_k^{-i} = \{\xi_k^j | j \in \mathcal{N}_i\}$. At time $k > 0$, by solving the optimization problem, there exits a parameter $\mathcal{E}_k^{i,l}$, $l = 1, \ldots, n_i$, for each element of $\xi_k^{i,l}$, $l = 1, \ldots, n_i$.

Define

$$\mathcal{E}_k^{i,l} = \max_t |\xi_{k-1,t}^{i,l}|. \tag{21}$$

And denote $\mathcal{E}_k^i = [\mathcal{E}_{k,x}^{i,q}, \mathcal{E}_{k,y}^{i,q}, \mathcal{E}_{k,x}^{i,v}, \mathcal{E}_{k,y}^{i,v}]^{\mathrm{T}}$, $\mathcal{E}_k^{-i} = \{\mathcal{E}_k^j | j \in \mathcal{N}_i\}$. At time $k+1 > 0$, set following constraint for each agent i.

$$|(x_{k+1,t}^i)^{\mathrm{T}} - (\hat{x}_{k+1,t}^i)^{\mathrm{T}}| < \mathcal{E}_k^i. \tag{22}$$

From (21) and (22), it is shown that $\xi_{k+1,t}^i < \mathcal{E}_k^i$ and $\xi_{k+1,t}^{-i} < \mathcal{E}_k^{-i}$.

Denote

$$C_x^{*i}(k) = \sum_{t=1}^{N-1} \{2(x_{k,t}^{*i})^{\mathrm{T}} \bar{Q}_i^{12} \mathcal{E}_k^{-i} + 2(\hat{x}_{k,t}^{-i})^{\mathrm{T}} \bar{Q}_i^3 \mathcal{E}_k^{-i} + (\mathcal{E}_k^{-i})^{\mathrm{T}} \bar{Q}_i^3 \mathcal{E}_k^{-i})^{\mathrm{T}}\}, \tag{23}$$

$$C_\xi^{*i}(k) = \sum_{t=1}^{N-1} (\xi_{k,t}^i)^{\mathrm{T}} \bar{T}_i \xi_{k,t}^i. \tag{24}$$

Then $C_x^*(k) \leq \sum_{i=1}^{N_v} C_x^{*i}(k)$, $C_\xi^*(k) = \sum_{i=1}^{N_v} C_\xi^{*i}(k)$.

By applying (21)-(24), it is shown that (19) is guaranteed by assigning

$$0 \leq \rho_i(k) \leq 1,$$

$$\sum_{i=1}^{N_v} -\rho_i(k)\{\|(x_k^i)^{\mathrm{T}}, (\hat{x}_k^{-i})^{\mathrm{T}}\|_{\bar{Q}_i}^2 + \|u_{k,0}^{*i}\|_{\bar{R}_i}^2\} + \sum_{i=1}^{N_v} C_x^{*i}(k) - \sum_{i=1}^{N_v} C_\xi^{*i}(k) \leq 0. \tag{25}$$

(25) is dispensed to agent i. However, the conservativeness is introduced.(25) is more stringent than (19).

Remark 2. By adding the deviation punishment term in the local cost function, the closed-loop stability follows with a large weight. The larger weight means the more loss of the performance [3].

3.3 Feasibility of Distributed MPC

The key issue to guarantee feasibility of the above receding horizon problems is how to satisfy the communication distance constraint.

$$\|\hat{q}^j_{k,t} - q^i_{k,t}\| \leq \bar{r}_i - \sqrt{(\mathcal{E}^{i,q}_{k,x})^2 + (\mathcal{E}^{i,q}_{k,y})^2}, \quad \sqrt{(\mathcal{E}^{i,q}_{k,x})^2 + (\mathcal{E}^{i,q}_{k,y})^2} < \bar{r}_i. \tag{26}$$

It is easy to find that (26) is the sufficient condition for communication distance constraint (6). But this constraint let the distributed receding horizon optimal problem is more conservative than the case of central. By revising the time-varying compatible constraint, the conservatism can be reduced.

4 Control Strategy

For practical implementation, distributed MPC is formulated as follow:

Off-line stage :

i) Set the value of the prediction horizon N and the communication radius $\bar{r}_i$.
ii) According to (3) (5) (9), find Q_i, R_i, $\bar{Q}_i$, $\bar{R}_i$, $t = 0, \cdots, N-1$, for all agents.
iii) Set the value of the compatibility constraint for all agents $\mathcal{E}_i(0) = +\infty$.
iv) Calculate the terminal weight P_i, local linear feedback control gain K_i and the terminal set Ω^i.
v) Set $\bar{T}_i(0) = 0$.

On-line stage :

For agent i, perform the following steps at $k \geq 0$:
i) Take the measurement of x^i_0.
ii) Send x^i_0 to its neighbor j, $j \in \mathcal{N}_i$ of agent i. Receive x^j_0.
iii) Set $\hat{x}^j_{t,0} = x^j_{0,0}$, $j \in \mathcal{N}_i$, $t = 0, \cdots, N-1$ and $\hat{x}^i_{0,t} = x^i_0$.
iv) Solve problem (20).
v) Implement $u^i_0 = u^{*i}_{0,0}$.
vi) Get $\hat{x}^i_{t,0}$ and compatibility constraint $\mathcal{E}_i(1)$.
vii) Send $\hat{x}^i_{0,t}$ and $\mathcal{E}_i(1)$ to neighbor j, $j \in \mathcal{N}_i$. Receive $\hat{x}^j_{0,t}$ and $\mathcal{E}_j(1)$. Calculate $\bar{T}_i(k)$.

5 Numerical Example

We consider the model of agent i [2] as

$$x^i_{k+1} = \begin{bmatrix} I_2 & I_2 \\ 0 & I_2 \end{bmatrix} x^i_k + \begin{bmatrix} 0.5I_2 \\ I_2 \end{bmatrix} u^i_k,$$

$(x_k^i = [q_k^{i,x}, q_k^{i,y}, v_k^{i,x}, v_k^{i,y}]^{\mathrm{T}}$, with sampling time interval of 0.5 second. There are four agents. A set of positions of the four agents constitute a formation.

The initial positions of the four agents are $[q_o^{1,x}, q_o^{1,y}] = [0, 1]$, $[q_o^{2,x}, q_o^{2,y}] = [-1, 0]$, $[q_o^{3,x}, q_o^{3,y}] = [0, -1]$, $[q_o^{4,x}, q_o^{4,y}] = [1, 0]$. Linear constraints on states and input are $\left|x^i\right| \leq \left[100\ 15\ 15\right]^{\mathrm{T}}$, $\left|u^i\right| \leq \left[2\ 2\right]^{\mathrm{T}}$. The agent i, $i = \{1, 2, 3\}$ are selected as the core agents of the formation. $\mathcal{A}_0$ is designed as $\mathcal{A}_0 = \{(1, 2); (1, 3); (2, 4)\}$. If all system achieve the desire formation and the core agents cooperatively cover the virtue leader, then $u_k^{i,x}(k) = 0$, $u_k^{i,y} = 0$. The global cost function is obtained as

$$\begin{aligned} J(k) = \sum_{t=0}^{\infty} \Big[& \left\| q_{k,t}^1 - q_{k,t}^2 + c_{12} \right\|^2 + \left\| q_{k,t}^1 - q_{k,t}^3 + c_{13} \right\|^2 + \left\| q_{k,t}^2 - q_{k,t}^4 + c_{24} \right\|^2 \\ & + \frac{1}{9} \left\| \left(q_{k,t}^1 + q_{k,t}^2 + q_{k,t}^3 \right) - q_c \right\|^2 \\ & + \left\| v_{k,t}^1 \right\|^2 + \left\| \ v_{k,t}^2 \right\|^2 + \left\| v_{k,t}^3 \right\|^2 + \left\| v_{k,t}^4 \right\|^2 + \left\| u_{k,t} \right\|^2 \Big]. \end{aligned}$$

They cooperatively track the virtual leader whose reference is $q_c = (0.5 * k, 0)$. The distance between agents is defined as $c_{12} = (1.5, 2)$, $c_{13} = (3, 1)$, $c_{24} = (0, -4)$. Choose $\mathcal{N}_1 = \{2\}$, $\mathcal{N}_2 = \{1\}$, $\mathcal{N}_3 = \{1\}$, $\mathcal{N}_4 = \{2\}$. Then,

$$Q = \begin{bmatrix} 2\frac{1}{9}I_2 & 0 & -\frac{8}{9}I_2 & 0 & -\frac{8}{9}I_2 & 0 & 0 & 0 \\ 0 & I_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{8}{9}I_2 & 0 & 2\frac{1}{9}I_2 & 0 & \frac{1}{9}I_2 & 0 & -I_2 & 0 \\ 0 & 0 & 0 & I_2 & 0 & 0 & 0 & 0 \\ -\frac{8}{9}I_2 & 0 & \frac{1}{9}I_2 & 0 & 1\frac{1}{9}I_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 \\ 0 & 0 & -I_2 & 0 & 0 & 0 & I_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_2 \end{bmatrix}, R = I_8.$$

$$\bar{Q}_1 = \begin{bmatrix} \frac{7}{9}I_2 & 0 & -\frac{4}{9}I_2 & 0 \\ 0 & \frac{1}{3}I_2 & 0 & 0 \\ -\frac{4}{9}I_2 & 0 & I_2 & 0 \\ 0 & 0 & 0 & \frac{1}{3}I_2 \end{bmatrix}, \ \bar{Q}_2 = \begin{bmatrix} 1\frac{1}{9}I_2 & 0 & -\frac{4}{9}I_2 & 0 \\ 0 & \frac{1}{3}I_2 & 0 & 0 \\ -\frac{4}{9}I_2 & 0 & \frac{4}{9}I_2 & 0 \\ 0 & 0 & 0 & \frac{1}{3}I_2 \end{bmatrix},$$

$$\bar{Q}_3 = \begin{bmatrix} 1\frac{1}{9}I_2 & 0 & -\frac{8}{9}I_2 & 0 \\ 0 & \frac{1}{2}I_2 & 0 & 0 \\ -\frac{8}{9}I_2 & 0 & \frac{8}{9}I_2 & 0 \\ 0 & 0 & 0 & \frac{1}{3}I_2 \end{bmatrix}, \ \bar{Q}_4 = \begin{bmatrix} I_2 & 0 & -I_2 & 0 \\ 0 & I_2 & 0 & 0 \\ -I_2 & 0 & I_2 & 0 \\ 0 & 0 & 0 & \frac{1}{3}I_2 \end{bmatrix}$$

and $\bar{R}_i = I_2$, $i \in \{1, 2, 3, 4\}$. Choose $Q_i = 6.85 * I_4$ and $R_i = I_2$, $i \in \{1, 2, 3, 4\}$, $N = 10$. The terminal set is $\alpha_i = 0.22$. The communication radius $\bar{r}_i = 4.5$. The above choice of model, cost and constraints allow us to rewrite problem (20) as a quadratic programming with quadratic constraint. To solve the optimal control problems numerically, the package NPSOL 5.02 is used.

From top to bottom, the first sub-graph of Fig 1 is the evolution of the formation with distributed MPC with time-varying compatible constraint; the second sub-graph of Fig 1 is the evolution of the formation with distributed MPC without the communication distance constraint. Compared with the first sub-graph, the second sub-graph have a large overshoot.

From top to bottom, the first sub-graph of Fig 5.2 is evolutions of actual distance between agents with distributed MPC with time-varying compatible constraint; the second sub-graph of Fig 2 is evolutions of actual distance between agents with distributed MPC without the communication distance constraint; As shown in Fig 2. "x" is for the distance between agent 1 and 2; "*" is for the distance between agent 1 and 3; "o" is for the distance between agent 2 and 4. Obviously, the distance between agent 1 and 2 is over the communication distance constraint in the second sub-graph.

The vale of $\rho_i(k)$ is shown in Fig 3. "□" for agent 1; "+" for agent 2; ">" for agent 3; "<" for agent 4.

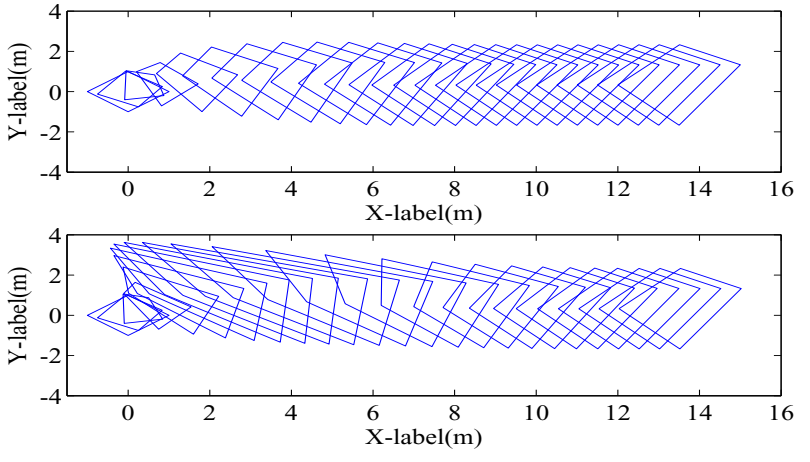


Fig. 1. Evolutions of the formation with different control schemes

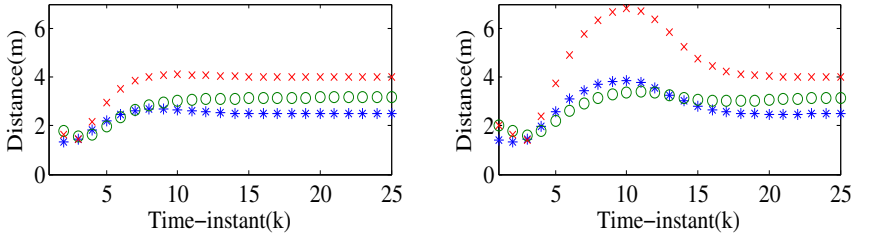


Fig. 2. Evolutions of the distance

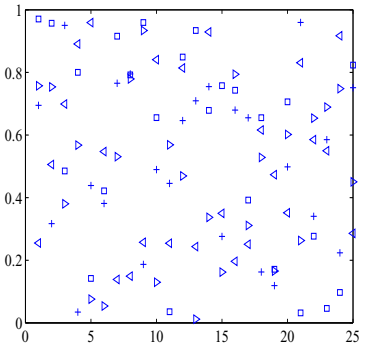


Fig. 3. The vale of $\rho_i(k)$

6 Conclusions

In this paper, we have proposed an improved distributed MPC scheme for multi-agent systems based on deviation punishment for communication distance constraint. One of the features of the proposed scheme is that the cost function of each agent penalizes the deviation between the predicted state trajectory and the assumed state trajectory, which improves the consistency and optimal control trajectory. At each sample time, the value of compatibility constraint is set by the deviation of previous sample time-instant. The close-loop stability is guaranteed with a small value for the weight of the deviation function term. Furthermore, the effectiveness of the scheme has been investigated by a numerical example. One of the future works will focus on the problem for the collision avoidance constraints.

References

1. Kuwata, Y., Richards, A., Schouwenaars, T., How, J.P.: Distributed robust receding horizon control for multivehicle guidance. IEEE Transactions on Control Systems Technology 15(4), 627–641 (2007)
2. Dunbar, W.B.: Distributed receding horizon control for multiagent systems. PhD thesis, California Institute of Technology, Pasadena, CA, 91125 (2004)
3. Dunbar, W.B.: Distributed Receding Horizon Control of Cost Coupled Systems. In: Proceedings of the 46th IEEE Conference on Decision and Control, pp. 2510–2515 (2007)
4. Chen, H., Allgöwer, F.: A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. Automatica 34, 1205–1217 (1998)
5. Johansen, T.A.: Approximate explicit receding horizon control of constrained nonlinear systems. Automatica 40, 293–300 (2004)
6. Ding, B.C.: Distributed robust MPC for constrained systems with polytopic description. Asian Journal of Control 13, 198–212 (2011)
7. Wei, S.B., Chai, Y., Ding, B.C.: Distributed model predictive control for multiagent systems with improved consistency. Journal Control Theory Application 8, 117–122 (2010)

Research on Human – Robot Collaboration in Rescue Robotics*

Haibo Tong, Rubo Zhang, and Guanqun Liu

College of Computer Science and Technology, Harbin Engineering University
Harbin, China
tonghaibo0801@hotmail.com,
{zhangrubo,liuguanqun}@hrbeu.edu.cn

Abstract. This paper presents a human-robot collaboration framework which describes a comprehensive structure of rescue robots which are expected to collaborate with human in urban search and rescue (USAR). We develop a victim's autonomous search and rescue robotic system based on biomimetic sensing technology and discuss the information flow model and control mode transition between human and robot. The intelligence control architecture used for human-robot collaboration is also proposed. Experiments indicate that this system makes the collaboration convenient and the rescue robotics utilizing the intelligence control architecture search and discover victims promptly and efficiently.

Keywords: human-robot collaboration, urban search and rescue, intelligence control architecture.

1 Introduction

Rescue robots are usually deployed in disaster areas mainly for search and rescue of victims [1]. It is commonly viewed as a tool: a device that performs tasks on command send by human, resulting that a robot has limited freedom and will perform poorly whenever it is unsuited for the task [2]. Robotic researchers have begun to change their view of robots from a tool or device operated by a human user to an assistant or a partner in the urban search and rescue. While USAR faces many unsolved problems such as mobility, sensing, and artificial intelligence, but the biggest challenge in rescue robotics stem from a limited understanding of human–robot interaction and collaboration. Thus, rescue robotics has been suggested by a recent DARPA/NSF study as an application domain for the research in human-robot collaboration (HRC) [3].

To make rescue robots more human-friendly and make them efficient in disaster environments, we need a robotic system which can execute tasks collaborating with

* This work was supported by the National High Technology Research and Development Program of China (863 Program) (No.2009AA04Z215) and the National Natural Science Foundation of China (No.60975071) and (No.61100005).

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 602–609, 2012.

human. Current applications of Urban Search and Rescue environment require a human operator to guide the robot remotely outside of the hot zone [4]. Although human operation can be effective and reliant, operators can become heavily stressed, fatigued and inefficient due to a loss of the general situation of disaster environment, causing critical error in control and victims identification [5]. An alternative to using human remote control is to develop autonomous controllers for rescue robot. However, there are a considerable number of issues in deploying an autonomous rescue robot in an unknown clutter disaster environment. To address this problem, collaboration between human and robot is essential.

K. Kosuge et al [6] propose a dynamic control algorithm for robot human collaboration system, in which human execute a task in cooperation with multiple robots in a common work environment. Ohba et al [7] developed a system where multiple operators in different locations control the collision free. In the paper we propose and realize a structure for human and robot collaboration. It describes the information, data and control flow between human and robot. Simultaneously, operator can change the control mode between manual control and autonomous control. Combining with unique hierarchical control architecture, the structure is used in a comprehensive victim's autonomous search and rescue system.

2 Human-Robot Collaboration

In this section, we are going to present a robot-human collaboration structure and consider the information and data flow and the control mode transition between human and robot.

To find a victim in a disaster scene, it is a difficult mission especially in the clutter environment. The physical characteristic of a victim that we can detect contains sound, shape of body, skin color, and clothing texture and so on. The victim's autonomous search and rescue system (Fig. 1) we developed is equipped with many kinds of sensors, such as sound sensor, vision sensor, sonar, compass CO sensor. The vision sensors including light camera and infrared camera can provide 3D scene of the disaster environment. Sound source directional localization algorithm, designed to orient the victims, is implemented, utilizing the data provided by sound sensor. Sonar is used to respond the distance from robot to obstacle, to implement robot's obstacle avoiding. CO sensor can detect the CO concentration.

The sensor data are sent to the control architecture used to implement victim's autonomous search and rescue, obstacle avoiding, and complete coverage of the disaster environment. On the other hand, these data are also delivered to user interface, operator can observe data change and make decision whether or not change control mode or not. The Fig. 2 show the data flow frame and control mode transition for human-robot collaboration.

Fig. 1. The Urban Search and Rescue Robotic System

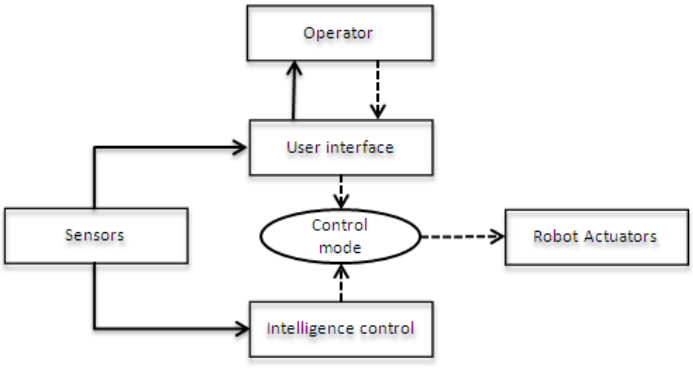


Fig. 2. Data and control command flow for human-robot collaboration

We also design a friendly user interface (Fig. 3) used to observe data of disaster environment and change the control mode for operator when necessary.

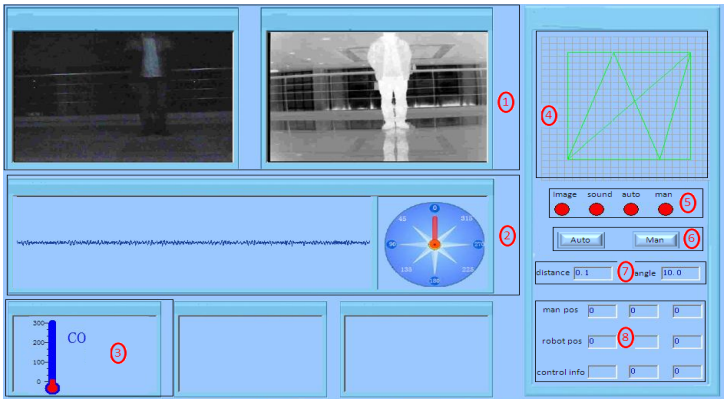


Fig. 3. User interface for human operator

① These two windows show the true disaster environment picked up by light camera (left) and infrared camera (right) respectively.
② This part shows the sound data in the form of sound wave (left) recorded by microphones and the direction (right) of victim detected by sound source localization algorithm.
③ This part displays the CO concentration of the disaster environment.
④ This window shows the track by which robot walks.
⑤ This part shows the control status.
⑥ Using these two buttons, operator can change the control mode between autonomous control and human control.
⑦ This part shows the command which the robot is executing.
⑧ This part shows the positions of victims and robot in the first two rows and the control information in the last row.

3 Intelligence Control Architecture

This section will present the data processing and intelligence control architecture we design. Data generated from sound module and vision module is real-time and isolated. Considering the disaster scene is usually noisy and clustered and sensor data do not consider the historical data, it will not be completely credible. We propose a data processing algorithm based on historical data cumulative statistics particularly for sound source detection and orientation. Simultaneously, we design a control architecture applying to guide the search and rescue robot to cover disaster area and discover victims completely and efficiently.

In general, robot control architecture can be defined as deliberative, reactive or hybrid. Deliberative control consists of high-level planning, whereas reactive control executes the results generated and calculated from the sensory data. Our proposed control architecture (Fig. 4) is hierarchical.

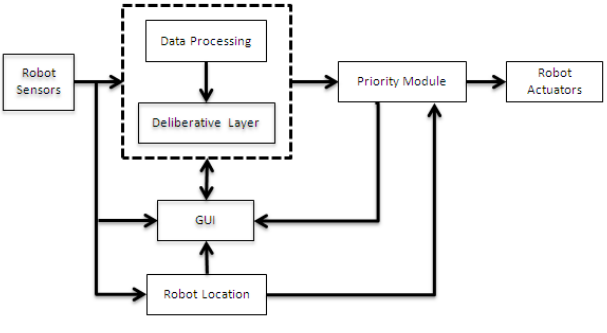


Fig. 4. Intelligence control architecture

The robot control architecture proposed in our search and rescue robot system consists of the following modules:

1) Robot Sensors: There are many kinds of sensor, such as light and infrared camera, three-dimensional digital compass and other internal/external sensors. Data generated and calculated from sensory system is the inputs of the control system. They have different characteristic and different advantages and disadvantages to reflex the environment from different respects.

2) Robot Location: The search and rescue system utilizes the three dimensional digital compass and internal encoders to calculate the position of the robot in the disaster environment. Robot location is the foundation of the system.

3) Data Processing: In this phase, data from different sensor modules such as sound module, vision module, and sonars are handled with the purpose of gathering effective result, reducing error, eliminating contradiction and redundancy.

4) Deliberative Layer: In this layer, we process the orthogonal data according the characteristic and role in task cycle. Decision is made by analyzing the results of last data processing phase. It is also the deliberative layer where the level of autonomy between human control and autonomous control is primarily decided.

5) Priority Module: Vision module is aimed at detecting the disaster environment and identifying victims, in contrast, sound module is aimed at detecting sound and confirm the orientation of the human and guiding the robot to move towards victims. Sonar data is used to implement robot obstacle avoiding and ensure the safety of search robot. Generally, in the priority module, we give the highest priority to obstacle avoiding, middle priority to vision module and lowest priority to sound module.

6) Robot Actuators: The robot actuators module consists of the robot's motors and motor control boards. Appropriate motor signals are sent based on the result of above modules.

4 Experiments

Experiments are designed to verify the performance of the Urban Search and Rescue Robotic System and the intelligence control architecture. The experiments are implemented in a room and a corridor respectively Fig. 5. The rubble-like objects within the simulative disaster scene include wood plastic, cardboard and so on.

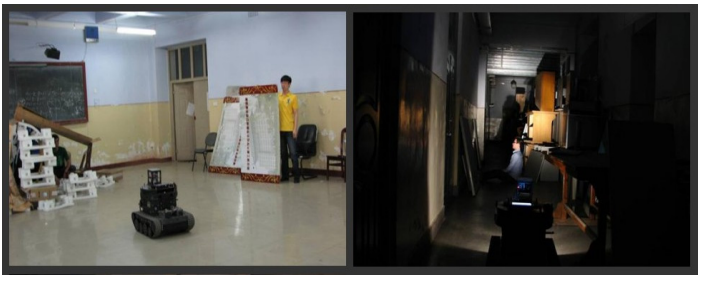


Fig. 5. Experiment environment

In the room, the scene consists of three victims who hold the cardboard in their hands, ensuring that only portion of their body was visible, limbs or head. They sit in a circle and call for help in turn (Fig. 6). When sound module of the search and rescue system detect voice, and oriented the victims (step 2). It will guide the robot to rotate and move towards the potential victims (step 3 and 4). The vision module detects the environment all the time. When vision identifies a victim, it will mark the victim, record the coordinate and inform the GUI by sending the region of victims on the image. GUI can draw this region (the red rectangle) on the screen (step 5).

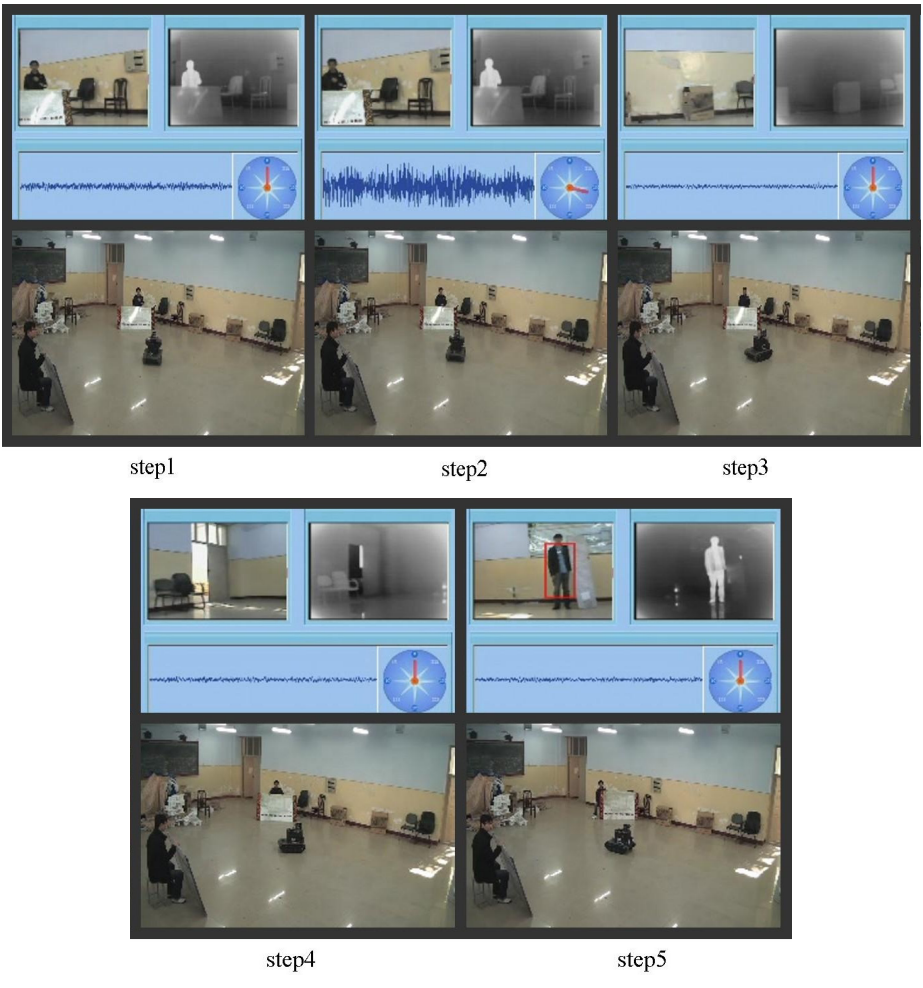


Fig. 6. Experiment scene in light environment

Fig. 7 show the experiment in corridor, where victims hide themselves in the woodpile and the robot is in manual control mode. Operator can observe disaster scene via the video generated from light camera and infrared camera and control robot with keyboard.

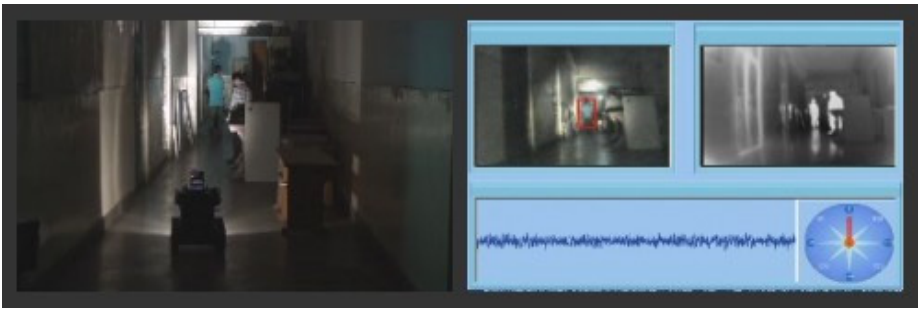


Fig. 7. Experiment scene in the corridor

The ratio of discovering victims under different environment is presented in Table 1. It is defined as the ratio that the search and rescue system detects and identifies victims. There are 11 items added up with different Illumination(ILL), Background(BG) and Pose of Victims(PoV). The average ratio is high than 90% obviously.

Table 1. Ratio of discover victims

SN	ILL	BG	PoV	Person-time of victims	Person-time detected	Ratio of discover victim (%)
1	Normal	Complex	Stand\ Block partly	32	30	93.75
2	Normal	Complex	Stand\ Block partly	36	34	94.44
3	Weak	Simple	Sit\ Non-blocked	10	9	90.00
4	Weak	Simple	Sit\ Non-blocked	17	16	94.12
5	Weak	Complex	Stand\Block	28	26	92.86
6	Weak	Complex	Stand\Block	10	9	90.00
7	Weak	Complex	Stand\ Block heavily	32	30	93.75
8	Weak	Complex	Stand\ Block heavily	34	31	91.18
9	Weak	Complex	Stand\ Non-blocked	25	23	92.00
10	Weak	Complex	Stand\ Non-blocked	11	10	90.91
11	Weak	Complex	Stand\Blocked	13	12	92.31

5 Conclusion and Future Work

Urban search and rescue robotic system has been an important application in the field of human-robot collaboration. In this paper, we propose a human-robot collaboration

structure and an intelligence control architecture. The intelligence control architecture provides the robot system with the ability to make decisions regarding which task should be implemented and which control mode is employed. Utilizing our proposed intelligence control architecture, the search and rescue robotic system can search and identify victims quickly and efficiently.

The future work will focus on the improving of victim's detection and identification algorithm and robot mission planning in the clutter and unknown environment. Furthermore, we will pay close attention to the design and guidelines of human-robot collaboration with the purpose of promoting robot performance and efficiency.

Acknowledgments. The authors would like to especially thank Rubo Zhang, Guanqun Liu, Zhihui Li, Chunyan Shao and Xianglei Zhang for their contributions to the paper and the robot system and gratefully acknowledge assistance of Jianxin Wang, Dahai Yu and Zhongqiu Guo for their accomplishing these experiments.

References

1. Adluru, N., Latecki, L.J., Lakaemper, R., Madhavan, R.: Robot mapping for rescue robots. In: Proc. of the IEEE Int. Workshop on Safety, Security and Rescue Robotics (SSRR), Gaithersburg, Maryland, USA (2006)
2. Terrence, F., Charles, T., Charles, B.: Collaboration, Dialogue, and Human-Robot Interaction. In: Springer Tracts in Advanced Robotics, vol. 6, pp. 255–266. Springer, Berlin (2003)
3. Murphy, R.: Human–Robot Interaction in Rescue Robotics. IEEE Transactions on System, Man, and Cybernetics-Part C: Application and Review 34, 138–153 (2004)
4. Barzin, D.: A Hierarchical Reinforcement Learning Based Control Architecture for Semi-Autonomous Rescue Robots in Cluttered Environment. In: 6th Annual IEEE Conference on Autonmation Science and Engineering, Toronto, Canada, vol. A247, pp. 948–953 (2010)
5. Casper, J., Murphy, R.: Workflow study on human-robot interaction in USAR. In: IEEE International Conference on Robotics and Autonomous, Florida, USA, vol. 2, pp. 1997–2003 (2002)
6. Kosuge, K., Yoshida, H., Fukuda, T.: Dynamic Control for Robot-Human Collaboration. In: Proc. IEEE Intl. Workshop on Robot and Human Interactive Communication, Tokyo, Japan, pp. 398–401 (1993)
7. Ohba, K., Kawabata, S., Chong, N.Y., Komoriya, K., Matsumaru, T., Matsuhira, N.: Remote collaboration through time delay in multiple teleoperation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 1999): Human and Environment Friendly Robots with High Intelligence and Emotional Quotients, Kyongju, South Korea, vol. 3, pp. 1866–1871 (1999)

Development of Visual Action Design Environment for Intelligent Toy Robot

Jianqing Mo[1,2], Hanwu He[1], and Hong Zhang[1,3]

[1] Guangdong University of Technology, Guangzhou, China
[2] Top Key Discipline of Computer Software and Theory in Zhejiang Provincial Colleges, Zhejiang Normal University, Jinhua, China
[3] Guangdong South China Institute of Industrial Design, Dongguan, China
{momolon,hwhe}@gdut.edu.cn

Abstract. To improve efficiency of action design for intelligent toy robot, a visual action design environment is created. Compared with the text-based action design software, it's intuitive, good usability. An algorithm for an automatic generation action sequence file (ASF) is present according to the data structure of action sequence, the file structure of ASF and the judgment methods of operation rules. Then using Eon Studio as virtual simulation development platform, the analysis to complicated action sequence files and reappearance of virtual action are realized.

Keywords: Intelligent Toy Robot, Virtual Reality, Action Design, Algorithm.

1 Introduction

For intelligent toy robots, numerous, complex and vivid action is one of the key factors that attract players. Direct programming according to action plans is the main way of completing action design traditionally. Codes are used to drive the actual robots to test, and then improving actions based on test results. It doesn't enable designers to see results immediately after adding, modifying or deleting actions. Thus, the efficiency of action design is low. It is not sure whether the actions are both consistent and elegant. And it is not sure whether the robot can keep balance at movement, or meet the requirement of avoiding self collision. Under the situation that multiple robots are performing dancing, wrestling or gymnastics, action coordination is necessary, but the traditional method is difficult to reach that. In addition, it is hard to intuitionally judge the allowable move space for robots by the traditional method during the period of product design. It is also difficult to make sure that the structure design of robots can meet the requirement of complicated movement balance. Therefore, it cannot insure the ability of complex action design for toy robots.

The programmable function is an important character of intelligent toy robots. Allowing players to design and develop actions would bring them great joy and intelligence practicing opportunities. However, the text-based or icon-based programming environment [1],[2] is not suitable for ordinary players, especially low-age children. Thus, it is necessary to develop a powerful, interactive and visual environment that enables fast action planning and real-time preview.

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part I, LNCS 7331, pp. 610–617, 2012.

In this paper, a visual action developing environment based icon editing and virtual scene has been created.

2 Data Structure of Action Sequence

2.1 Logic Structure

Action sequence is an action assembly made up of one or more actions arrayed by time. A set of consecutive actions can become an action sequence. Given that there may be concurrent actions that are totally or partly synchronized in time, and there may be some repeated serial actions, the logic relation of action sequence is not merely linear.

Parallel actions are illustrated by branch structure. For those repeated single action or series actions, cyclic structure is used. Several different type of action sequence graphs are illustrated in Fig.1. a) is a linear action sequence, b) includes parallel action sequence, c) cyclic action sequence and d) shows the nested parallel structures .

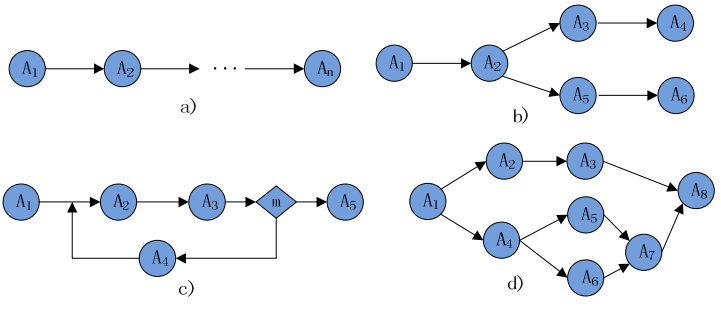


Fig. 1. Different action sequence graphs

2.2 Storage Structure

The logic structure of action sequence is shown as a directed graph, so the adjacency list is chosen for the storage structure. For the convenience of regulation judgment and traverse algorithm, redundant data is added. The redundant data will take extra storage space, resulting in inconvenience when deleting actions. The pros and cons are compared comprehensively. The structure of the list and head nodes is shown as Fig. 2.

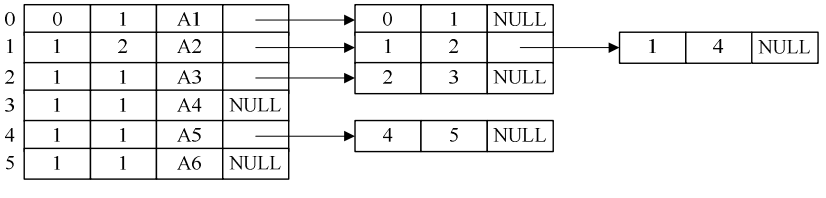


Fig. 2. Adjacency list

3 Algorithm for Automatic Generation of ASF

3.1 Operation Rules and Judgment Algorithm

Action sequence is an action assembly made up of one or more actions arrayed by time order. This sequence must conform to certain constraint conditions to satisfy the correctness and integrity of action sequence. Rule judgment must be conducted while the user adds, deletes, or changes the actions. Warning and rejecting events are activated when a certain action does not conform to the rules.

Take adding directed line as an example. The directed lines in action sequence graph indicate the continuation of actions. It means the ending of an old action and the starting of a new one. There are two reasons for breaking the correctness and integrity of action sequence. One is transmitting antinomy, another is concurrent mistake.

- **Transmitting Antinomy**

Transmitting antimony occurs when two routes such as $\mathbf{A_i} \rightarrow \ldots \rightarrow \mathbf{A_j} \rightarrow \ldots \rightarrow \mathbf{A_k}$ and $\mathbf{A_i} \rightarrow \mathbf{A_k}$ in one action sequence appear at the same time.

- **Concurrent Mistake**

If two intersecting paralleling structures in one action sequence contain a public subsequence, this action sequence has a concurrent mistake as shown in Fig.3.

Further analyses of the characteristics of transmitting antinomy and concurrent mistake reveal their common characteristic--the *source node* is directly connected with the *joint node*. Here source node is a node with more than one direct subsequence, while joint node is a node with more than one direct predecessor. As shown in Fig. 3, node A_2 becomes a source node and node A_6 becomes a joint node after directed line 2 is added.

Therefore, the rule for right connection operation is as follows: A directed line starting from a source node and ending to a joint node is not allowed in an action sequence.

According to the above rules, two situations can be used to judge whether the user breaks the rules during connection operation.

(1) Direct breach of the rules after connection because A_i becomes the source node and A_j becomes the joint node when the line<A_i, A_j> is added.
(2) Two situations of the indirect breach of rules occur after connection.

a) Some direct subsequence of A_i is joint node. A_i becomes the source node when the line <A_i, A_j> is added.
b) Some direct predecessor of A_j is source node. A_j becomes the joint node when the line<A_i, A_j> is added.

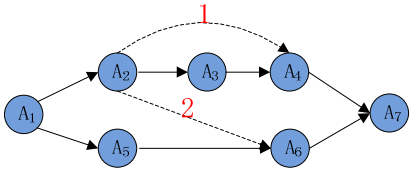


Fig. 3. Transmitting antinomy raised when arc1is added, and concurrent mistake when line2 is added

3.2 Automatic Generation of ASF

ASF (Action sequence file) is a structured data file stored in txt or xml format. The files are made up of different markups and action data. Take the action sequence in Fig. 1 b) as an example. The file structure of this action sequence is shown as Fig.4.

The linear structures are represented by a couple of markups named Begin Linear Action/End Linear Action. The parallel structures are represented by Begin Parallel Action/End Parallel Action. For the convenience of algorithm, cyclic structure is turned into an equal linear structure.

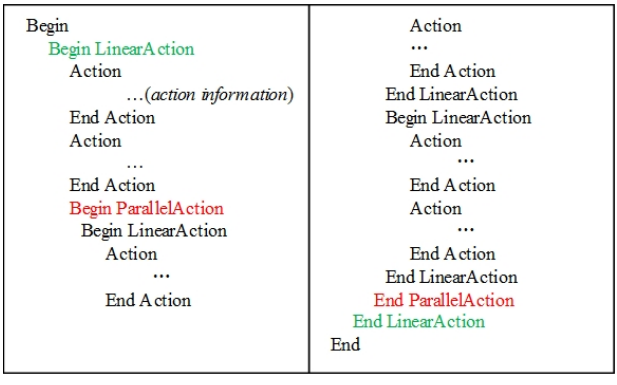


Fig. 4. The structure of action sequence file

The idea of the algorithm lies in that different markups are created when traversing each action node of directed graph. The key of the algorithm is how to traverse a directed graph and when the text information is add to the file. Correct action sequence files must be generated for all legal action sequences. Handling parallel structure is the difficult part of the algorithm, especially when complicated action sequences are included. To avoid ambiguity, the parallel structure in ASFs contains strict expression.

In general situations, parallel structure starts from the source node and ends at the joint node. Nonetheless, for convenience and usability of action sequence edition, not all parallel structures are required to contain source node and joint node at the same time. To deal with the matching between source nodes and joint nodes, virtual source nodes (S_0 in Fig. 5) and virtual joint nodes (J_0, J_1, and J_2 in Fig.5) are add to the action sequence graph.

- **Pretreatment**

The goal of pretreatment is to add virtual source nodes and virtual joint nodes by algorithm, as well as to find the corresponding relations of all source nodes and joint nodes.

(1) Data Structure

Four linear arrays are used to store data. The array *Rear* is used to store the end nodes that have no direct subsequence, the array *RSource* to store the source nodes

corresponding to the end nodes, the array *Source* to store all the source nodes, and the array *Joint* to store all the joint nodes.

(2) Adding Virtual Source Nodes

All the starting nodes that have no predecessor are identified. If the staring node is more than 1, a public direct predecessor S_0 is added and the value is set as -1.

(3) Adding Virtual Joint Nodes

The algorithm of adding virtual joint nodes is as follows:

Step1: Find all the end nodes and store them in the array *Rear*.
Step2: Stop when the length of the array *Rear* is equal to or smaller than 1.
Step3: Take a node A_i out from the array *Rear* one by one. Searching by backtracking method in the action sequence graph until a source node is found. Put it into the array *RSource*.
Step4: A virtual joint node J_i is added according to the nodes with the same value in the array *RSource*. Delete two direct predecessors of virtual node J_i from the *Rear* and put J_i into the *Rear*.
Step5: According to Step4, store A_i in the array *Source* and store J_i into the array *Joint*. Then turn to Step2.

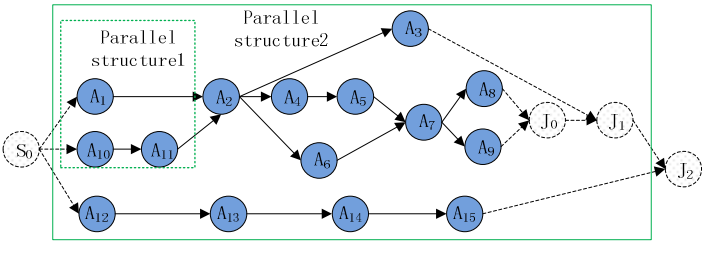


Fig. 5. The virtual source node and the virtual joint nodes were added

(4) Matching Between Joint Nodes and Source Nodes

Source nodes corresponding to virtual joint nodes have already been found when the virtual nodes were added. Therefore, only the relations of other joint nodes and source nodes (including virtual source nodes) need to be identified. It is the way to search source node that backtracking along all routes starting from the joint node A_j until a public source node A_i is found. But the joint node found along different routes starting from source node A_i may be different if there is more than one parallel structure. In this case, special handling is needed.

(5) Special Handling

One solution is to take the method of adding virtual source nodes in order to keep the source node and the joint node in a one-to-one relationship. However, this method will break the original structure of action sequence, making it inconvenient for storage and editing. Another solution is to record the routes between a source node A_j and its different corresponding joint nodes. Such as the source node A_2 in Fig.5, three routes need to be recorded: $A_2A_3J_1$, $A_2A_4J_7$, and $A_2A_6J_7$. The second solution is taken in this paper.

- **Algorithm Process**

The purpose of the algorithm is to insert markups (Table 1) and action data into the ASF when traversing an action sequence graph. The traverse method is not merely depth-first traverse or width-first traverse. The algorithm described by pseudo code is showed in Fig.6.

Table 1. Diffrent markups

Name	BM	EM	BPAM	EPAM	BLAM	ELAM	AM	EAM
Mark-up	Begin	End	Begin Parallel-Action	End Parallel-Action	Begin Linear-Action	End Linear-Action	Action	End Action

```
TranverseFromNode(int n)
{
  if( n is a head node )
      visit(n,1); // Add a BM markup
  if(n is a source node)
  {
     SourceStack.Push(n);
     if( n is not a virtual source node )
        visit(n,0);    // Output the action
        data
     visit(n,2); // Add a BPAM markup
     while( Branch i has not been visited )
     {    visit(n,3); // Add a BLAM markup
          j=GetFirstNode(i);// Get the head
          node in branch i
          TranverseFromNode(j);
        visit(n,-3); // Add a ELAM markup
     }
     visit(n,-2); // Add a EPAM markup
     j= SourceStack .Pop();
     k=GetCurrentJoint(j);
     if( k is not a source node)
     {      if( k is not a virtual joint node)
          visit(n,0);
          k=Next(k);
     }
     if(k exists )
       TranverseFromNode(k);
  }
  else if(n is a joint node )  return;
  else
  {
     visit(n,0);
     j= Next(n); // Backward traverse
     with depth-first method
     if(j exists)
       TranverseFromNode(j);
  }
 if(n is a end node)
   visit(n,-1); // Add a EM markup
}
```

Fig. 6. The algorithm described by pseudo code

4 Editor Development and Testing Results

VC++ is used as a development tool for action editor while Eon Studio 7.0 is used for the virtual simulation. After the action sequence is edited, it is stored as an act format file (.act). An action sequence file in txt format can be generated afterwards. It can drive the humanoid robot to perform actions (shown as Fig. 7) after the action sequence file analysis (another paper will be written to describe this part) is

conducted. Therefore, during the process of action design, a preview of the action effect can be accomplished, which can help hasten action design. Through this editor, action designers can efficiently design action sets without editing any code.

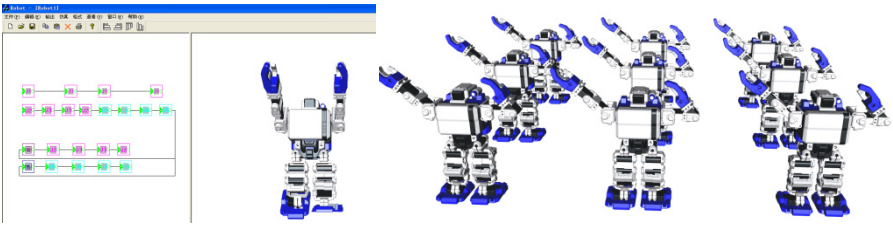


Fig. 7. Toy intelligent robots are performing actions

5 Conclusion and Prospect

Majority of the current virtual reality simulation applications are accomplished by creating the activities of virtual objects inside the virtual scene files [5] [6] [7]. This paper proposes a way to obtain the data of a driven virtual object's action changes by reading external files. The paper is a significant attempt to create the dynamic interaction of virtual objects (scenes) driven by large amount of external data. By separating virtual scene files from simulation data, more flexible and stronger virtual simulation functions can be completed. When real-time external data need to be collected, driving the scenes to obtain real-time changes by collected results is extremely significant.

Acknowledgments. This work was supported by the Project of Science and Technology of Guangdong Province (Grant No. 2010A040307001, 2008A060301003), the Project of Science and Technology of Dongguan City (Grant No. 201010810202, 200910810208), and the Opening Fund of Top Key Discipline of Computer Software and Theory in Zhejiang Provincial Colleges at Zhejiang Normal University (Grant No. ZSDZZZZXK19).

References

1. Bao, X.-J., Chen, W.-D., Cao, Q.-X.: An Icon-Based Robot Programming Environment: Design and Implementation. Robot 28(6), 617–622 (2006)
2. Kim, S.H., Jeon, J.W.: Programming LEGO Mindstorms NXT with visual programming. In: International Conference on Control, Automation and Systems 2007, Seoul, Korea (October 2007)
3. Hong, B.-R., Lu, D.-S., Gao, Q.-S.: Modeling of home robot based on virtual reality. Journal of Harbin Institute of Technology 36(7), 899–901 (2004)
4. Yang, K., Liu, S., Chen, D.: Modeling in Virtual Reality. Journal of Wuhan University of Technology 23(6), 47–50 (2001)

5. Gu, Y., He, H., Hu, Z., Wu, Y.: Web Based Teach-in Simulation for S-700 Robot. Microcomputer Information 23, 253–255 (2007)
6. Wu, Y., He, H., Gu, Y., Lu, Y.: Virtual Reality Based CNC Machine Modeling and Machining Simulation. In: Proceedings of the 7th International Conference on Frontiers of Design and Manufacturing, vol. 3, pp. 67–72 (2006)
7. He, H., Wu, Y., Pan, H., Zheng, D.: Visualized interactive manipulation in virtual assembly. Journal of Computational Information Systems 3(1), 387–392

Author Index