Ying Tan
Yuhui Shi
Kay Chen Tan (Eds.)

LNCS 6145

Advances in Swarm Intelligence

First International Conference, ICSI 2010
Beijing, China, June 2010
Proceedings, Part I

1 Part I

Springer

Ying Tan Yuhui Shi Kay Chen Tan (Eds.)

Advances in Swarm Intelligence

First International Conference, ICSI 2010
Beijing, China, June 12-15, 2010
Proceedings, Part I

Springer

Volume Editors

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

Yuhui Shi
Xi'an Jiaotong-Liverpool University, Research and Postgraduate Office
Suzhou, 215123, China
E-mail: yuhui.shi@xjtlu.edu.cn

Kay Chen Tan
National University of Singapore
Department of Electrical and Computer Engineering
4 Engineering Drive 3, 117576 Singapore
E-mail: eletankc@nus.edu.sg

Library of Congress Control Number: 2010927598

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

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

ISSN 0302-9743
ISBN-10 3-642-13494-7 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-13494-4 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Preface

This book and its companion volume, LNCS vols. 6145 and 6146, constitute the proceedings of the International Conference on Swarm Intelligence (ICSI 2010) held in Beijing, the capital of China, during June 12-15, 2010. ICSI 2010 was the first gathering in the world for researchers working on all aspects of swarm intelligence, and provided an 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 of swarm intelligence research.

ICSI 2010 received 394 submissions from about 1241 authors in 22 countries and regions (Australia, Belgium, Brazil, Canada, China, Cyprus, Hong Kong, Hungary, India, Islamic Republic of Iran, Japan, Jordan, Republic of Korea, Malaysia, Mexico, Norway, Pakistan, South Africa, Chinese Taiwan, UK, USA, Vietnam) across six continents (Asia, Europe, North America, South America, Africa, and Oceania). Each submission was reviewed by at least three reviewers. Based on rigorous reviews by the Program Committee members and reviewers, 185 high-quality papers were selected for publication in the proceedings with the acceptance rate of 46.9%. The papers are organized in 25 cohesive sections covering all major topics of swarm intelligence research and development.

In addition to the contributed papers, the ICSI 2010 technical program included four plenary speeches by Russell C. Eberhart (Indiana University Purdue University Indianapolis, IUPUI, USA), Gary G. Yen (President of IEEE Computational Intelligence Society, CIS, Oklahoma State University, USA), Erol Gelenbe (London Imperial College, UK), Nikola Kasabov (President of International Neural Network Soceity, INNS, Auckland University of Technology, New Zealand). Besides the regular parallel oral sessions, ICSI 2010 also had several poster sessions focusing on wide areas.

As organizers of ICSI 2010, we would like to express sincere thanks to Peking University and Xi'an Jiaotong-Liverpool University for their sponsorship, to the IEEE Beijing Section, International Neural Network Society, World Federation on Soft Computing, Chinese Association for Artificial Intelligence, and National Natural Science Foundation of China for their technical co-sponsorship. We appreciate the National Natural Science Foundation of China and K.C. Wong Education Foundation, Hong Kong, for their financial and logistic supports.

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 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 2010 successful and all the hard work worthwhile.

June 2010

Ying Tan
Yuhui Shi
Tan Kay Chen

Organization

Honorary Chairs

Qidi Wu, China
Russell C. Eberhart, USA

General Chair

Ying Tan, China

Advisory Committee Chairs

Zhenya He, China
Xingui He, China
Xin Yao, UK
Yixin Zhong, China

Program Committee Chairs

Yuhui Shi, China
Tan Kay Chen, Singapore

Technical Committee Chairs

Gary G. Yen, USA
Jong-Hwan Kim, South Korea
Xiaodong Li, Australia
Xuelong Li, UK
Frans van den Bergh, South Africa

Plenary Sessions Chairs

Robert G. Reynolds, USA
Qingfu Zhang, UK

Special Sessions Chairs

Martin Middendorf, Germany
Jun Zhang, China
Haibo He, USA

Tutorial Chair

Carlos Coello Coello, Mexico

Publications Chair

Zhishun Wang, USA

Publicity Chairs

Ponnuthurai N. Suganthan, Singapore
Lei Wang, China
Maurice Clerc, France

Finance Chair

Chao Deng, China

Registration Chairs

Huiyun Guo, China
Yuanchun Zhu, China

Program Committee Members

Peter Andras, UK
Bruno Apolloni, Italy
Payman Arabshahi, USA
Sabri Arik, Turkey
Frans van den Bergh, South Africa
Christian Blum, Spain
Salim Bouzerdoum, Australia
Martin Brown, UK
Jinde Cao, China
Liang Chen, Canada
Zheru Chi, Hong Kong, China
Leandro dos Santos Coelho, Brazil
Carlos A. Coello Coello, Mexico
Emilio Corchado, Spain
Oscar Cordon, Spain
Jose Alfredo Ferreira Costa, Brazil
Xiaohui Cui, USA
Arindam Das, USA
Prithviraj Dasgupta, USA
Kusum Deep, India
Mingcong Deng, Japan
Yongsheng Ding, China
Haibin Duan, China
Mark Embrechts, USA
Andries Engelbrecht, South Africa
Meng Joo Er, Singapore
Peter Erdi, USA
Yoshikazu Fukuyama, Japan
Wai Keung Fung, Canada
Ping Guo, China
Luca Maria Gambardella, Switzerland
Erol Gelenbe, UK
Mongguo Gong, China
Jivesh Govil, USA
Suicheng Gu, USA
Qing-Long Han, Australia

Haibo He, USA
Zhengguang Hou, China
Huosheng Hu, UK
Xiaohui Hu, USA
Guangbin Huang, Singapore
Amir Hussain, UK
Zhen Ji, China
Colin Johnson, UK
Nikola Kasabov, New Zealand
Arun Khosla, India
Franziska Klugl, Germany
Lixiang Li, China
Yangmin Li, Macao, China
Kang Li, UK
Xiaoli Li, UK
Xuelong Li, UK
Guoping Liu, UK
Ju Liu, China
Fernando Lobo, Portugal
Chris Lokan, Australia
Wenlian Lu, China
Hongtao Lu, China
Wenjian Luo, China
Xiujun Ma, China
Jinwen Ma, China
Bernd Meyer, Australia
Martin Middendorf, Germany
Hongwei Mo, China
Francesco Mondada, Switzerland
Ben Niu, China
Erkki Oja, Finland
Mahamed Omran, Kuwait
Paul S. Pang, New Zealand
Bijaya Ketan Panigrahi, India
Thomas E. Potok, USA
Jose Principe, USA
Ruhul A. Sarker, Australia
Gerald Schaefer, UK
Giovanni Sebastiani, Italy
Michael Small, Hong Kong, China
Ponnuthurai Nagaratnam Suganthan, Singapore
Norikazu Takahashi, Japan
Ying Tan, China
Ran Tao, China
Peter Tino, UK
Christos Tjortjis, Greece
G.K. Venayagamoorthy, USA
Ling Wang, China
Guoyin Wang, China
Bing Wang, UK
Lei Wang, China
Cheng Xiang, Singapore
Shenli Xie, China
Simon X. Yang, Canada
Yingjie Yang, UK
Dingli Yu, UK
Zhigang Zeng, China
Yanqing Zhang, USA
Qingfu Zhang, UK
Jie Zhang, UK
Lifeng Zhang, China
Liangpei Zhang, China
Junqi Zhang, China
Yi Zhang, China
Jun Zhang, China
Jinhua Zheng, China
Aimin Zhou, China
Zhi-Hua Zhou, China

Reviewers

Ajiboye Saheeb Osunleke
Akira Yanou
Antonin Ponsich
Bingzhao Li
Bo Liu
Carson K. Leung
Changan Jiang
Chen Guici
Ching-Hung Lee
Chonglun Fang
Cong Zheng
Dawei Zhang
Daoqiang Zhang
Dong Li

Table of Contents – Part I

Table of Contents – Part II

Stability Problem for a Predator-Prey System

Zvi Retchkiman Konigsberg

Instituto Politecnico Nacional

Abstract. This paper considers the growth rate dynamics of a predator-prey system 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. Lyapunov stability theory provides the required tools needed to aboard the stability problem for the predator-prey system treated as a discrete event system modeled with timed petri nets. By proving boundedness one confirms a dominant oscillating behavior of both populations 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: Predator-Prey System, Discrete Event Dynamical Systems, Max-Plus Algebra, Lyapunov Method, Timed Petri Nets.

1 Introduction

Consider the interaction of populations, in which there are exactly two species, one of which the *predators* eats the other the *prey* thereby affecting each other's growth rates. In the study of this growth rate dynamics Lotka-Volterra models have been used [5]. This paper proposes a new modeling and analysis methodology which consists in considering the predator-prey system 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 the predator-pray system treated as a discrete event system modeled with timed petri nets whose mathematical model is given in terms of difference equations [3]. 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 guaranteed. By proving boundedness one confirms a dominant oscillating behavior of both populations 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 stability problem for predator-pray systems treated as discrete event dynamical systems modeled with timed Petri nets. This results in a qualitative approach vs the quantitative approach obtained by solving the Lotka-Volterra differential equations system that models it. 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 predator-prey dynamical system's stability. Finally, the paper ends with some conclusions.

2 Lyapunov Stability and Stabilization of Discrete Event Systems Modeled with Petri Nets [3]

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*

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

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 $a_{ij}^- = W(p_j, t_i)\dot{}$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 $(N^m_{n_0}, d)$ be a metric space where $d : N^m_{n_0} \times N^m_{n_0} \rightarrow R_+$ 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 [1,2,4]

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 $\mathbb{R}^{n\times n}_{max}$ be the set of $n \times n$ matrices with coefficients in $\mathbb{R}_{max}$ with the following operations: The sum of matrices $A, B \in \mathbb{R}^{n\times n}_{max}$, 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 $A \in \mathbb{R}^{n\times l}_{max}, B \in \mathbb{R}^{l\times n}_{max}$, 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 $\mathcal{E} \in \mathbb{R}^{n\times n}_{max}$ denote the matrix with all its elements equal to ϵ and denote by $E \in \mathbb{R}^{n\times n}_{max}$ 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 $\Re^{n\times n}_{\max} = (\mathbb{R}^{n\times n}_{max}, \oplus, \otimes, \mathcal{E}, E)$ has the algebraic structure of a noncommutative idempotent semiring.*

Definition 10. *Let $A \in \mathbb{R}^{n\times n}_{max}$ 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 $A \in \mathbb{R}^{n\times n}_{max}$ 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 $\circ : \mathcal{G}(A) \times \mathcal{G}(A) \rightarrow \mathcal{G}(A)$ 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 $\mathcal{N} = \bigcup_{i \in \mathcal{N}} [i]$; $[i] = \{j \in \mathcal{N} : i\mathcal{C}j\}$.

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 $\widetilde{\mathcal{N}} = \{[i_1], [i_2], \dots [i_q]\}$ 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 $A \in \mathbb{R}^{n\times n}_{max}$ 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 $A \in \mathbb{R}^{n\times n}_{max}$ be a matrix. If $\mu \in R_{max}$ is a scalar and $v \in R^n_{max}$ 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: $\lambda = \max\limits_{p \in \mathcal{C}(A)} \frac{|p|_w}{|p|_1}$ 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 $A \in \mathbb{R}^{n\times n}_{max}$ 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 $G(A)$ $\lambda(A) = \max\limits_{p \in \mathcal{C}(A)} \frac{|p|_w}{|p|_1}$.*

Theorem 6. *Let $A \in \mathbb{R}^{n\times n}_{max}$ and $b \in \mathbb{R}^n_{max}$. 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 $A_m \in \mathbb{R}^{n\times n}_{max}$ for $0 \leq m \leq M$ and $x(m) \in \mathbb{R}^n_{max}$ for $-M \leq m \leq -1$; $M \geq 0$. Then, the recurrence equation: $x(k) = \bigoplus_{m=0}^{M} A_m \otimes x(k-m)$; $k \geq 0$ is called an Mth order recurrence equation.*

Theorem 7. *The Mth order recurrence equation, given by equation $x(k) = \bigoplus_{m=0}^{M} A_m \otimes x(k-m)$; $k \geq 0$, can be transformed into a first order recurrence equation $x(k+1) = A \otimes x(k)$; $k \geq 0$ 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: $x(k) = \bigoplus_{m=0}^{M} A_m \otimes x(k-m)$; $k \geq 0$ 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 $x(k) = \bigoplus_{m=0}^{M} A_m \otimes x(k-m)$; $k \geq 0$ can be expressed as: $\hat{x}(k+1) = \hat{A} \otimes \hat{x}(k)$; $k \geq 0$, 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 Predator-Prey Dynamical Systems

Consider the growth rate dynamics of a predator-prey system with TPN model depicted in (Fig.1). Where the events (transitions) that drive the system are: t: prays at threat , s: the predator starts attacking the prey, d: the predator departs. The places (that represent the states of the system) are: R: preys resting, P: the preys are in danger, B: the preys are being eaten, I: the predator is idle. The holding times associated to the places R and I are Cr and Cd respectively, (with $Cr > Cd$). The incidence matrix that represents the PN model is

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

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 t, the marking in P grows indefinitely. However, by taking $u = [k, k, k]; k > 0$, (with k unknown) we get that $A^T u \leq 0$. Therefore, the PN is stabilizable which implies that the TPN is

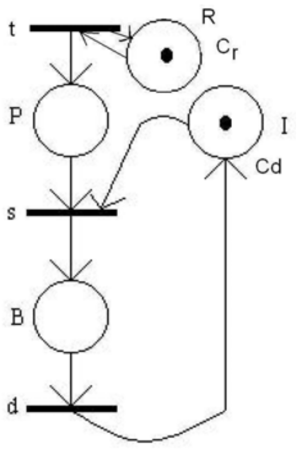


Fig. 1. Timed Petri net model

stable. Now, from the TPN model we obtain that matrix $\hat{A}$, which defines the standard autonomous equation, is equal to:

$$\hat{A} = \begin{pmatrix} Cr & \varepsilon & \varepsilon \\ Cr & \varepsilon & Cd \\ \varepsilon & \varepsilon & Cd \end{pmatrix}. \tag{15}$$

Therefore, $\lambda(A) = \max_{p \in \mathcal{C}(A)} \frac{|p|_w}{|p|_1} = \max\{Cr, Cd\} = Cr$. This tells that in order for the TPN to be stable and oscillate at a fixed known frequency, all the transitions must fire at the same speed as the preys arrive, which is consistent with the observed oscillatory behavior in real life.

6 Conclusions

The main contribution of the paper consists in combining Lyapunov theory with max-plus algebra to study the stability problem for predator-pray systems treated as discrete event dynamical systems modeled with timed Petri nets. The presented methodology is new and results to be innovative. The results obtained are consistent with how the predator-prey system behaves in real life.

References

1. Baccelli, F., Cohen, G., Olsder, G.J., Quadrat, J.P.: Synchronization and Linearity, Web-edition (2001)
2. Heidergott, B., Olsder, G.J., van der Woude, J.: Max Plus at Work. Princeton University Press, Princeton (2006)
3. Retchkiman, Z.: Stability theory for a class of dynamical systems modeled with Petri nets. International Journal of Hybrid Systems 4(1) (2005)
4. 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)
5. Haberman, R.: Mathematical Models in mechanical vibrations, population dynamics, and traffic flow. Prentice Hall, Englewood Cliffs (1977)

Study on the Local Search Ability of Particle Swarm Optimization

Yuanxia Shen[1,2,3] and Guoyin Wang[1,2]

[1] School of Information Science and Technology, Southwest Jiaotong University, Chengdu 600031, China
[2] Institute of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
[3] Department of Computer Science, Chongqing University of Arts and Sciences, Chongqing 402160, China
yuanxiashen@163.com, wanggy@cqupt.edu.cn

Abstract. Particle swarm optimization (PSO) has been shown to perform well on many optimization problems. However, the PSO algorithm often can not find the global optimum, even for unimodal functions. It is necessary to study the local search ability of PSO. The interval compression method and the probabilistic characteristic of the searching interval of particles are used to analyze the local search ability of PSO in this paper. The conclusion can be obtained that the local search ability of a particle is poor when the component of the global best position lies in between the component of the individual best position and the component of the current position of the particle. In order to improve the local search ability of PSO, a new learning strategy is presented to enhance the probability of exploitation of the global best position. The experimental results show that the modified PSO with the new learning strategy can improve solution accuracy.

Keywords: Particle swarm optimization, local search, interval compression.

1 Introduction

PSO is a population optimization method based on a simple simulation of bird flocking or fish schooling behavior [1]. Due to the fast convergence speed and easy implementation, it has been successfully applied in many areas such as neural network, image processing, power systems and mountain clustering.

In the standard PSO (SPSO), the search points are known as particles, each particle adjusts its velocity according to its own flight experience and the flight experience of other particles in the swarm in such a way that it accelerates towards positions that have had high objective (fitness) values in previous iterations. The position vector and velocity vector of the particle i in D-dimensional space can be indicated as $x_i=(x_{i1},\cdots, x_{id}, \cdots, x_{iD})$ and $v_i=(v_{i1},\cdots, v_{id}, \cdots ,v_{iD})$ respectively. The updating velocities and positions of the particles are calculated using the following two equations:

$$V_i(t+1) = wV_i(t) + c_1r_1(P_{bi}(t) - X_i(t)) + c_2r_2(G_b(t) - X_i(t)). \tag{1}$$

$$X_i(t+1) = X(t) + V_i(t+1). \tag{2}$$

where, P_{bi} is the personal best position achieved up to the current iteration for i-th particle (*pbest*); G_b is the global best position obtained so far by all particles (*gbest*); $w$is the inertia weight; c_1 and c_2 are positive constants known as acceleration coefficients; r_1 and r_2 are uniform random numbers in the range [0,1]. The first part of (1) represents the previous velocity, which provides the necessary momentum for particles to roam across the search space. The second part, known as the individual cognition, represents the personal thinking of each particle. The third part is known as the social cognition, which represents the collaborative effect of the particles.

Though SPSO has been shown to perform well on many optimization problems, it can easily get trapped in the local optima when solving complex problems. Therefore, avoiding the local optima has become one of the most important goals in PSO research, thereby deriving many variants. Shi and Eberhart [2] introduced inertia weight into PSO to balance the global exploration and the local exploitation; Ratnaweera et al. [3] developed PSO with time-varying acceleration coefficients to modify the local and the global search ability, and take advantage of the mutation to increase the diversity; Kennedy [4] constructed neighborhood topologies to restrict the information interaction between the particles ; Arumugam et al. [5] took advantage of extrapolation technique to update the current position without velocity equations of particles . Liang et al. [6] proposed a comprehensive learning strategy to preserve the diversity and discourage the premature convergence; Zhan et al. [7] presented the adaptive PSO that can improve global search ability by evaluating the evolution states of population distribution. The mentioned methods improved the performance of SPSO. But form the results of above methods, the improved PSO algorithms can not find the optimal solution for unimodal functions. SPSO needs try to avoid the local optima, but it does not mean that SPSO has the high local search ability. It is obvious the algorithm can not reach the optimum without the global exploration. But the local search ability is same important to improve accuracy of the algorithm. In this paper, the interval compression method and the probabilistic characteristic of the searching interval of particles are used to analyze the local search ability of SPSO. To increase the local search ability of particles, a new learning strategy is presented to enhance the probability of exploitation of the global best position.

The remainder of this paper is organized as follows. An analysis of the searching interval of PSO is introduced in section 2. The Modified PSO with new learning strategy is presented in section 3. Simulation experiment results on some benchmark optimization problems are discussed in section 4. Conclusions are drawn in section 5.

2 Analysis of the Searching Interval of SPSO

In the latter evolution of SPSO, the algorithm can find the better solution, but it often can not go on searching the global optimum. Much work has been done to analyze the reasons causing this case. The major reasons include diversity loss, the high speed of sharing information and fit parameters selection. Besides the mentioned reasons, the poor local search ability of SPSO may lead to can not find the global optimum or the local optimum. In this section, the interval compression method and the probabilistic characteristic of the searching interval of particles are used to analyze the searching interval of SPSO.

2.1 The Interval Compression Method

Each particle updates its position according to its own velocity, *pbest* and *gbest*. If the *pbest* and *gbest* of the particle can not be updated for a while, then the velocity of the particle should become small, and the particle should circle around *pbest* and *gbest*. In this case, the fit searching interval is important for the particle to find the better position.

From equation (1), a particle searches the new position by tracking the each dimension of *pbest* and *gbest*. In other words, a particle makes use of each component of *pbest* and *gbest* to search in one dimension. Therefore, it is a question how to determine a fit searching interval in one dimension. The interval compression method is introduced to discuss how to select the searching interval.

Assume that $f(x)$ is unimodal in the interval $[a, b]$, where exists a minimum x^*. a_1 and a_2 are any two points on this close interval. If $f(a_1)<f(a_2)$, then x^*should lie in the left of a_2, $x^* \in [a, a_2]$. Therefore, the searching interval $[a, b]$ can be compressed into a new searching interval $[a, a_2]$. If $f(a_1) \geq f(a_2)$, then x^*should lie in the right of a_1, $x^* \in [a_1, b]$. Therefore, the searching interval $[a, b]$ can be compressed into a new searching interval $[a_1, b]$. The interval compression method can be used to determine searching area to improve the precision of SPSO.

2.2 The Searching Interval of One Dimension for SPSO

The SPSO model can be reduced to the one-dimensional case. The following discussion needs a precondition that the position of a particle is nearby around an optimum solution and the *pbest* and *gbest* can not be updated for a while. The hypothesis is that the p_b and g_b locate in the interval $[a, b]$ of quadratic function and $p_b \neq g_b$, where exists a global optimum or local optimum. The p_b and g_b are the component of *pbest* and *gbest*. The above hypothesis is reasonable because that any complex function can be locally approximated by a quadratic Taylor polynomial, which makes a complex function reduce a quadratic function. The analysis of the searching interval for quadratic function can extend other functions. Assume that the component of the current position of the particle x lie in the left of the p_b and g_b, where exists two cases: (1) $(p_b-x)>(g_b-x)$; (2) $(p_b-x)<(g_b-x)$, as shown in Fig.1.

For $(p_b-x)>(g_b-x)$, the searching interval can be compressed into the new interval $[p_g, p_b]$ or $[a, p_g]$ by the compression method of the searching interval. Then the probabilistic characteristic is analyzed for different intervals. The searching interval dependents on the second component and third component of equation (1) because the velocity of the particle is small in context of the above hypothesis. Let $c_1r_1(p_b-x)+c_2r_2(g_b-x)=U$, where the value range of U is the searching interval of one dimention for a particle. The graphical method is used to analyze the probability of the value range of U. Set $c_1= c_2=2$, and let $c_1r_1(p_b-x)=Z$, the value of Z ranges from 0 to $2(p_b-x)$;Let $c_2r_2(g_b-x)=Y$, the value of Y ranges from 0 to $2(g_b-x)$; Then $Z+Y=U$, the value of U is the interception of Y-coordinate axis. When x lies in the left of the p_b and g_b and $(p_b -x)>(g_b-x)$, the value range of U is form 0 to $2(p_b+g_b-2x)$, i.e. point A to point G,

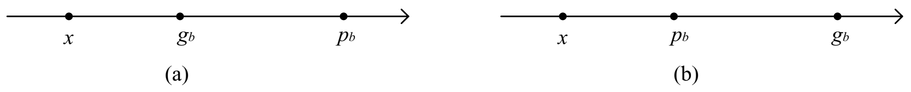


Fig. 1. The position of the particle

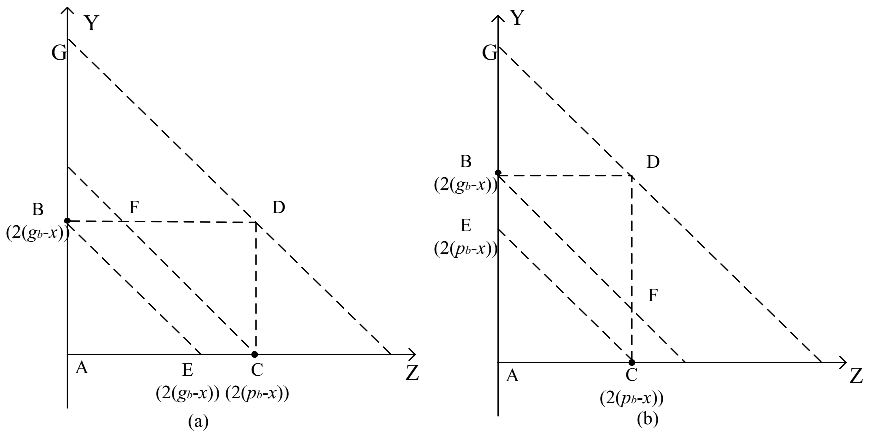


Fig. 2. The searching interval of SPSO

as shown in Fig.2(a). But the probability of the different ranges of U is different. From Fig.2(a), when $2(p_b-x) \le U \le 2(g_b-x)$, the probability of U lied in this interval is same and lager than that of U lied in other interval. If $g_b \ge 2x$, then g_b can lie in $[2(p_b - x), 2(g_b - x)]$. When $0 \le U \le 2(p_b-x)$, the probability of U lied in this interval increases with increase of the value of U, while $2(g_b-x) \le U \le 2(p_b+g_b-2x)$, the probability of U lied in this interval decreases with increase of the value of U. From probability analysis, the particle locates the right of p_b with lager probability, while the particle locates in interval $[x, g_b]$ with very small probability. This case is unfavorable to the local search and may lead to lose opportunity to find the better solution.

For $(p_b-x)<(g_b-x)$, the searching interval should be the new interval $[p_b, p_g]$ or $[p_g, b]$ by the interval compression method. The value range of U is form 0 to $2(p_b+g_b\text{-}2x)$, i.e. point A to point G, as shown in Fig.2(b). When $2(p_b-x) \le U \le 2(g_b-x)$, the probability of U lied in this interval is same and lager than that of U lied in other interval. If $p_b \le (0.5g_b+x)$, then g_b can lie in $[2(p_b -x), 2(g_b -x)]$. From probability analysis, the particle locates the right of g_b with lager probability, which makes full use of *gbest*.

When the component of the current position of the particle x lies in the right of the p_b and g_b, the conclusion can be obtained that the particle can not make full use of *gbest* when $(p_b-x)>(g_b-x)$. Therefore, the conclusion can be obtained that the local search ability of particles is poor when the component of the global best position lies in between the component of the individual best position and the component of the current position of the particle through above analysis. When the component of the current position of the particle x lies between p_b and g_b, which means that the p_b and g_b can not locate in the interval $[a, b]$ of quadratic function. This paper does not discuss this case.

3 Modified PSO (MPSO)

Form above analysis, the searching mechanism of SPSO is unfavorable to the local search when g_b lies between x and p_b. Then the probability of the position of the particle located in [g_b x] or [x g_b] should be increased to improve the local search ability of particle.

The modified PSO (MPSO) is presented, where particles adopt the modification of the learning example to improve the probability of exploitation of g_b. The updating velocities of particles are calculated using the following equations in MPSO.

$$V_i(t+1) = wV_i(t) + c_1r_1(P_{bi}(t) - X_i(t)) + c_2r_2(G_b(t) - X_i(t))$$
$$\begin{cases} P_{bi}(t) = P_{bi}(t) + z(G_b(t) - P_{bi}(t)) \\ G_b(t) = G_b(t) + z(G_b(t) - P_{bi}(t)) \end{cases} if\,(P_{bi}(t) - G_b(t))(X_{bi}(t) - G_b(t)) < 0. \tag{3}$$

Where z is learning parameter, $z \in [0\ 1]$. The increase of the value of z can enhance the probability of exploitation of g_b. The searching interval of MPSO is analyzed for x lied in the left of the p_b and g_b and $(p_b-x)>(g_b-x)$,as shown in Fig.1(a). In MPSO, the value range of U is form 0 to $4(g_b-x)$, i.e. point A to point G, as shown in Fig.3, where let z=0.5. When $(3g_b-p_b-2x) \le U \le (g_b+p_b-2x)$, the probability of U lied in this interval is same and lager than that of U lied in other interval. If $p_b \ge 2x$, then g_b can lie in [$3g_b-p_b-2x$, g_b+p_b-2x]. In SPSO, g_b can lie in [$2(p_b-x)$,$2(g_b-x)$] when $g_b \ge 2x$. The probability of $p_b \ge 2x$ is larger than that of $g_b \ge 2x$ because $(p_b-x)>(g_b-x)$. Therefore, MPSO improves the probability of exploitation of g_b and enhances the local search ability of particles.

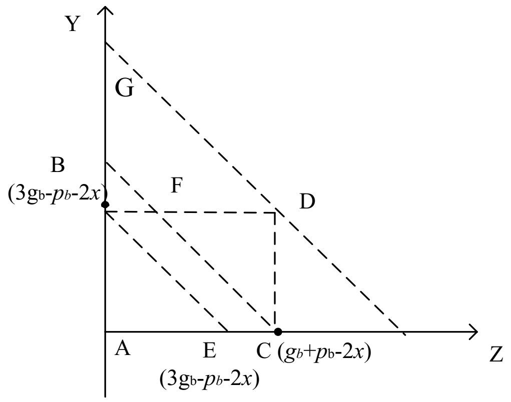


Fig. 3. The searching interval of MPSO

4 Experiments

In order to test the effectiveness of MPSO, four famous benchmark functions were optimized by PSO with linearly decreased inertial weight (PSO-LDIW), PSO with time-varying acceleration coefficients (PSO-TVAC) and MPSO.

4.1 Test Functions

All test functions have to be minimized. Functions f_1, f_2 and f_3 are unimodal, while f_4 is multimodal function with the global optima enclosed by many local optima. The properties and the formulas of functions are presented below.
Sphere's function

$$f_1(x)=\sum_{i=1}^{D} x_i^2 \text{ ,} x\in[-100,100] \text{ , } D=30, \min(f_1)=f_1(0,0,\ldots,0)=0.$$

Quadric's function

$$f_2(x)=\sum_{j=1}^{D}(\sum_{k=1}^{j} x_k)^2 \quad x\in[-100,100], D=30, \min(f_2)=f_2(0,0,\ldots,0)=0.$$

Rosenbrock's function

$$f_3(x)=\sum_{i=1}^{D}100(x_{i+1}-x_i^2)^2+(x_i-1)^2 \text{, } x\in[-10,10], D=30, \min(f_3)=f_3(1,1,\ldots,1)=0.$$

Griewank's function

$$f_4(x)=\sum_{i=1}^{D}\frac{x_i^2}{4000}-\prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})+1 \text{, } x\in[-600,600], D=30, \min(f_4)=f_4(0,0,\ldots,0)=0.$$

4.2 Parameters Setting for PSO Algorithms

Parameters setting for PSO-LDIW and PSO-TVAC come form Refs. [2], [3]. In PSO-LDIW, PSO-TVAC and MPSO the inertia weight w is decreased linearly from 0.9 to 0.3; Let $c_1= c_2=2$ for PSO-LDIW and MPSO, while the cognitive coefficient decreases linearly from 2.5 to 0.5, while the social coefficient increased linearly from 0.5 to 2.5 in PSO-TVAC. The number of iterations is set as 100000. The population size is set 20. All results reported are minimums, averages and standard deviations over 20 simulations.

4.3 Experiment Results and Discussions

Table 1 presents the means and variances of the 20 runs of the three algorithms on the four test functions and other improved PSO algorithms from Ref.[6],[7]. From Table 1, MPSO achieved better results than the PSO-LDIW and PSO-TVAC for four functions and can find global optimum for Sphere's function. Compared with other improved PSO in Ref.[6],[7], MPSO can increase the precision of solutions for unimodal functions, but can not effectively improve the quality of solutions for multimodal functions, which have many local optima for multimodal functions. Therefore, it is not enough for searching global optimum to only enhance the local search ability of particles.

Fig.4 presents the convergence characteristics in terms of the mean best fitness value of each algorithm for each function. As can been seen form Fig.4, MPSO converges slower than PSO-LDIW and PSO-TVAC because of the high local search ability of MPSO.

Table 1. The results of different PSO algorithms

Functions		PSO-LDIW	PSO-TVAC	MPSO	APSO(Ref.[7])	CLPSO(Ref.[6])
f_1	Mean	1.36e-320	1.51e-35	0	1.45e-150	1.89e-19
	Std.	3.45e-300	2.06e-34	0	5.73e-150	1.73e-14
f_2	Mean	3.24e-60	2.01e-8	8.56e-78	1.00e-10	3.95e+2
	Std.	8.46e-62	2.51e-7	5.58e-79	2.13e-10	1.42e+2
f_3	Mean	2.13e+1	3.10e+1	2.34e-6	2.84e-0	2.10e+1
	Std.	5.02e+1	2.57e+1	6.41e-5	3.27e-0	2.98e+0
f_4	Mean	8.42e-1	5.03e-2	8.23e-3	1.67e-2	6.45e-13
	Std.	4.12e-1	3.72e-2	7.19e-2	2.41e-2	2.07e-12

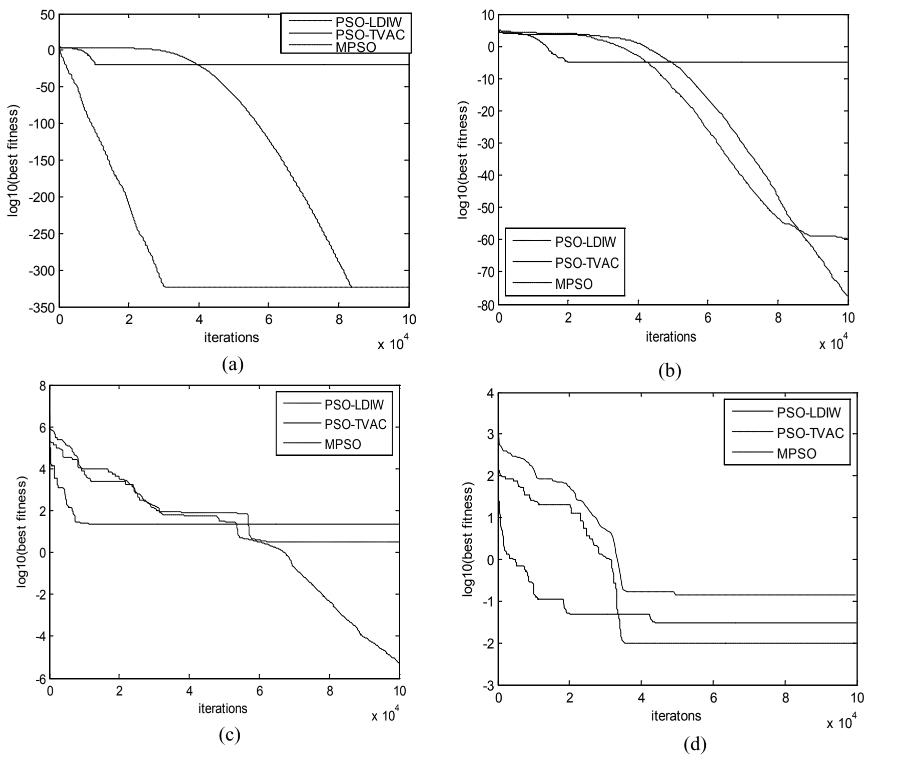


Fig. 4. The convergence curve of test function for different function. (a) Sphere function. (b) Quadric function (c) Bosenbrock function (d) Griewank function.

5 Conclusion

This paper makes an analysis for the local search ability of SPSO. The conclusion can be obtained that the local search ability of the particle is poor when the component of the global best position lies in between the component of the individual best position and the component of the current position of the particle through the analysis. A new learning strategy is presented to improve the local search ability of particles in this paper. The experimental results show that MPSO can improve the accuracy of solution, especially for unimodal functions. It is obvious that the high local search ability is not enough for searching global optimum. Future work will enhance exploration ability of particles for MPSO.

Acknowledgments. This paper is supported by National Natural Science Foundation of China under Grant No.60773113, Natural Science Foundation of Chongqing under Grants No.2008BA2017 and No.2008BA2041, Chongqing Key Lab of Computer Network and Communication Technology No.CY-CNCL-2009-03.

References

1. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proceeding of International Conference on Neural Networks, pp. 1942–1948. IEEE Press, Perth (1995)
2. Shi, Y.H., Eberhart, R.C.: A Modified Particle Swarm Optimizer. In: Proceeding of IEEE Congress on evolutionary Computation, pp. 69–73. IEEE Press, Piscataway (1998)
3. Asanga, R., Saman, K.H., Harry, C.W.: Self-organizing Hierarchical Particle Swarm Optimizer with Time-varying Acceleration Coefficients. IEEE Transaction on evolutionary computation 8(3), 240–255 (2004)
4. Kennedy, J.: Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance. In: Proceeding of IEEE Congress on evolutionary Computation, pp. 1931–1938. IEEE Press, Washington (1999)
5. Arumugam, M.S., Rao, M.V.C., Tan Alan, W.C.: A Novel and Effective Particle Swarm Optimization Like Algorithm with Extrapolation Technique. Applied soft computing 9, 308–320 (2009)
6. Liang, J.J., Qin, A.K., Suganthan, P.N.: Comprehensive Learning Particle Swarm Optimizer for Global Optimization of Multimodal Functions. IEEE Transaction on evolutionary computation 10(3), 281–295 (2006)
7. Zhan, Z.H., Zhang, J., Li, Y.: Adaptive Particle Swarm Optimization. IEEE Transaction on systems, man and cybernetics-part B 39(6), 1362–1381 (2009)

The Performance Measurement of a Canonical Particle Swarm Optimizer with Diversive Curiosity

Hong Zhang[1] and Jie Zhang[2]

[1] Department of Brain Science and Engineering, Kyushu Institute of Technology
2-4 Hibikino, Wakamatsu, Kitakyushu 808-0196, Japan
zhang@brain.kyutech.ac.jp

[2] Wuxi Bowen Software Technology Co., Ltd
Room 601-603, Building A, No. 5 Xinhua Road, WND, Wuxi, Jiangsu, China
leozhang@126.com

Abstract. For improving the search performance of a canonical particle swarm optimizer (CPSO), we propose a newly canonical particle swarm optimizer with diversive curiosity (CPSO/DC). A crucial idea here is to introduce diversive curiosity into the CPSO to comprehensively manage the trade-off between exploitation and exploration for alleviating stagnation. To demonstrate the effectiveness of the proposed method, computer experiments on a suite of five-dimensional benchmark problems are carried out. We investigate the characteristics of the CPSO/DC, and compare the search performance with other methods. The obtained results indicate that the search performance of the CPSO/DC is superior to that by EPSO, ECPSO and RGA/E, but is inferior to that by PSO/DC for the *Griewank* and *Rastrigin* problems.

Keywords: canonical particle swarm optimizer, evolutionary particle swarm optimization, real-coded genetic algorithm, exploitation and exploration, model selection, specific and diversive curiosity.

1 Introduction

As generally known, canonical particle swarm optimizer (CPSO) [2] is a constriction version of particle swarm optimization (PSO) [4,7]. Since it is based on a rigorous analysis of the dynamics of a simplified particle swarm optimizer, the CPSO has broad implications for optimization and applications [9,11,12].

Despite the good convergence in the CPSO search [8], it is likely to get stuck in local minimum for dealing with multimodal optimization problems. For alleviating the phenomenon, Zhang et al. proposed an evolutionary canonical particle swarm optimizer (ECPSO) [16] which estimates appropriate values of parameters in the CPSO by genetic computation. Although the ECPSO successfully provides a framework of model selection for the CPSO, and improves its search performance, stagnation cannot be exhaustively prevented.

To overcome the above difficulty, this paper proposes a newly canonical particle swarm optimizer with diversive curiosity (CPSO/DC). This is an extension of the existing method, particle swarm optimization with diversive curiosity (PSO/DC) [15]. It has the following outstanding features: (1) Introducing diversive curiosity into the CPSO search for managing the trade-off between exploitation and exploration, (2) Adopting the optimized CPSO to search for ensuring higher search performance. Owing to the combination of strengthening swarm intelligence and meta-optimization technique, effective performance improvement for the CPSO and the ECPSO even is fulfilled.

2 Basic Algorithms

Let the search space be N-dimensional, $\mathbf{S} \in \Re^N$, the number of particles of a swarm be P, and the position and velocity of ith particle be $\boldsymbol{x}^i = (x_1^i, x_2^i, \cdots, x_N^i)^T$ and $\boldsymbol{v}^i = (v_1^i, v_2^i, \cdots, v_N^i)^T$, respectively.

The CPSO: The update rule of the CPSO is formulated as

$$\begin{cases} \boldsymbol{x}_{k+1}^i = \boldsymbol{x}_k^i + \boldsymbol{v}_{k+1}^i \\ \boldsymbol{v}_{k+1}^i = \chi\big(\boldsymbol{v}_k^i + c_1\boldsymbol{r}_1 \otimes (\boldsymbol{p}_k^i - \boldsymbol{x}_k^i) + c_2\boldsymbol{r}_2 \otimes (\boldsymbol{q}_k - \boldsymbol{x}_k^i)\big) \end{cases} \tag{1}$$

where χ is a constriction coefficient, c_1 is a coefficient for individual confidence, and c_2 is a coefficient for swarm confidence. $\boldsymbol{r}_1, \boldsymbol{r}_2 \in \Re^N$ are two random vectors, each component of which is uniformly distributed on $[0, 1]$, and $\otimes$ is an element-by-element vector multiplication operator. $\boldsymbol{p}_k^i$ $(= arg \max_{j=1,\cdots,k} \{g(\boldsymbol{x}_j^i)\}$, where $g(\boldsymbol{x}_k^i)$ is the fitness value of ith particle at time-step k) is the local best position of ith particle up to now, $\boldsymbol{q}_k (= arg \max_{i=1,2,\cdots} \{g(\boldsymbol{p}_k^i)\})$ is the global best position among the whole particles at time-step k.

According to Clerc's constriction method, c_1 and c_2 are set to 2.05, and χ is approximately 0.7298 in the CPSO.

The ECPSO: It is a meta-optimization method which optimizes the values of parameters in the CPSO. The ECPSO is composed of two loops: an inner loop running the CPSO and an outer loop running a real-coded genetic algorithm with elitism strategy (RGA/E) [16]. The former solves a given real-coded optimization problem with a selected set of parameter values in the CPSO. The latter simulates the survival of the fittest among individuals (individual being a set of parameter values) over generations for finding the best parameter values.

The genetic operations adopted in the RGA/E are roulette wheel selection, BLX-α crossover, random mutation, elitism strategy, non-redundant search, rank operation, and mixing operation for efficiently obtaining the optimized CPSO to a given optimization problem.

A criterion value which evaluates the search performance of each CPSO in estimation is under stochastic fluctuation. To reduce stochastic fluctuation, the following temporally cumulative fitness function is used.

$$F(\chi, c_1, c_2) = \sum_{k=1}^{K} g(\boldsymbol{q}_k)\big|_{\chi, c_1, c_2} \tag{2}$$

where K is the maximum number in the CPSO runs.

The CPSO/DC: Curiosity is a concept in psychology seeking of stimulus/ sensation in humans and animals [13]. Berlyne categorized it into two types: diversive and specific [1]. As to the former, Loewenstein insisted that "diversive curiosity occupies a critical position at the crossroad of cognition and motivation" [10]. According to the assumption that "cognition" is the act of exploitation, and "motivation" is the intention to exploration, the following internal indicator is proposed for presenting the above two patterns [14].

$$y_k(L, \varepsilon) = max\Big(\varepsilon - \sum_{l=1}^{L} \frac{\big|g(\boldsymbol{q}_k) - g(\boldsymbol{q}_{k-l})\big|}{L},\ 0\Big) \tag{3}$$

where L is duration of judgment, ε is the positive tolerance parameter.

It is clear from Fig. 1 that based on the output of the internal indicator, y_k, the behaviors of the CPSO search are switched from one situation (exploitation) to another (exploration). The active behavior of continually finding solutions, which directly expresses the intrinsic and spontaneous property of a particle swarm itself, is interpreted as a mechanism of diversive curiosity.

It is to be noted that the repeat of initialization decided by the signal d_k in Fig. 1 is a mere represented form which concretely realizes diversive curiosity to explore. By reducing boredom behavior and managing the trade-off between exploitation and exploration in the CPSO search, the efficiency finding the best solution will be drastically improved in a fixed period, K.

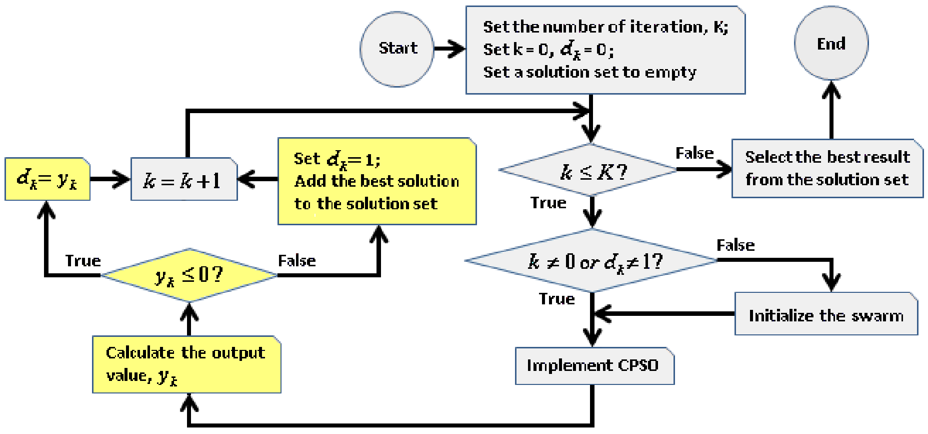

Start
Set the number of iteration, K;
Set k = 0, d_k = 0;
Set a solution set to empty
End
$d_k = y_k$
$k = k+1$
Set $d_k = 1$;
Add the best solution to the solution set
$k \le K$?
False
True
Select the best result from the solution set
True
$y_k \le 0$?
False
$k \ne 0$ or $d_k \ne 1$?
False
True
Initialize the swarm
Calculate the output value, y_k
Implement CPSO

Fig. 1. A flowchart of the CPSO/DC

In addition, the CPSO optimized by the ECPSO as CPSO*, which is used in the CPSO/DC for ensuring higher search performance.

3 Computer Experiments

To facilitate comparison and analysis of the performance of the proposed method, we use a suite of benchmark problems in Table 1 [17]. And Table 2 shows the major parameters emplored for the next experiments.

Therefore, Table 3 gives the resulting values of parameters in the CPSO by the ECPSO for each 5-dimensional (5D) benchmark problem. They are quite different from the parameter values in the original CPSO. These optimized CPSO as CPSO* are used in the CPSO*/DC for handling the given benchmark problems.

Performance Measurement: Fig. 2 illustrates the resulting performance indexes of the CPSO*/DC. The following characteristics of the CPSO*/DC with tuning the parameters, L and ε, are observed.

- The number of re-initializations increases with increment of the parameter ε, and decrement of the parameter L, for each problem.
- To obtain higher search performance of the CPSO*/DC, the recommended range of $L^o_{Sp} \in (30, \cdots, 90)$ and $\varepsilon^o_{Sp} \in (10^{-6}, \cdots, 10^{-2})$ for the *Sphere* problem; $L^o_{Gr} \in (30, 40)$ and $\varepsilon^o_{Gr} \in (10^{-3}, 10^{-2})$ for the *Griewank* problem;

Table 1. Functions and criteria in the suite of benchmark problems. The search space for each benchmark problem is limited to $\mathbf{S} \in (-5.12, 5.12)^N$.

Function	Equation	Criterion
Sphere	$f_{Sp}(\boldsymbol{x}) = \sum_{d=1}^{N} x_d^2$	$g_{Sp}(\boldsymbol{x}) = \frac{1}{f_{Sp}(\boldsymbol{x}) + 1}$
Griewank	$f_{Gr}(\boldsymbol{x}) = \frac{1}{4000} \sum_{d=1}^{N} x_d^2 - \prod_{d=1}^{N} cos\left(\frac{x_d}{\sqrt{d}}\right) + 1$	$g_{Gr}(\boldsymbol{x}) = \frac{1}{f_{Gr}(\boldsymbol{x}) + 1}$
Rastrigin	$f_{Ra}(\boldsymbol{x}) = \sum_{d=1}^{N} \left(x_d^2 - 10\, cos\,(2\pi x_d) + 10\right)$	$g_{Ra}(\boldsymbol{x}) = \frac{1}{f_{Ra}(\boldsymbol{x}) + 1}$
Rosenbrock	$f_{Ro}(\boldsymbol{x}) = \sum_{d=1}^{N-1} \left[\left(100(x_{d+1} - x_d^2)\right)^2 + (x_d - 1)^2\right]$	$g_{Ro}(\boldsymbol{x}) = \frac{1}{f_{Ro}(\boldsymbol{x}) + 1}$

Table 2. The major parameters used in the ECPSO and the CPSO/DC

Parameters	Values	Parameters	Values
the number of individuals, M	10	the number of generation, G	20
the number of iterations, K	400	probability of BLX-2.0 crossover, p_c	0.5
the number of particles, P	10	probability of random mutation, p_m	0.5

Table 3. The resulting values of parameters in the CPSO and the appearance frequency for each 5D benchmark problem with 20 trials

Problem	Parameters			Fitness	Freq.
	χ	c_1	c_2	F	
Sphere	0.6528±0.1760	4.3175±0.7283	2.1319±1.1178	397.3±0.689	95.0%
Griewank	0.7354±0.1723	1.6764±1.3422	1.1250±0.7046	397.2±0.812	100.0%
Rastrigin	0.8158±0.2396	1.6761±1.1867	1.5339±0.7950	158.0±26.20	95.0%
Rocenbrock	0.9397±0.1333	0.9694±0.4533	1.7525±0.4529	365.8±19.86	90.0%

$L^o_{Ra} \simeq 60$ and $\varepsilon^o_{Ra} \simeq 10^{-3}$ for the *Rastrigin* problem; $L^o_{Ro} \simeq 80$ and $\varepsilon^o_{Ro} \simeq 10^{-3}$ for the *Rosenbrock* problem are available, respectively.

– The distributions of the average of criterion values in Fig.2(b.Ra) and (b.Ro) basically reflect the fundamental finding, "the zone of curiosity" and "the zone of anxiety", in psychology [3].

Fig. 3 shows the resulting difference, $\Delta = \bar{g}^* - \bar{g}$, of the average of criterion values between the CPSO*/DC and the CPSO/DC for each problem. We can see that the results on their differences are almost positive for the *Rastrigin* and *Rocenbrock* problems. This indicates that the optimized CPSO play an important role in efficiently solving these problems. Nevertheless, if the given problem is such simple or complex as the *Sphere* or *Rastrigin* problem, the effect of the CPSO*/DC is little and unremarkable by the limits of search ability.

Comparison with Other Methods: Table 4 gives the experimental results of implementing the CPSO, CPSO*, CPSO*/DC, PSO*/DC, and RGA/E with 20 trials. It indicates that the search performance of the CPSO*/DC is superior to that by the CPSO or the CPSO* based on comparison with the average of criterion values, and diversive curiosity plays an essential role in finding an optimal solution or near-optimal solutions.

Discussions: The search performance of the CPSO*/DC is superior to that by the RGA/E (except for the *Rastrigin* problem). However, why the search performance of the RGA/E is so better than that by the CPSO*/DC for the *Rastrigin* problem, the reason directly correlates with the intrinsic convergence of the CPSO and the CPSO* even specially for dealing with multimodal optimization problems.

Table 4. The mean and standard deviation of criterion values for each benchmark problem with 20 trials. The values in bold signify the best results for each problem.

Problem	CPSO	CPSO*	CPSO*/DC	PSO*/DC	RGA/E
Sphere	**1.0000**±0.000	**1.0000**±0.000	**1.0000**±0.000	**1.0000**±0.000	0.9990±0.0005
Griewank	0.8688±0.0916	0.9901±0.0050	0.9959±0.0048	**1.0000**±0.000	0.9452±0.0784
Rastrigin	0.1828±0.1154	0.2710±0.1300	0.2926±0.3402	**1.0000**±0.000	0.9064±0.2257
Rosenbrock	0.6206±0.2583	0.7008±0.3557	**0.8076**±0.283	0.7841±0.1471	0.3898±0.2273

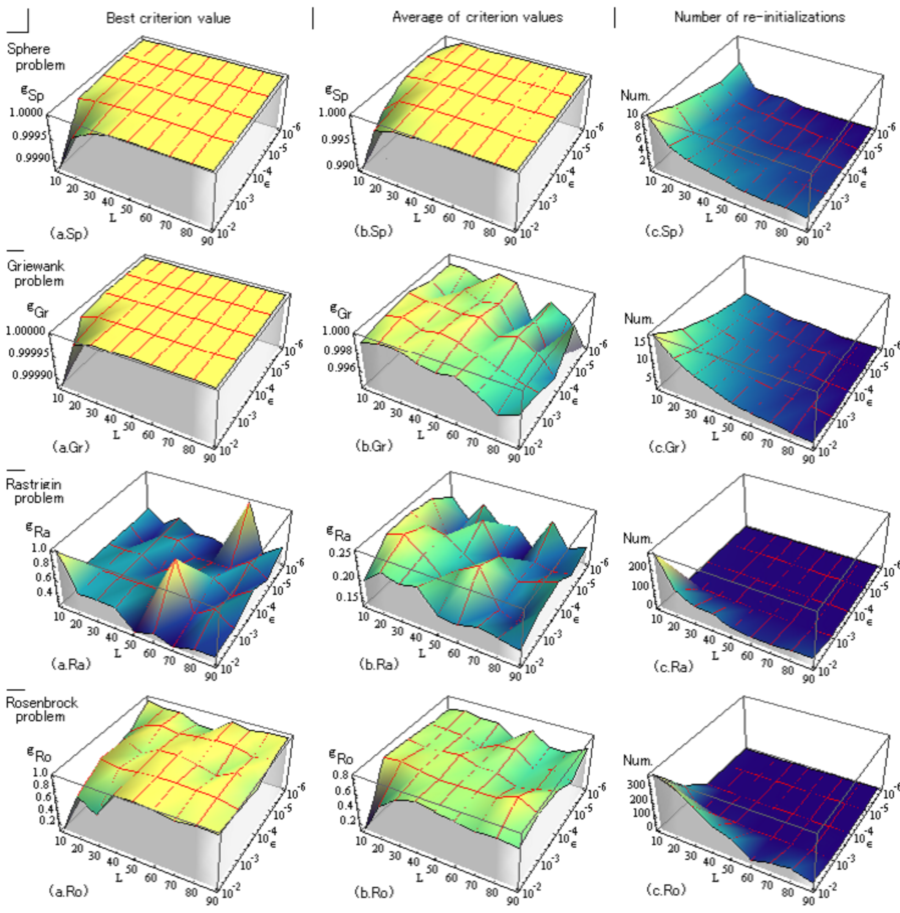


Fig. 2. The search performance of the CPSO*/DC for each problem with 20 trials. (a) Best criterion value, (b) Average of criterion values, (c) Number of re-initializations.

On the other hand, even though diversive curiosity takes an active part in search, there is not any change in the intrinsic character and search capacity of the CPSO itself. Namely, it just applies diversive curiosity to the CPSO search for alleviating stagnation in optimization without other technical supports such as hybrid genetic algorithm and particle swarm optimization [6] which implements the GA and the PSO to explore owing to a mixed operation. This is the shortcoming of the CPSO/DC in search strategy.

Comparison with the results of the PSO*/DC in Table 4 [14,15], we can see that the search performance of the CPSO*/DC is better than that by the PSO*/DC for the *Rosenbrock* problem, but is worse for the *Griewank* and *Rastrigin* problems. Accordingly, it is also verified that both of the PSO and the CPSO have the property of problem-dependence in search.

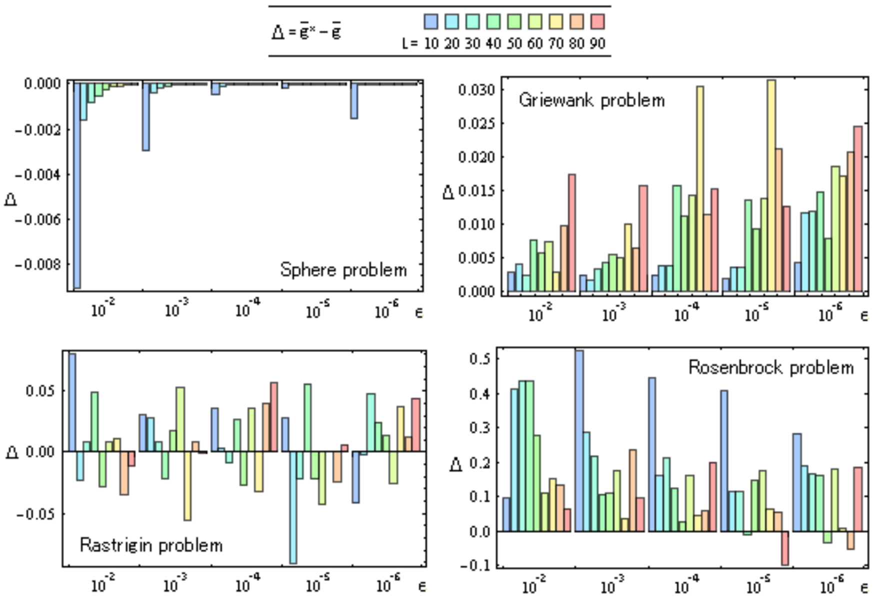


Fig. 3. The average difference of criterion values between the CPSO*/DC and the CPSO/DC with tuning the parameters, L and ε, for each problem

4 Conclusions

A new canonical particle swarm optimizer with diversive curiosity has been proposed in this paper. Owing to the function of the used internal indicator, it constantly makes a particle swarm active in search for efficiently solving a given optimization problem. Theoretically, the CPSO/DC has good capability which alleviates stagnation by comprehensively managing the trade-off between exploitation and exploration.

Applications of the CPSO/DC to a suite of 5D benchmark problems well demonstrated its effectiveness. The experimental results verified that the combination of strengthening swarm intelligence and meta-optimization technique is successful and effective manner for acquiring more efficiency to optimization search.

It is left for further study to apply the CPSO/DC to complex problems in the real-world and dynamic environments, and to apply the mechanism of diversive curiosity to cooperative particle swarm optimization for developing the swarm intelligence [5].

Acknowledgment. This research was partially supported by Grant-in-Aid Scientific Research(C)(22500132) from the Ministry of Education, Culture, Sports, Science and Technology, Japan.

References

1. Berlyne, D.: Conflict, Arousal, and Curiosity. McGraw-Hill Book Co., New York (1960)
2. Clerc, M., Kennedy, J.: The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation 6(1), 58–73 (2000)
3. Day, H.: Curiosity and the Interested Explorer. Performance and Instruction 21(4), 19–22 (1982)
4. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, pp. 39–43 (1995)
5. El-Abd, M., Kamel, M.S.: A Taxonomy of Cooperative Particle Swarm Optimizers. International Journal of Computational Intelligence Research 4(2), 137–144 (2008)
6. Juang, C.-F.: A Hybrid of Genetic Algorithm and Particle Swarm Optimization for Recurrent Network Design. IEEE Transactions on Systems, Man and Cybernetics Part B 34(2), 997–1006 (2004)
7. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of the 1995 IEEE International Conference on Neural Networks, Piscataway, New Jersey, USA, pp. 1942–1948 (1995)
8. Kennedy, J.: In Search of the Essential Particle Swarm. In: Proceedings of the 2006 IEEE Congress on Evolutionary Computation, Vancouver, BC, Canada, pp. 6158–6165 (2006)
9. Lane, J., Engelbrecht, A., Gain, J.: Particle Swarm Optimization with Spatially Meaningful Neighbours. In: Proceedings of Swarm Intelligence Symposium (SIS 2008), St. Louis, MO, USA, pp. 1–8 (2008)
10. Loewenstein, G.: The Psychology of Curiosity: A Review and Reinterpretation. Psychological Bulletin 116(1), 75–98 (1994)
11. Poli, R.: Analysis of the Publications on the Applications of Particle Swarm Optimisation. Journal of Artificial Evolution and Applications 2008(1), 1–10 (2008)
12. Reyes-Sierra, M., Coello, C.A.C.: Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art. International Journal of Computational Intelligence Research 2(3), 287–308 (2006)
13. Wohlwill, J.F.: A Conceptual Analysis of Exploratory Behavior in Advances in Intrinsic Motivation and Aesthetics. Plenum Press, New York (1981)
14. Zhang, H., Ishikawa, M.: Characterization of particle swarm optimization with diversive curiosity. Journal of Neural Computing & Applications, 409–415 (2009)
15. Zhang, H., Ishikawa, M.: Particle Swarm Optimization with Diversive Curiosity and Its Identification. In: Ao, S., et al. (eds.) Trends in Communication Technologies and Engineering Science. Lecture Notes in Electrical Engineering, vol. 33, pp. 335–349. Springer, Netherlands (2009)
16. Zhang, H., Ishikawa, M.: The performance verification of an evolutionary canonical particle swarm optimizer. Neural Networks 23(4), 510–516 (2010)
17. http://www.ntu.edu.sg/home/epnsugan/index_files/CEC-05/Tech-Report-May-30-05.pdf

Mechanism and Convergence of Bee-Swarm Genetic Algorithm

Di Wu[1,*], Rongyi Cui[2], Changrong Li[1], and Guangjun Song[1]

[1] College of Computer and Control Engineering, Qiqihaer University, 161006 Qiqihaer Heilongjiang, China
[2] College of Computer Engineering, Yanbian University, 133002 Yanji Jilin, China
wudiqqhru@tom.com

Abstract. Bee-Swarm genetic algorithm based on reproducing of swarm is a novel improved genetic algorithm. Comparing to GA, there are two populations, one for global search, and another for local search. Only best one can crossover. The genetic operators include order crossover operator, adaptive mutation operator and restrain operator. The simulated annealing is also introduced to help local optimization. The method sufficiently takes the advantage of genetic algorithm such as group search and global convergence, and quick parallel search can efficiently overcome the problems of local optimization. Theoretically, the capability of finding the global optimum is proved, and a necessary and sufficient condition is obtained namely. The convergence and effective of BSGA is proved by Markov chain and genetic mechanism. Finally, several testing experiments show that the Bee-Swarm genetic algorithm is good.

Keywords: absolute mating, adaptive crossover, simulated annealing algorithm, effective.

1 Introduction

Genetic algorithm based on "evolution" and "genetics" biology theory is proposed by Holland from United States in 1970s. According to the accumulation of knowledge, it is an effective non-numerical parallel optimization method, so it has been widely used in various fields. However, the genetic algorithm still has premature convergence and slow convergence shortcomings, causing by the contradiction between population diversity and pressure of selection. So some improved genetic algorithms combine with other algorithms like chaos theory[1], simulated annealing algorithm[2,3] and fuzzy theory[4] to deal with uncertain information. Others simulate the natural phenomena to improve the performance, like gender-specific genetic algorithm[5,6], monkey king genetic algorithm[7] and niche genetic algorithm[8] also has achieved good results.

This paper proposed the bee-swarm genetic algorithm based on reproducing of swarm and combining with annealing algorithm[9,10]. The BSGA is introduced briefly in

[*] Sponsor: 2009 post-graduate research and innovation fund of Heilongjiang province(YJSCX2009-102HLJ).

second part, the convergence and mechanism are analyzed in third part, and the simulation results about the optimization of classic function and knapsack problem are shown in the last part.

2 BSGA

The BSGA population is composed of queen bee(one number), drones(N numbers) and worker bees(M numbers), the main operators of BSGA include the absolute mating right between queen bee and drones, the simulated suppression between queen bee and worker bees, local optimization of queen bee based on simulated annealing, adaptive crossover between drones and queen bee, adaptive mutation of worker bee[11-14]. The theory of BSGA is as below:

2.1 Drones Operations

According to adaptive crossover rate, stronger drone has greater chance to crossover with queen, but less crossover frequency, on the contrary of weaker drone, so the good model can easily been saved. According to schema theorem, this method guides the direction of the algorithm, and high convergence speed is maintained initially. However, as the algorithm progressing, cross-deceptive problem are easily caused by two high-fitness individuals of high similarity. So the crossover number operator is added, through queen and low fitness drone crossover several times. It can reduce the randomness of search and promote uniform distribution of the individuals, so the low fitness but potential ones get more chances to survival.

2.2 Worker Population Operation

Queen and the workers are restrained in accordance with the similarity degree of similarity. For the binary-coded genetic algorithm bees, the Hamming distance is used to calculate the similarity between two individuals. Specific method is, if the Hamming distance D between worker individual and queen less than or equal to T, the individual mutates. The other individuals who have not been suppressed determine their mutation rate and the location of dynamic mutation in accordance with the relationship of fitness. Principle is that the greater the fitness of individuals, the lower the mutation rate, variation in the chromosomal location of the back of genes, on the other hand, the smaller the fitness, the higher the mutation rate, variation in the chromosomal location of the initial gene. The worker population mainly maintains the population diversity and avoids premature convergence.

2.3 Description of BSGA

Step 1: in the function definition domain, two populations are generated randomly. Drones have N individuals, and the worker population has M individuals.

Step 2: the best one as queen is selected from worker population.

Step 3: Drones select ones through roulette wheel selection, crossover in adaptive rate and crossover frequency, then mutate.

Step 4: In the method of League selection, the female offspring and the original worker population reorganize, and then the queen is reselected again.

Step 5: the queen optimizes in neighborhood by simulated annealing.

Step 6: the queen restrains the new worker population.

Step 7: the other individuals determine the mutation rate and location according to fitness. The queen is selected again.

Step 8: If the algorithm termination conditions are not met, go to Step2, otherwise, queen is output as a global optimal solution, and algorithm terminates.

3 Mechanism Analyzing

Relative to the biological basis of genetic algorithm, its mathematical theoretical[15] foundation is far behind the actual development of the algorithm. Schema theorem and the implicit parallelism[16] is seen as the cornerstone of the theory of genetic algorithms, then there are the building block hypothesis. Schema theorem and building block hypothesis describes the genetic algorithm optimization possibilities and capabilities, but the phenomenon of premature convergence of genetic algorithms, models and genetic algorithm for the measurement of fraud problems, and so are unable to explain. Based on the theory of genetic algorithm, BSGA search procession will be proved limited traverse homogeneous Markov chain, with a global convergence, meanwhile, the selection of the best crossover and adaptive mutation operator are discussed.

3.1 Convergence Analyzing

Theorem 1: BSGA is convergent.

Prove: the queen as the best individual in each generation is preserved. The various states of probability matrix P are in descending order, the first state is the global optimal solution; the second state is the global sub-optimal solution, …. , the N state is the global worst. So P can become:

$$P = \begin{bmatrix} C & 0 \\ R & T \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ p_{21} & p_{22} & \cdots & 0 \\ & \cdots & \cdots & \\ p_{N1} & p_{N2} & \cdots & p_{NN} \end{bmatrix} R, T \neq 0 \tag{1}$$

From the analysis of P, we can see that BSGA is reducible stochastic matrix, $C = [1]$ is a first-order strictly positive stochastic matrix. Then

$$\lim_{k\to\infty} P^k = \lim_{k\to\infty}\begin{bmatrix} C & 0 \\ \sum_{i=0}^{k-1} T^i R C^{k-1} & T^k \end{bmatrix} = \begin{bmatrix} C^\infty & 0 \\ R^\infty & 0 \end{bmatrix} \tag{2}$$

Because of $C^\infty = 1$, BSGA is not only finding the best solution, but also preserving the best one. So BSGA is global convergence.

3.2 Crossover Operator Analyzing

Although BSGA can converge to global optimal solution, but the results are achieved in an infinite algebraic theory of evolution, the actual operation is not feasible. Therefore, algorithms must be completed within a limited generation.

Theorem 2: (1)if $\wp$ is a pattern, then $|\wp| = 2^{L-O(\wp)}$.

(2) if $\wp_1 = \lfloor (i_k, a_{i_k}); K_1 \rfloor$ and $\wp_2 = \lfloor (i_k, a_{i_k}); K_2 \rfloor$ are two patterns, and satisfy the

condition $a_{i_k} = b_{j_k} \; (i_k = j_k)$ then:

$\wp = \wp_1 \lfloor (i_k, a_{i_k}); K_1 \rfloor \cap \wp_2 \lfloor (i_k, a_{i_k}); K_2 \rfloor$ is also a pattern.

In the crossover process, we can see that the randomly selected individuals of matching is more difficult to control, hybrid offspring is produced by different models, not only easily lead to genetic algorithms deceptive problem, but also influence the convergence speed. So the crossover operation is adapted between queen and other individuals in the BSGA. The better model is achieved through the best models and the other model's matching, according to building blocks hypothesis.

3.3 Mutation Operator Analyzing

GA mutation probability is fixed, single-point mutation location is random, although the mutation operator of the search scope is global, but mutation probability is low, and random changes in a very strong need to maintain the diversity of population. Once a local optimal solution is caught, relying on simple mutation operator is difficult to jump out. But BSGA worker population can increase population diversity, and enhance its global search capability.

The worker individual adaptive adjusts the mutation rate, increasing the diversity of population. High gene from the front of the individual has the larger weight to determine the position in the solution space. Low gene from the rear of the individual has the small weight to determine the local position in the nearby of high gene individual. Adaptive adjustment of the gap varies according to the intention to enhance the expected value occurred at a certain unknown probability.

4 Experiments

In order to verify BSGA convergence speed and the ability of convergence to the global optimal solution, a classical test function and knapsack problem is selected.

Example 1 (Schaffer F6 function):

$$\begin{cases} \max f(x_1,x_2) = 0.5 - \dfrac{\sin^2\sqrt{x_1^2+x_2^2} - 0.5}{\left[1+0.001(x_1^2+x_2^2)\right]^2} \\ -5 \le x_i \le 5, (i=1,2) \end{cases} \tag{3}$$

The function maxima of (0,0), maximum value is 1. However, the global maximum is around of local minimum points like a circle, and leading to a general algorithm is very difficult to converge to the optimal solution.

In the same parameter settings, 100-300 times iteration, using method from paper[17], solution results were shown as follows in Table 1 comparing to BSGA.

Table 1. Compared result of calculation

	New Hybrid GA	BSGA
100	0.99944277008649	0.99999038159228
200	0.99998879898559	0.99999999544797
300	0.99999957113885	0.99999999544797

Example 2: Knapsack problem

Example 2 selects knaps number is 50 from paper[18].

The goods value ={220, 208, 198, 192, 180, 180, 165, 162, 160, 158, 155, 130, 125, 122, 120, 118, 115, 110, 105, 101, 100, 100, 98, 96, 95, 90, 88, 82, 80, 77, 75, 73, 72, 70, 69, 66, 65, 63, 60, 58, 56, 50, 30, 20, 15, 10, 8, 5, 3, 1}, the volume of goods ={80, 82, 85, 70, 72, 70, 66, 50, 55, 25, 50, 55, 40, 48, 50, 32, 22, 60, 30, 32, 40, 38, 35, 32, 25, 28,30, 22, 25, 30, 45, 30, 60, 50, 20, 65, 20, 25, 30, 10, 20, 25, 15, 10, 10, 10, 4, 4, 2, 1},maximum volume backpack is 1000, size is 50.

The optimal value is 3095 from paper[19] using the greedy algorithm, 3077 from paper[20] using simple genetic algorithm, and 3103 from paper[18] using hybrid genetic algorithm. In the same parameters of paper[18], BSGA runs independently 50 times, the objective function optimal value was 3103, with an average of 170 generations. From the above comparison we can see that the optimal value of this algorithm is superior to simple genetic algorithm and greedy algorithm, and hybrid genetic algorithm for the same, and this algorithm has good results in speed and stability.

5 Conclusion

In this paper, through the introduction of the concept of gender, BSGA is conscious control of the crossover between the different chromosomes to speed up the convergence rate, so that populations remain the most likely to succeed state. Through similar inhibition and self-adaptive mutation rate, adaptive mutation, the population has a good global search capability. Then the algorithm was proved from a theoretical point of view the effectiveness and convergence, the final test results from a simulation point of view, the algorithm is compared with other classical genetic algorithm and improved algorithm, population size is small, fast convergence, failure rate is low, the average convergence algebra less.

Of course, the drones in this paper and the relationship between the proportion of worker bees and produce random number and sex of the individual issues are beyond the need for further research.

References

1. Qi, R.B., Qian, F., Li, S.J., Wang, Z.L.: Chaos-Genetic Algorithm for Multiobjective Optimization. In: 6th World Congress on Intelligent Control and Automation, pp. 1563–1566. IEEE Press, Dalian (2006)
2. Shen, H.L., Zhang, G.L., Li, Z.T.: Adaptive genetic algorithm based on distance measurement. J. Journal of Computer Applications 27, 1967–1972 (2007)
3. Xing, G.H., Zhu, Q.B.: Genetic Algorithm Based on Adaptive Big Mutation Rate and Simulated Annealing and Its Application. J. Computer Engineering 31, 170–172 (2005)
4. Xu, H.Y., Vukovich, G.: A Fuzzy Genetic Algorithm with effective search and optimization. In: Proceedings of 1993 International Joint Conference on Neural Networks, pp. 2967–2970. Canadian Space Agency, Canada (1993)
5. Xiong, W.Q., Liu, M.D., Zhang, S.Y.: A Genetic Algorithm with Sex Character. J. Computer Engineering. 31, 165–166 (2005)
6. Joanna, L.: A multi-sexual genetic algorithm for multiobjective optimization, pp. 59–64. IEEE Press, Indianapolis (1997)
7. Li, Y.Z., Liu, H.X., Zhang, S.: Improving Monkey-King Genetic Algorithm. J. Journal of Nanjing Normal University. 4, 53–56 (2004)
8. Goldberg, D., RichardSon, J.: Genetic Algorithm with sharing for multi-modal function optimization. In: 2nd International conference on Genetic Algorithms, pp. 41–49. Massachusetts Institute of Technology, Cambridge (1987)
9. Wu, D., Cui, R.Y.: Bee-Swarm Genetic Algorithm. In: China Artificial Intelligence Society of the 11th National Annual Conference Proceedings, Wuhan, pp. 733–736 (2005)
10. Wu, D., Cui, R.Y., Cheng, N.: Improvidng bee-swarm genetic algorithm. J. Journal of Harbin institute of technology 38, 1150–1154 (2006)
11. Nicol, N.S., Richard, K.D.: Dynamic Parameter Encoding for Genetic Algorithm. J. Machine Learning, 1–8 (July 20, 1992)
12. Li, L.: Adaptive Genetic Algorithms Based on Nash Game. J. Computer Engineering and Applications. 40(33), 86–88 (2004)
13. Li, C.W., Ma, H., Han, Z.G.: Study on Randomness in Genetic Algorithm Evolution. J. Application Research of Computers 22, 61–63 (2005)
14. Zhang, J.B., Chen, B.X., Sui, G.R.: New mechanism of GA based on intelligent crossover. J. Computer Engineering and application 45, 35–37 (2009)
15. Fogel, D.B.: An introduction to simulated evolutionary optimization. IEEE Transaction on Neural Networks 5, 3–14 (1994)

16. Michlewicz, Z.: Genetic Algorithms + Data Structure = Evolution Programs. Springer, Heidelberg (1992)
17. Zhao, P.X., Cui, Y.Q., Liu, J.Z.: A New Hybrid Genetic Algorithm for Optimization Problems. J. Computer Engineering and Applications 40, 94–96 (2004)
18. Ma, H.M., Ye, C.M., Zhang, S.: Binary improved particle swarm optimization algorithm for knapsack problem. J. Journal of University of Shanghai For Science and Technology. 28, 31–34 (2006)
19. Jiang, L., Wu, K.: Research for 0—1 Knapsack problem in reedy algorithm. J. Computer and Data Engineer. 38, 32–33 (2007)
20. Ma, H.M., Ye, C.M.: Parallel Particle Swarm Optimization Algorithm Based on Cultural Evolution. J. Computer Engineering 34, 193–195 (2008)

On the Farther Analysis of Performance of the Artificial Searching Swarm Algorithm

Tanggong Chen, Lijie Zhang, and Lingling Pang

Province-Ministry Joint Key Laboratory of Electromagnetic Field and Electrical Apparatus Reliability, Hebei University of Technology, Tianjin, China
tgchen@hebut.edu.cn

Abstract. Artificial Searching Swarm Algorithm (ASSA) is an intelligent optimization algorithm, and its performance has been analyzed and compared with some famous algorithms. For farther understanding the running principle of ASSA, this work discusses the functions of three behavior rules which decide the moves of searching swarm. Some typical functions are selected to do the simulation tests. The function simulation tests showed that the three behavior rules are indispensability and endow the ASSA with powerful global optimization ability together.

Keywords: artificial searching swarm algorithm, bionic intelligent optimization algorithm, optimization, evolutionary computation, swarm intelligence.

1 Introduction

Most animals and insects show the amazing abilities of completing complex behaviors. Since the 1940s, the optimization design problems in the engineering fields have been solved by using the inspiration of the biological systems, and some of algorithms were found and called Bionic Intelligent Optimization Algorithm (BIOA). At present, the popular BIOAs are genetic algorithm [1], ant colony algorithm [2], particle swarm optimization [3], artificial fish-swarm algorithm [4], and shuffled frog leaping algorithm [5], etc. These bionic algorithms have become a hot research focus in the fields of intelligent optimization.

Artificial Searching Swarm Algorithm (ASSA) is an abstract BIOA. It bases on the simulation of the running principle of BIOA and the process of finding the optimal goal by a troop of soldiers [6]. For farther understanding the principle of ASSA, this work discusses the functions of three behavior rules which decide the moves of searching swarm. Some typical functions are selected to do the simulation tests. The function simulation showed that the three behavior rules are indispensability, and together endow the ASSA with powerful global optimization ability.

2 Artificial Searching Swarm Algorithm

According to the principle of BIOA, ASSA uses the searching swarm (simulate a troop of soldiers) composed of searching individuals (simulates a soldier) as the main executive to implement the searching task, and utilizes the vested rules to change the searching individual's position. With the algorithm iteration the searching swarm moves in the searching region continuously, and finds the optimal solution finally.

So whether the searching swarm moves forward to the optimal solution is the key to the success of algorithm, the rules become the important problem which decides the searching swarm's movement mechanism directly, and affects the efficiency of the algorithm.

Three rules are defined to restrict the searching individual's behavior:

1) Communication: the basic communication relation is kept between searching individuals. If an individual receives a call sent by his companions, it moves forward to the called companion's position by a step with a certain probability;
2) Reconnaissance: If the individual does not receive the call of his companions, it implements reconnaissance according to his and swarm's historical experience. If finds a better goal, moves forward to the position by a step;
3) Move: If the searching individual does not receive a signal of his companions, and does not find a better goal, it moves a step randomly.

If finds a better goal during above moves, sends a call to his companions.

The running flow of ASSA is shown as follows:

1) Set the parameters, initialize the swarm randomly and evaluate the fitness value;
2) Iteration counter add 1, deal with the individuals in turn as follows:
   a) If receive his companion's call, then move forward to the called companion by a step with a certain probability;
   b) Otherwise, according to its own and swarm's historical experience implement the reconnaissance. If find a better goal, move forward to the better goal by a step;
   c) Or move by a step randomly.

If find the better goal during above moves, send a call to his companions.

3) Calculate the fitness value. Compare with the best fitness of swarm and each individual respectively, if better, log on the bulletin board;
4) Determine whether or not to meet the conditions of termination, if so, end the iteration; otherwise, return to 2).

According to the searching individual's behaviors above, the searching swarm moves constantly in the searching area, and approaches to the optimal solution through the continuous iteration of the algorithm, and obtains the optimal solution finally.

3 The Function Analysis of Behavior Rules of ASSA

3.1 The Simulation Functions

To analyze the functions of the three behavior rules of ASSA, the following typical functions are chosen for simulation experiment:

$$F_1(x,y) = \frac{\sin(x)}{x} \times \frac{\sin(y)}{y} \quad x, y \in [-10,10] \tag{1}$$

$$F_2(x_1, x_2, \cdots, x_n) = \sum_{i=1}^{n-1} 100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2 \tag{2}$$
$$x_i \in [-2.048, 2.048],\ \ i = 1,2,\cdots,n$$

$$F_3(x_1, x_2, x_3) = 30 + \sum_{i=1}^{3} (\frac{1}{10}(x_i - 20)^2 - 9\cos\frac{2\pi}{5} x_i) \tag{3}$$
$$x_i \in [1,39] \quad i = 1,2,3$$

$$F_4(x_1, x_2) = 0.5 + \frac{\sin^2\sqrt{x_1^2 + x_2^2} - 0.5}{[1.0 + 0.001(x_1^2 + x_2^2)]^2} \tag{4}$$
$$. \qquad x_i \in [-100,100] \quad i = 1,2$$

$$\begin{aligned} F_5(x_1, x_2) = {} & [1 + (x_1 + x_2 +)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 \\ & + 6x_1x_2 + 3x_2^2)[30 + (2x_1 - 3x_2)^2 (18 - \\ & 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)] \end{aligned} \tag{5}$$
$$x_i \in [-2,2] \quad i = 1,2$$

Function F_1 has a global maximum 1 at point [0, 0], and a lot of local maximums distributed around it.

F_2 is a general Rosenbrock function. It has the global minimum value 0 at the points [1, 1,…,1].

F_3 has a global minimum value 3 at point [20,20,20], and a lot of local minimum value around it, it is a multi-variable problem.

F_4 is Schaffer function and has a global minimum 0 at point [0, 0].

F_5 is Goldstein-Price function and has global minimum 0 at point [0, 0].

3.2 The Function Analysis of Communication Rule

According to the communication rule, the basic communication relation is kept between the searching individuals. If an individual finds a better goal, it sends a call to his companions. At next iteration the individual checks the communication record firstly, if receives a signal sent by his companions, it moves forward to the called companion's position by a step with a certain probability.

It is important to limit an individual to respond the call of his companions with a certain probability. Therefore the value of the probability is a key factor that influences the function of communication rule and the performance of ASSA.

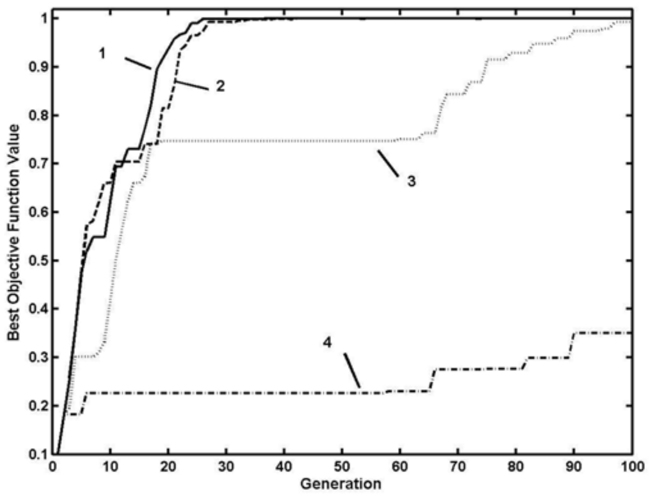


Fig. 1. Comparative results with different value of Pc (1 : Pc=0.002, 2 : Pc=0.02, 3 : Pc=0.2, 4 : Pc=1.0)

For verifying the analysis above, function F_1 is selected to do the simulation tests. It is a maximum optimization problem. The swarm size is 10, searching step is 0.3, iterative number is 100, and the value of probability is signed by Pc ($0 \le Pc \le 1$, generate a stochastic number r by randomizer, if r<Pc, then perform communication rule). The results are shown as Figure 1.

In order to achieve the same purpose, function F_2 is selected to do simulation tests. It is a minimum optimization problem. The swarm size is 10, n is 3, searching step is 0.2, iterative number is 100, and the value of probability is signed by Pc. The results are shown as Table 1 (the figure does not show distinctly).

Table 1. The Testing Results of Function F_2

Pc	The Optimal Value					
	10	30	50	70	90	100
0.002	1.429595	0.190104	0.001533	0.000961	0.000264	0.000264
0.02	1.429595	0.190104	0.001533	0.000961	0.000049	0.000049
0.2	0.719591	0.040261	0.005819	0.001653	0.000560	0.000560
1.0	3.276597	0.167835	0.007813	0.007813	0.007621	0.001392

Here the numbers 10, 30, ---, 100 denote the iterative number.

From results ASSA gets optimal value 0.0015 at 50-th generation while Pc is 0.002, gets 0.0015 at same generation while Pc is 0.02, gets 0.0058 at same generation while Pc is 0.2, and gets 0.0078 at same generation while Pc is 1.0.

Communication rule can influence the performance of ASSA obviously; the value of probability which limits the use of communication rule is a key factor to perform this rule; selecting the value of probability properly can enhance the performance of ASSA.

3.3 The Function Analysis of Reconnaissance Rule

According to the reconnaissance rule, if an individual does not receive the calls of his companions, it implements reconnaissance according to his and swarm's historical experience. If finds a better goal, moves forward to the position by a step.

Suppose the r_1, r_2 are two random real numbers, $0 \le r_1, r_2 \le 1$, the current individual's position is X_i , the historical best solution of the individual is X_s, the historical best solution of the swarm is X_g, and the absolute value function is $|\cdot|$, then the individual decides the reconnaissance goal X by the formula (6) as follows.

$$X = X_i + r_1 * |X_s - X_i| + r_2 * |X_g - X_i| \tag{6}$$

Reconnaissance rule plays a main role to find the better goals and prevent the local convergence.

How to define the "better goal" is a key problem to perform the reconnaissance rule. There are three definitions to be selected in practice. The first is better than the current value of objective function of the individual; the second is better than the historic value of the objective function of individual; the third is better than the historic value of objective function of the swarm. The different definitions are related to different performance of ASSA.

For verifying the influence of the different definition of the better goal, function F_3 is selected to do simulation tests. The swarm size is 50, Pc is 0.002, searching step is 2.6, and iterative number is 100. The simulation results are shown as Figure 2.

From the results this rule is a main rule of ASSA to find the better goal, and to keep a good performance. The first definition endows the algorithm the best performance than the otherwise definitions.

Without reconnaissance rule the performance of ASSA is very low. The results are shown as table 2.

Table 2. The Testing Results without reconnaissance rule

	The Optimal Value					
	10	30	50	70	90	100
With rule	8.341815	6.179004	3.169186	3.087715	3.009977	3.004805
Without tule	10.452804	10.452804	10.452804	10.452804	10.452804	10.452804

(10, 30, ---, 100 denote the iterative number)

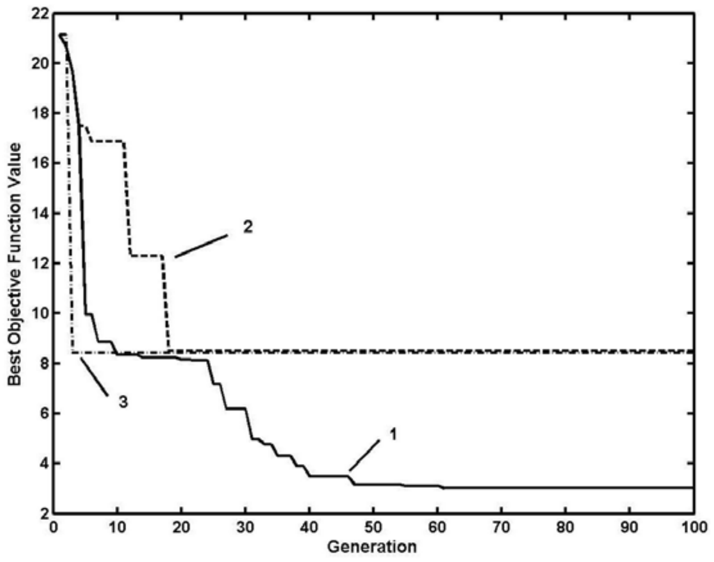


Fig. 2. Comparative results with different definition of the better goal (1 : with the first definition, 2: with the second definition, 3: with the third definition)

3.4 The Function Analysis of Move Rule

According to the move rule, if the searching individual does not receive a call of his companions, and does not find a better goal, it moves a step randomly.

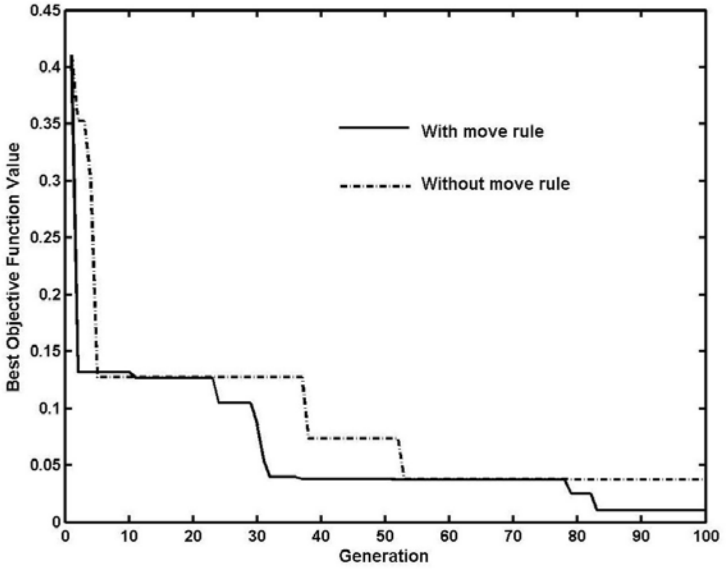


Fig. 3. Performance Comparison with (without) move rule

Obviously the move rule only has the assistant function to find the better goals, bit it is necessary. For analyzing the efficiency of move rule, function F_4 is used to do the simulation tests. The swarm size is 100, searching step is 2.6, Pc is 0.2, iterative number is 100. With and without the move rule, Figure 3 shows the simulation results. Although the influence is not so strong, as an assistance rule, the efficiency is obvious.

Actually move rule can be used more than one times to improve the opportunity to find the better goal and performance of ASSA. If an individual does not find the better goal during the first random move, it will try three times totally to perform move rule.

It is clear that the more times the move rule be executed the more chances the algorithm can finds the better solution. But the trying times is restricted to decrease the running time of the algorithm.

The results show that performing the move rule three times in the iteration can improve the searching ability and accelerate the convergent speed of ASSA.

3.5 Analysis of Running Example of ASSA

For analyzing the functions of the behavior rule in the running examples, function F_5 is selected to do simulation tests. The size of the searching swarm is 10, searching step is 0.3, Pc is 0.2, and the iteration number is 100. The results are shown as table 3.

Table 3. The Testing Results of Function F5

	Experiment Times							
	1	2	3	4	5	6	7	8
Rule 1	16	40	13	29	42	22	25	15
Rule 2	863	798	945	852	917	901	947	896
Rule 3	121	162	42	119	41	77	28	89
Optimal Value	3.007613	3.177619	3.000190	3.000589	3.001296	3.001130	3.000584	3.000039

Here the numbers 1, 2, ---, 8 denote the experiment times, the third row numbers are the executing times of rule 1 during the one experiment, and so on. For example rule 1 is executed 16 times in first experiment while rule 2 is executed 863 times; rule 3 is executed 121 times.

From the results reconnaissance rule plays a leading role, the smaller the Pc value and more times of this rule, the more accurate optimal value. But the supplementary role of communication rule and move rule can not be neglected.

4 Conclusion

Swarm intelligence is a research branch that models the population of interacting agents or swarms that are able to self-organize. The Bionic Intelligent Optimization Algorithms (BIOAs) have become a hot research focus in the fields of intelligent optimization. For almost all the BIOAs the behavior rules are the key factor that influences the algorithm performances, and decides the biology mechanism of algorithms. Artificial Searching Swarm Algorithm (ASSA) is an abstract BIOA, it has no direct simulation foundation, and just simulates a troop of soldiers (searching individuals) to search for the vested goal. So its behavior rules have special characteristics.

Reconnaissance rule plays a leading role to find the better goals and influence the performance of ASSA among three rules of ASSA; communication rule and move rule have an assistant function. But the three behavior rules are indispensability, and endow the ASSA with powerful global optimization ability together. The more works will be done to analyze the running principle of ASSA deeply in the future.

Acknowledgment

The author would like to acknowledge gratefully to National Natural Science Foundation of China (NNSFC). This work was supported by NNSFC under Grant No. 50877022.

References

1. Holland, J.H.: Adaptation in Nature and Artificial System. MIT Press, Cambridge (1992)
2. 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)
3. Kennedy, J., Eberha, R.C.: Particle Swarm Optimization. In: Proceedings of the IEEE International Conference on Neural Networks, pp. 1942–1948 (1995)
4. Li, X.L., Shao, Z.J., Qian, J.X.: An Optimization Method Based on Autonomous Animats: Fish-swarm Algorithm. Systems Engineering-Theory & Practice 22(11), 32–38 (2002)
5. Eusuffm, M., Lansey, K.E.: Optimization of Water Distribution Network Design Using Shuffled Frog Leaping Algorithm. J. Water Resources Planning and Management 129(3), 21–225 (2003)
6. Chen, T.G.: A Simulative Bionic Intelligent Optimization Algorithm: Artificial Searing Swarm Algorithm and its performance Analysis. In: Proceedings of the Second International Joint Conference on Computational Sciences and Optimization, vol. 2, pp. 864–866 (2009)

Orthogonality and Optimality in Non-Pheromone Mediated Foraging

Sanza Kazadi, James Yang, James Park, and Andrew Park

Jisan Research Institute

Abstract. We describe the general foraging task, breaking it into two different subtasks: map-making and collection. Map-making is a task in which a map is constructed which contains the location(s) of an item or of items in the search area. Collection is the task in which an item is picked up and carried back to a central known location. We theoretically examine these tasks, generating minimal conditions for each one to be accomplished. We then build a swarm made up of two castes to accomplish this, theoretically motivating the design of the swarm. Finally, we demonstrate that the swarm is optimal in the class of swarms utilizing line-of-sight communication, and give performance measures for open and closed search spaces.

1 Introduction

Foraging is a task required of virtually every animal on earth. Generally speaking, the task consists of two phases: the items being foraged need to be located, and the items need to be carried back to a central location. These two phases of the overall task are combined into a single goal which must be achieved in order for the group to function.

In the insect world, the goals are achieved in a variety of ways. Bees use a combination of map making[11], communication, and random search to find, mark, and communicate the position of food being foraged. Ants either move their central location to the food or move the food to their home by marking the environment and using it as a map to recruit collection of the food.

Many groups have investigated foraging from numerous different angles. Evolution of foraging in which foraging behaviors evolve generally center around the development of a foraging behavior using evolutionary processes [6]. These studies conclude with the evolutionary algorithm generating successful foraging behavior. However, the agents and their potential instantiations are broadly outlined by the studies' computational set-ups. Others have examined the development of swarms and their functions to their agent's limited capabilities.

These studies mostly followed the ant model in which wandering agents encounter the target, form a "map", which might be via wireless communication, a GPS-like locator [4], deposit pheromones [5], and collect the target with the aid of other agents. This approach works quite well when a map or a method of marking the environment is available. However, remote missions to distant locations may preclude the use of GPS, limit wireless communication, and occur where pheromones are limited. All of these might hinder the ability to generate and update a map a with garnered information. Agents might be limited to sight-only communication, and not have access to global positional information or a map. Moreover, with the current ability of agents to employ dead-reckoning, which is significantly limited when compared to the insect world, such environments may prove challenging.

A prospective solution to such issues is the bucket brigading method. Bucket brigading involves agents from home being sent out to locate pucks. Once an agent has located a cluster, it alerts the swarm and consequently the swarm forms a line from home to the puck, with the agent closest to the cluster takes one and passes it down the line until the puck is deposited at home. However, a key obstacle they encounter is spatial interference. Spatial interference refers to disruptive interference between robots to perform work in a system [6]. Shell and Mataric's robots productivity decreased as the number of robots increased, due to interaction and collisions.

To reduce spatial interference, we turn to a simulation by Lein and Vaughan in [8]. What separates their simulation from Shell and Mataric's is that the search radius of each robot would decrease when it entered another robot's search radius, effectively reducing spatial interference. They recorded an increased number of pucks collected as the number of robots increased at the cost of search efficiency.

We examine the foraging problem from the point of view that three tasks must be achieved. We constrain the method to the use of line of sight communication, precluding global communication or map-making. We develop a model that breaks the task into three categories (search, mapping, and collection), and accomplishes these using two different castes of agents. We demonstrate that the solution is optimal in the spaces that require line of sight communication. We further examine the use of this method in closed and open spaces, generating performance measures for closed spaces as a function of the position of the agents' "home".

2 Engineering Foraging without Stigmergy, Gps, or Dead Reckoning

We are interested in generating model independent design requirements for the foraging task. Moreover, we would like to include one which requires the use of localized technologies:

1. *Agents must use technologies that are entirely independent of an existing external infrastructure.*

This requirement limits our likelihood to generate a foraging strategy that works in a vast subset of interesting environments. Such environments include underground, underwater, physically hostile (radioactive, chemically polluted, or otherwise hostile environments), and remote areas with little built in infrastructure.

These environments are very likely to escape coverage from GPS, and normal communication networks. As a result, the swarm must be expected to bring with it all hardware one might require to accomplish the task.

This and available technologies requirement limits the technologies and the strategies we can use. As an example, dead reckoning in robotics today is currently incapable of providing the kind of positional knowledge that ants exhibit. In general, if $err(t) = |x_{est}(t) - x_{act}(t)|$ then $t \nearrow \Rightarrow err \nearrow$. After some time, the error in the estimate of the position is so great that no reliable measure of the position of the robot can be made.

We must utilize things that the foraging agents can reliably measure and information that is readily and locally available. Given the state of localization technologies today, we can make the assumption that agents can localize objects within their field of "vision" along with other agents. That is, each agent is capable of determining the distances $\overrightarrow{x - x_j}$ where $\overrightarrow{x}$ is the position of the agent and $\overrightarrow{x_j}$ is the position of the j^{th} neighbor. Agents can communicate, and therefore can share information. As a result, we may assume that each agent has the ability to know the location of some subset of the other agents in the same continuous group[1], including the possibility that each agent knows the positions of all other agents. We assume that the agents have a sensor range $d_s > 0$ for targets defined by the maximum distance from an agent to a target that it can sense. We also assume that the agents have a maximum communication and sensor range $d_c > 0$ for other agents defined analgously.

The question is, how does one arrive at a design method for foraging which *is not dependent on the agent model*?

2.1 General Considerations

We can start by defining a position for every agent and every target. Let $\{\overrightarrow{x_i}\}_{i=1}^{N_A}$ be the set of positions of agents. Let $\{\overrightarrow{y_i}\}_{i=1}^{N_T}$ be the set of positions of all the targets. Let $\{s_i\}_{i=1}^{N_T}$ represent the set of all states of targets. Let us assume that each target can take on one of two values which we can represent by the set $T = \{0, 1\}$. The state 0 represents a target whose position is not currently known by any agent. State 1 represents a target whose position is currently known by an agent. Note that it is possible for an agent to find a target and "lose" it again. Let us define the home position as $\overrightarrow{x_h}$. We can also define the amount of energy required by each of the agents per unit time in motion and in computation as $\{e_i\}_{i=1}^{N_A}$. Of course, in homogeneous swarms, each of these e_i's are identical.

Using these definitions, we can define the global properties [1] $\overrightarrow{X} = \sum_{i=1}^{N_T} (\overrightarrow{y_i} - \overrightarrow{x_h})$ and $S = \sum_{i=1}^{N_T} s_i$. These represent the aggregate position and state of the targets. What then must happen is that

$$\frac{dS}{dt} \geq 0 \tag{1}$$

[1] We define a *continuous group* to be a group of individuals in which, given any pair of agents, a set of agents can be found which "connects" the agents in the pair.

and that

$$\frac{d\left|\overrightarrow{X}\right|}{dt} \leq 0. \tag{2}$$

In order for this to happen, the first condition tells us that any agents must have a behavior that leads them to find a continually increasing number of targets, losing fewer than are found in any given time period. This must eventually lead to either stagnation in the number found or the number found converging to the total number, when the total number is finite. The second condition requires that the targets' positions become closer to that of the home. If we further require that

$$\int_{t=0}^{t=t_f} \frac{d\left|X\right|}{dt} dt = -\left|X\left(t=0\right)\right| \tag{3}$$

and

$$\int_{t=0}^{t=t_f} dS = N_T \tag{4}$$

then conditions (2) and (3) give us that the final position of the targets should be at home. Also, the conditions (1) and (4) require that all targets are found.

Since we do not know initially where the targets are, we must define a search area A and a behavior of the agents that satisfies conditions (1), (2), (3) and (4). There is no restriction on the method that achieves this.

2.2 Foraging without Dead Reckoning, GPS, Existing Maps, and Stigmergic Communication

As our restrictions are that there is no GPS nor adaptive mapping, we must use either a form of stigmergic communication or direct communication to map the region. It is possible to dig trenches or otherwise change the characteristics of the environment in which the search is progressing. In sandy and windy areas, digging trenches would be non-permanent. However, in the absence of wind, the agents themselves and the trenches might produce a confusing set of tracks to and from a source that yields little in the way of information communication.

Other methods such as imprinting a small magnetic signature in the soil or leaving short-lived imprint on the environment depends very much on the kind environment. It is unlikely that utilizing a magnetic field would leave an imprint. In contrast, utilizing a high temperature heater on dry land might heat the local area in a way that is relatively short-lived. Such a device would be useless in an underwater or submerged environment, and have limited use in the presence of significant wind.

These considerations lead us to search for a system which is capable of carrying out the search with the following requirements.

1. The agents are capable of obtaining objects whose physical distance is well beyond the ability of an individual agent to sense or to reliably retrieve;

2. The system is optimal in the sense that the physical distance traveled is minimized;
3. Any mapping is achieved using either the physical positions of the agents or communicated positions that have been calculated and communicated to allow for retrieval.

These requirements are subject to the following restrictions

1. The agents have a decreasing probability of knowing the direction and range to their "home" as they move further away from direct contact with it either through direct sensing or through a network of other agents;
2. The agents have a limited sensor range, where the sensors are searching for the target; and
3. The agents have a limited communication and sensor range, where the sensors can determine range and bearing to other agents.

The modality we are looking for, which will be robust in many different situations must come from the very limited set of line-of-sight or line-of-sound sensory modalities. These include sound transceivers and/or sight or with light transceivers. These might also include touch sensors [2] which would require physical contact between the agents.

These restrictions leave us with the requirement that $\forall i \; \exists j$

$$|\overrightarrow{x_i} - \overrightarrow{x_j}| \leq d_c \tag{5}$$

where d_c represents the sensor range of the agents. This is a requirement of the behavior as well.

Given these requirements, it is easy to see that, given a single home location $\overrightarrow{x_h}$, the swarm will have the longest reach if the agents form a linear structure. I.e. it must be that $\forall i < N_s - 1$, renumbering if necessary,

$$|\overrightarrow{x_i} - \overrightarrow{x_{i-1}}| = d_c \tag{6}$$

and

$$\hat{x_i} \cdot \hat{x_j} = 1. \tag{7}$$

These requirements indicate that the optimal solution occurs when the agents are arranged linearly and each agent is located at the edge of their neighbors' sensor ranges. Therefore, the solution that we are looking for is one which covers the largest search space by creating a line of agents whose positions are at their neighbor's sensory edge. Given this, together with the requirements (1) and (4), we must have a behavior that touches every point in the covered area A. Behaviors which achieve this have been discussed elsewhere [2,3,9], though these discussions were not motivated by optimizing the agent's behaviors. Note that the useful sensory distance d_s may be very much less than that of the sensors designed to facilitate inter-agent communication and coordination. As a result, it is quite possible for the agents to be arranged so as to have no overlap between these sensory regions.

2.3 Castes

Now that we have determined that the agents must generate physical organizations linking the agents together, we must examine what behaviors the agents can employ to find targets within range of the home location $\overrightarrow{x_h}$. We start by referring back to our global goals. The first is the change of state goal ([1]). This goal can be achieved by bringing one or more of the agents within sensor distance of the target, while connected to the linear structure. This means that the linear structure must be either constructed near a target or brought to the target once constructed. The latter is more energetically favorable.

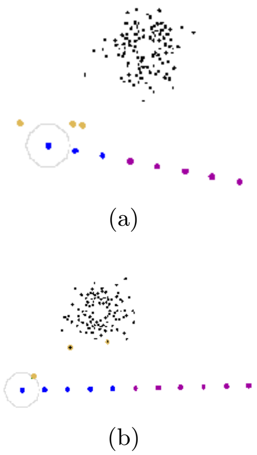
Once the map has been made, it can be left in place so as to facilitate recovery of the targets or can communicate the precise distance and direction. As the agents cannot maintain an accurate direction for long distances, the line must be held in place while the target is retrieved. The retrieval task can occur using agents already in the line, or using a reserve waiting at the home location. It can be demonstrated that using agents in the line is energetically unfavorable when compared to using agents in reserve. As a result, two castes of agents - one which carries out the search and one which retrieves identified targets - emerge as the most energetically favorable solution. Improvements in technology may serve to alleviate the requirement[2].

Our swarm requirements then are that the agents are capable of communication, movement, and sensing. We require that the agents be capable of generating and moving a linear structure in a coherent way around a home. We also require that the agents are capable of moving from the home location to the targets and carrying them back to home. All which are dependent on the platform, its capabilities, and its usage. From the point of view of the engineer, this represents a model-independent set of requirements, and must now be applied to produce a specific behavioral, sensory, and computational model.

3 Implementing the Swarm Requirements

We developed a linear two-caste swarm based on these requirements. One caste generates a linear structure, while the other waits for targets to be identified. Those agents in the linear structure employ behaviors which result in the coherent counterclockwise sweep of the structure, which eventually covers the entire searchable area. If the targets are identified, the second caste is deployed from the "nest" and individuals pick up and transport targets to the "nest". Typical situations are depicted in Figure 1.

[2] In the presence of a truly workable dead reckoning capability, it is possible to utilize coordinated space-filling curves to identify targets, with all agents working in tandem and without overlaps. Agents would be capable, as in the natural world, of retrieving the target, returning to the home, and recruiting assistance from the home.


(a)

(b)

Fig. 1. Two simulations containing seven (a) and ten (b) agents in caste one and three in caste two

By calculating the theoretically minimal distance travelled to locate and retreive the targets, we can estimate the efficiency of the model as a function of the actual distance travelled. We define the efficiency as

$$e = \frac{d_t}{d_a} \tag{8}$$

where d_a is the actual total travel distance and d_t is the theoretical total travel distance. Optimal swarms achieve an efficiency of 1.0, while very inefficient swarms achieve an efficiency of nearly 0.

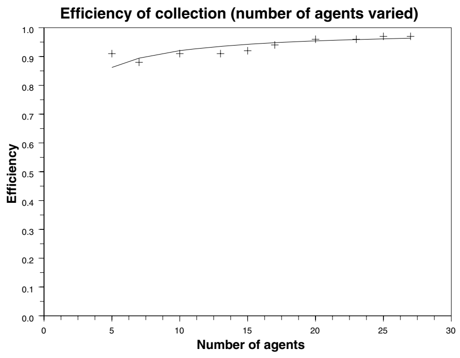


Fig. 2. The efficiencies of the search caste over various swarm sizes are quite high and relatively constant. As expected, the simple reactive agents have very good efficiencies. Surprisingly, the efficiencies tend to increase somewhat as the search caste increases in size.

The two caste swarm has extremely high performance efficiencies, as illustrated in Figures 2 and 3. Figure 2 graphs the efficiencies of the search caste for several different swarm sizes.

The distance calculations for the carrier robots are based on a single carrier robot vacillating between each puck and a home. The theoretical model for the total collecting distance traveled is $\sum_{n=1}^{x} 2p_n$ units where p_n is the distance between the n^{th}puck and home, and x is the total number of pucks. We measure efficiency as above, and graph, in Figure 3, the collection efficiencies of the agents for various swarms.

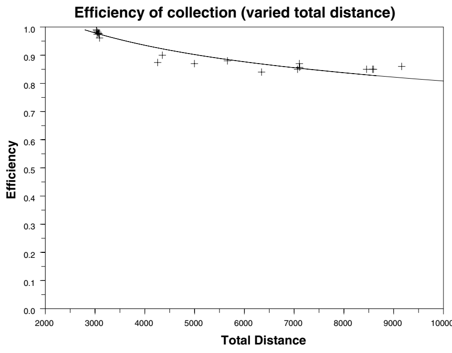


Fig. 3. The efficiencies of the collection cast are somewhat lower than their respective efficiencies for the search caste, though they are still quite high. The collectors tend to have decreasing efficiencies with increasing total distance traveled, though they tend to level off at a relatively high efficiency.

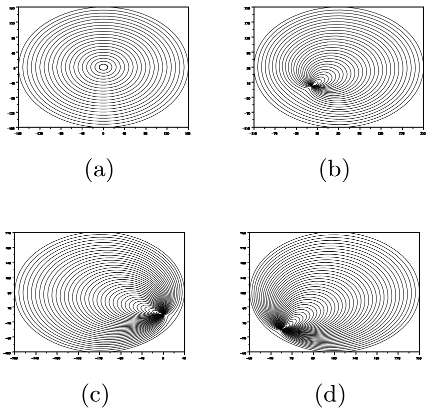


Fig. 4. This gives the search spaces and paths taken by agents during the searches of various closed paths. The efficiencies of the searches are 0.95 (a), 0.79 (b), 0.60 (c), and 0.58 (d).

Overall, these represent very good performances for swarms of autonomous individuals with relatively noisy sensors and distance limited peer-to-peer communication.

Similar techniques may be applied to closed spaces even when the "nest" is not near the center. If the initial caste is adaptive in the sense that the linear structure reaches from "nest" to boundary of the closed space and can lengthen or shorten when needed, the situation becomes as illustrated in Figure 4. Despite these unknown and restricted spaces, the efficiencies can still be quite high.

One interesting feature is that the number of agents in use at each moment is adaptive, and this leads to tight packing of pathways near the "nest". Far from the "nest", however, the path distribution is indistinguishable from the optimum.

4 Discussion and Conclusions

The majority of work in swarm engineering has bypassed questions regarding optimality of the algorithms or the use of multiple types of agents in the swarm. Indeed, some [10] have specifically stated that one of the design requirements for swarms, particularly for swarm robotics, is that the agents be identical or have a small number of groups of large numbers of individuals which are individually homogeneous. However, in stark contrast to this, the natural world seems to have evolved a multitude of caste systems which allow swarms to accomplish tasks far more efficiently than a single agent system might.

Our paper discusses just such a system in which a caste system is more efficient than the same system with the absence of separate castes. The motivation in this paper was to provide a framework for generating a swarm in which the agents are unable to utilize stigmergic communication or any type of GPS or dead reckoning. In our current technological state, and in a variety of realistic environments, these potential tools would be impractical or impossible to use. Therefore, a robust and general purpose swarm must not depend on them. The system we generated is optimally energetically efficient (or nearly so) with observed deviations from theoretical optima being less than 10%. Imperfections in the actual behaviors of the agents, caused by their relatively uncoordinated actions, lower, but do not significantly degrade the swarm behavior.

The two-caste swarm is useful in a variety of domains. Certainly, search and retrieval tasks are obvious areas, including underwater mining and coordinated search in hazardous terrain. However, other tasks such as coordinated surveillance and deployment of resources to needed areas might also fit the bill. Finally, off-world development could benefit from surface-based mining, despite the unavailability of basic mapping technologies. It is clear that improvements in autonomous mapping capabilities may obviate the need for the such a swarm. Until these improvements are achieved, this swarm represents a generally applicable robust and optimally efficient swarm.

Very few natural swarms with sophisticated activities are made up of identical agents. Indeed, our own natural human swarm originally bifurcated into two castes, and from there into literally tens of thousands. This paper takes a step in this direction for the field of swarm engineering. We have demonstrated that it is possible to develop an optimally efficient swarm of more than one caste. Moreover, the fact that this approach is more efficient than a single caste swarm indicates the existence of problems that are properly handled by multi-caste swarms. Development of theoretical and engineering tools to handle these situations and, indeed, to be able to identify these situations, is an open and interesting problem in swarm engineering.

Future work in this area must begin to address how problems best solved by multi-caste swarms can be identified. Once such a problem has been identified, it is important to begin examining how one might go about designing a swarm of this type. Finally, developing a general methodology for validating swarm design in the multi-caste system is an important step that needs to be addressed.

References

1. Kazadi, S.: On the Development of a Swarm Engineering Methodology. In: Proceedings of IEEE Conference on Systems, Man, and Cybernetics, Waikoloa, Hawaii, USA, October 2005, pp. 1423–1428 (2005)
2. Kazadi, S., Kondo, E., Cheng, A.: A Robust Centralized Linear Spatial Search Flock. In: Proceedings of IASTED International Conference Robotics and Applications, Honolulu, Hawaii, USA, pp. 52–59 (August 2004)
3. Chang, K., Hwang, J., Lee, E., Kazadi, S.: The Application of Swarm Engineering Technique to Robust Multi-chain Robot System. In: Proceedings of IEEE Conference on Systems, Man, and Cybernetics, Waikoloa, Hawaii, USA, pp. 1429–1434 (October 2005)
4. Sauter, J.A., Matthews, R., Parunak, H.V.D., Brueckner, S.A.: Performance of Digital Pheromones for Swarming Vehicle Control. In: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems, The Netherlands (July 2005)
5. Panait, L., Luke, S.: A Pheromone-Based Utility Model for Collaborative Foraging. In: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, New York, USA, pp. 36–43 (July 2004)
6. Panait, L., Luke, S.: Learning Ant Foraging Behaviors. In: Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems, ALIFE-IX (2004)
7. Shell, D.A., Mataric, M.J.: On Foraging Strategies For Large-scale Multi-robot Systems. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China, pp. 2717–2723 (2006)
8. Lein, A., Vaughan, R.T.: Adaptive Multi-robot Bucket Brigade Foraging. In: Proceedings of the Eleventh International Conference on Artificial Life (ALife XI) (August 2008)
9. Werger, B.B.: A Situated Approach to Scalable Control for Strongly Cooperative Robot Teams. Ph. D. Thesis, University of Southern California (September 2000)
10. Dorigo, M., Sahin, E.: Guest editorial: Swarm robotics. Autonomous Robots 17(2-3), 111–113 (2004)
11. Menzel, R., et al.: Honey bees navigate according to a map-like spatial memory. PNAS 102(8), 3040–3045 (2005)

An Adaptive Staged PSO Based on Particles' Search Capabilities

Kun Liu, Ying Tan[⋆], and Xingui He

Key laboratory of Machine Perception, Ministry of Education,
Peking University, Beijing 100871, China
ytan@pku.edu.cn
http://www.cil.pku.edu.cn

Abstract. This study proposes an adaptive staged particle swarm optimization (ASPSO) algorithm based on analyses of particles' search capabilities. First, the search processes of the standard PSO (SPSO) and the linear decreasing inertia weight PSO (LDWPSO) are analyzed based on our previous definition of exploitation. Second, three stages of the search process in PSO are defined. Each stage has its own search preference, which is represented by the exploitation capability of swarm. Third, the mapping between inertia weight, learning factor (w-c) and the exploitation capability is given. At last, the ASPSO is proposed. By setting different values of w-c in three stages, one can make swarm search the space with particular strategy in each stage, and the particles can be directed to find the solution more effectively. The experimental results show that the proposed ASPSO has better performance than SPSO and LDWPSO on most of test functions.

Keywords: Particle Swarm Optimization, Exploitation and Exploration, Staged Search Strategy, Performances.

1 Introduction

Particle swarm optimization (PSO) algorithm [1] comprises a very simple concept and can be implemented easily on various optimization problems. A review of the applications of PSO is presented in [2] by identifying and analyzing around 700 PSO application papers stored in the IEEE Xplore database. Researchers keep working on improving the performance of PSO by introducing various mechanisms into the model.

The inertia weight w and the learning constant c are the most important parameters in PSO. w is usually considered as the knob of the PSO's exploration and exploitation capabilities. Many variants of PSO focus on better setting strategy of w. One of the classical variants is the linearly decreasing inertia weight PSO (LDWPSO) [3]. More variants on the settings of w for PSO have been made, such as a nonlinear decreasing inertia weight PSO [4] and chaotic inertia weight PSO [5], to name a few. In most of the existing studies, the idea that bigger w leads to stronger exploration capability while smaller w will facility exploitation capability of the swarm is unanimous. However, with

[⋆] Corresponding author.

different settings of the learning factor c, even the same value of w can lead to different search strategies of the swarm.

In this study, we learn that the search process of PSO can be divided into three stages, and each stage should be treated separately. The analysis to the search process is based on the statistical results of the records about our previous definitions of exploitation capability in SPSO and LDWPSO. Furthermore, The mapping between w-c and the exploitation can be drawn, which can help users set different w-c in each stage to improve the performances of PSO. This new variant of PSO is called adaptive staged particle swarm optimization (ASPSO).

The remainder of this paper is organized as follows. Section 2 describes a standard PSO algorithm. Section 3 elaborates the mathematical analyses about the PSO's search process based on the particles' search capabilities. In Section 4, the ASPSO is proposed and compared among SPSO and LDWPSO. Finally, the concluding remarks are drawn in Section 5.

2 Particle Swarm Optimization

Bratton and Kennedy defined a standard PSO (SPSO) as a baseline for performance testing of improvements to the technique [6]. SPSO includes a local ring topology, 50 particles, non-uniform swarm initialization, and boundary conditions wherein a particle is not evaluated when it exits the feasible search space. The constricted update rules suggested in [6] can be altered by the rules with inertia weight as shown in (1) and (2) [7].

$$\begin{aligned}\hat{V}_{id}(t+1) &= wV_{id}(t) + c_1 r_1 (P_{iBd}(t) - X_{id}(t)) \\ &\quad + c_2 r_2 (P_{nBd}(t) - X_{id}(t)), \qquad (1)\\ X_{id}(t+1) &= X_{id}(t) + \hat{V}_{id}(t+1). \qquad (2)\end{aligned}$$

where $i = 1, 2, \cdots, n$, n is the number of particles in the swarm, $d = 1, 2, \cdots, D$, and D is the dimension of solution space. The learning factors c_1 and c_2 are nonnegative constants, r_1 and r_2 are random numbers uniformly drawn from the interval $[0, 1]$, which are all scalar quantities for each particle in each dimension. P_{iBd} and P_{nBd} are the best positions found so far by particle i and its neighbors in the d-th dimension .

The suggested values of the inertia weight and the constants are: $w = 0.72984$ and $c = c_1 = c_2 = 1.496172$. The termination criterion for SPSO depends on whether it reaches the designated value of fitness or the fixed maximum number of fitness evaluations.

3 Analysis of PSO's Search Process

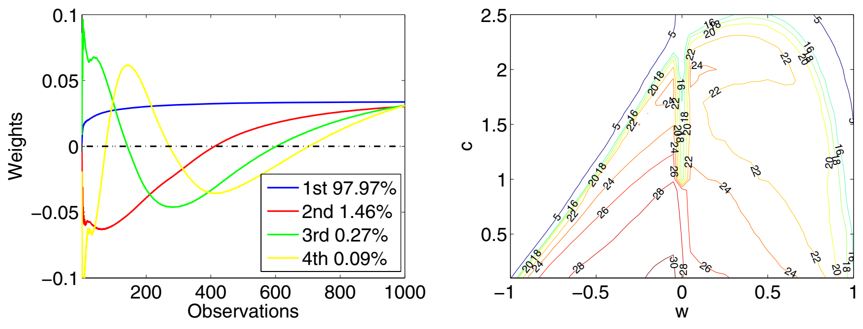
All the variants of PSO actually tried to make a more delicate trade-off between exploitation and exploration capabilities of the swarm. In [8], the authors gave quantitative definitions of the two capabilities, and offered a way to accurately analyze the PSO's search process. In the following, $P(ita)$ will be used to represent the exploitation probability for the swarm.

3.1 P(ita) Analysis for Standard PSO

In this part, we will analyze the search process of SPSO with different settings of w-c on 14 benchmark test functions [9]. The names of the functions are listed in Table 2, dimensionality of the functions is 30. For detailed information of the test functions, please refer to [9]. We choose 41 values of w in $-1 : 0.05 : 1$, and 25 values of c in $0.1 : 0.1 : 2.5$. With each set of w-c, SPSO independently run 30 times on each function.

The averaged $P(ita)$ sequences are recorded over 50 particles, 30 dimensions and 30 runs. The maximum iteration is set to be 1000, because most of the optimization processes enter the stagnation phase or reach convergence in about 1,000 iterations on the selected functions. As a result, we get $14 * 41 * 25 = 14,350$ $P(ita)$ sequences, and the size of the $P(ita)$ matrix is $14350 * 1000$.

The principal component analysis is used to analyze the $P(ita)$ matrix. Fig. 1(a) shows the four eigenvectors for the first four principal components. The accumulated contribution rate of the four principal components is 99.79%, which means they can represent most information content of the $P(ita)$ sequences. The eigenvectors are actually the weights of the pita sequences for computing the principal components.



(a) The Eigenvectors For The First Four Principal Components of The Search Process (b) The contour of the mapping between w-c and FPC

Fig. 1. The Eigenvectors and The contour of the Mapping

From Fig. 1(a), we can see that the contribution rate of the first principal component (FPC) is 97.97%. The eigenvector is positive, and the curve is monotonically increasing and tend towards stability at later stages. This curve represents the major search trend of SPSO. In the beginning, the exploitation capability of the swarm is small, and particles randomly explore the search space. As the search proceeds, exploitation job becomes more important, and particles focus on digging more precise solutions in some local spaces. The exploration job is still going on because of the random mechanism in SPSO, but it is not the dominant force anymore. In other words, the weight sequence of FPC represents the general convergence process of the swarm.

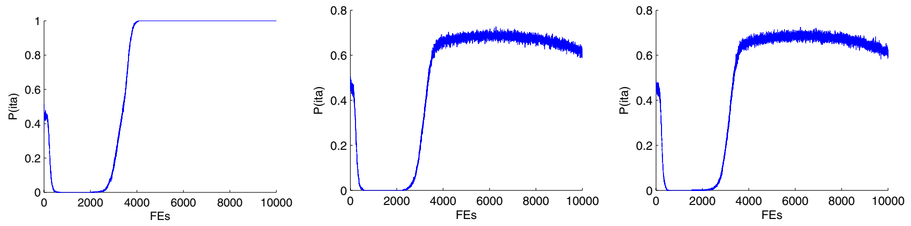
The second principal component (SPC) comprises 1.46% of the information in the $P(ita)$ matrix. At first, the weight for SPC is less than zero and still decreases.

After about 100 iterations, it begins to increase and becomes greater than zero after 400 iterations. Then it keeps increasing to the end. The trend of this curve means that, in the early stage of the search, SPC restrains exploitation. The constraining force first enhances and then declines. After about 400 iterations, SPC becomes to facilitate exploitation. Since FPC means the general trend of the search process, the SPC can represent the random factors that cause the exploration work. In fact, the other principal components can be also considered as random factors, but the effects are very weak. SPC is the most powerful of them, which can be representative. Furthermore, the eigenvector curves of the other principal components are more and more complicated, which can be considered as random noises.

In general, Fig. 1(a) tells us that even though the general trend of the search looks smooth, there exist different stages hidden in the search process. According to the eigenvector of SPC, the search process of SPSO can be divided into three stages. In the first 10% part of the search process, the random search of particles is very active and the exploration capability is strong. In the next 30% of the iterations, particles become to adjust their search to do more exploitation work, while their exploration capability gets weaker. After 40% of the iterations, exploitation becomes the dominant power to facilitate the convergence.

3.2 P(ita) Analysis for LDWPSO

In the linear decreasing inertia weight PSO (LDWPSO), the swarm has the fully connected topology, and P_{gB} is the best position found by the entire swarm. Here $c = c1 = c2 = 2$ and w decreases linearly from 1 to 0 along with the iteration. The $P(ita)$ sequences are recorded in Fig. 2.



(a) Stagnation Phase 1: $P_{pB} = P_{gB} = 1$ (b) Stagnation Phase 2: $P_{pB} = 1, P_{gB} = -1$ (c) Shifted Sphere Function

Fig. 2. The averaged $P(ita)$ sequences of the LDWPSO

In Fig. 2, the curves show the average $P(ita)$ sequences over 30 independent runs of the LDWPSO. Five particles evolve in five dimensions for 2,000 iterations (10,000 fitness evaluations (FEs)) in each run. Figs. 2(a) and 2(b) are taken during stagnation phases, the initial velocity and position of each particle are set as -3 and 10, respectively. In Fig. 2(a), The personal best position P_{pB} and the global best position P_{gB} are both held at 1, while in Fig. 2(b), P_{pB} and P_{gB} are held at ±1, respectively. Fig. 2(c) records the $P(ita)$ sequence when optimizing the shifted sphere function from Table 2.

Along with the decreasing inertia weight, the $P(ita)$ shows a tendency to decrease first and almost fade away after about 200 iterations. Then it begins to increase quickly after 800 iterations and becomes stable since. Fig. 2(a) shows that the $P(ita)$ becomes equal to 1 when PSO converges since 800 iterations. In Figs. 2(b) and 2(c), the $P(ita)$ remains stable around a certain value after about 800 iterations. However, PSO retains some exploration capability because P_{pB} and P_{gB} are not the same and the convergence is unreachable.

Fig. 2 shows that the trend of the $P(ita)$ sequence is not simply decreasing or increasing along with the linear decrease of w, and there exist different stages during the search process. In the previous 10% part of iterations, the $P(ita)$ decreases from an initial value (around 0.5) to almost 0. Most particles are considered to be doing the exploration job. During the next 30% part of iterations, particles are adjusted to gradually focus on the exploitation job. After 40% of the iterations, the particles' exploration capabilities are stably small, and the swarm becomes to converge to a stable point.

4 Adaptive Staged Particle Swarm Optimization (ASPSO)

The analyses about the search processes of SPSO and LDWPSO both show that the process can be divided into three stages. The search preferences during the three stages are different, so the setting of parameters in each stage should be considered more precisely so that the effectiveness of PSO can be improved.

4.1 The Proposal of ASPSO

Three stages of PSO's search process are defined as follows:

Definition 1. *Three stages of the search process of PSO:*

Stage 1 (Free Search Stage): *Particles have their own preference to search the space freely (First 10% of the iterations).*

Stage 2 (Adjustment Stage): *Particles find certain ways to improve their own search capabilities (10% to 40% of the iterations).*

Stage 3 (Convergence Stage): *Particles are absorbed by several points and the swarm tends to converge (40% to 100% of the iterations).*

The mapping between w-c and the FPC can be the guideline to parameter settings for search preferences in the three stages. Fig. 1(b) shows the contour of the mapping. The w-c with bigger value of FPC on the contour can facilitate the exploitation more. On the other hand, settings with small values of FPC will restrict the exploitation and enhance the exploration capability of the swarm. As can be seen, the relationship between w-c and exploitation capability of swarm is complicated, and it is not wise to use only w or c to adjust the local or global search.

Furthermore, in (1), the inertia velocity represents the current knowledge of a particle, while the cognition and the social parts give a particle new information about the environments. The parameters w and c control the balance of these two forces, and should be considered more delicately.

Table 1. Setups of ASPSO in three stages

Stages	Iterations(%)	w-c	FPC
1	0%–10%	0.2-2	24.01
2	10%–40%	0.75-1.5	21.05
3	40%–100%	0.8-1	20.49

Based on above analyses, we can summarize the ASPSO, which only set staged values of w-c on the basis of standard PSO. Theoretically, the setting of w-c in each stage depends on specific problems. Here in Table 1, we suggest a compromise strategy.

According to Table 1, in the first stage, the value of FPC is big. w is small and c is big, so the particles learn fast from cognition and social connections. In the second stage, a medial value of FPC is used, and w-c are set nearly as the same as suggested in [6]. The purpose is to balance the exploitation and exploration in this adjustment stage, since this settings of w-c is delicate according to the paper. The last stage is designed to help swarm find more precise solutions. The value of FPC is small. Since the swarm will converge or run into stagnation in this stage, particles need a bigger chance to fly out again rather than just refine the current solutions in a small space. This is even more crucial when the problem is complex with many local optima. As a result, the ASPSO tends to find more accurate solutions even faster.

4.2 Experimental Results

To make comparisons among ASPSO, SPSO and LDWPSO, the same 14 test functions that are used above are used as the objective functions. F_1 to F_5 are simple unimodal functions, F_6 to F_{12} are basic multimodal functions, F_{13} and F_{14} are extended multimodal functions. The dimensionality of test functions is set to be 30. Each of the three PSO algorithms contains 50 particles. The averaged results over 30 runs for each model

Table 2. Statistical means (M) and standard deviations (SD) of the solutions over 30 independent runs, (Sh=Shifted, Rt=Rotated, GB=Global on Bounds, Ep=Expanded)

No.	Name	ASPSO's M (SD)	SPSO's M (SD)	LDWPSO's M (SD)
F_1	Sh Sphere	**-450 (3.7e-08)** (–)	-449.999 (0.0012)	-450 (3.77e-06)
F_2	Sh Schwefel 1.2	-436.29 (37.52)	-445.85 (3.1167)	**-446.25 (2.49)** (–)
F_3	Sh Rt Elliptic	**2.20e+05 (1.88e+05)** (+)	2.75e+05 (1.17e+05)	6.43e+05 (3.64e+05)
F_4	f_2 with Noise	**-155.1012(164.02)** (+)	-9.17 (332.44)	1.85e+03 (994.06)
F_5	Schwefel 2.6 GB	**4.03e+03 (1.07e+03)** (+)	5.62e+03 (1.16e+03)	4.88e+03 (1.18e+03)
F_6	Sh Rosenbrock	**403.87 (8.31)** (+)	423.36 (75.65)	417.64(26.30)
F_7	Sh Rt Griewank	**-179.99 (0.0013)** (–)	-179.99 (0.0054)	-179.98 (0.01)
F_8	Sh Rt Ackley GB	**-119.10 (0.0741)** (–)	-119.03 (0.0771)	-119.03 (0.05)
F_9	Sh Rastrigin	**-258.03 (11.12)**(+)	-229.71 (21.55)	-247.74 (21.84)
F_{10}	Sh Rt Rastrigin	**-238.95 (15.20)** (+)	-203.14 (31.58)	-211.75 (37.25)
F_{11}	Sh Rt Weierstrass	**112.26 (1.28)** (+)	121.70 (2.34)	122.10 (1.47)
F_{12}	Schwefel 2.13	4.55e+04 (3.33e+04)	**3.07e+03 (2.86e+03)** (+)	9.12e+03 (7.69e+03)
F_{13}	Sh Ep F8F2	**-122.69 (1.55)** (+)	6.86e+03 (1.11e+04)	1.41e+04 (2.37e+04)
F_{14}	Sh Rt Scaffer F6	**-287.52 (0.27)** (–)	-287.43 (0.26)	-287.21 (0.23)

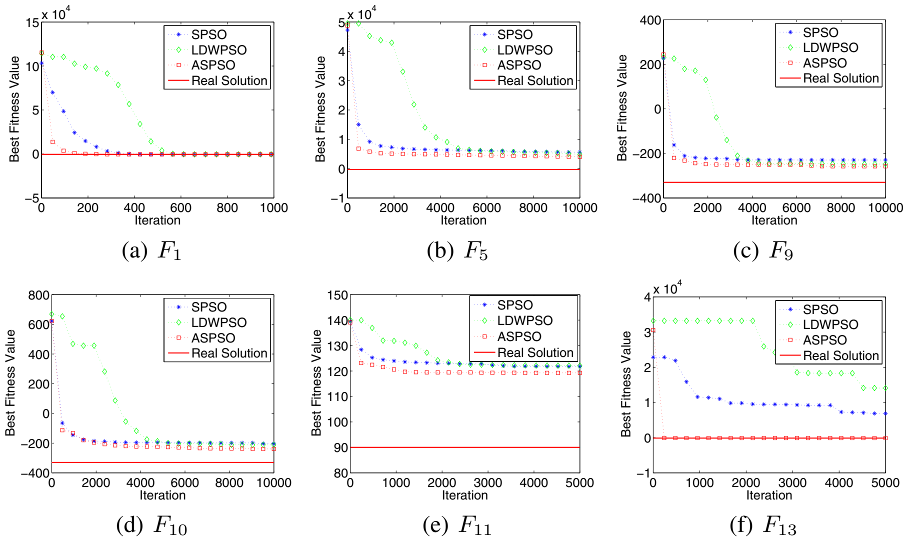


Fig. 3. The convergence curves of three PSOs

will be recorded. In LDWPSO, w linearly decreases from 0.9 to 0.4 along with the iteration [3].

Table 2 shows the statistical results of the three models. As can be seen, ASPSO achieves the best performance on most of the functions, especially on multimodal problems. We implement the variance analysis to verify the performance improvements. The symbol (+) means the performance improvement is statistical significant at 5% significance level, while (–) represents not statistical significant. ASPSO is beaten only on F_{12} by SPSO, but gets significant improvements on other eight functions. Fig. 3 shows the convergence curves on six of the functions. As can be seen, ASPSO has faster convergence speed.

The better performance of ASPSO is expected. For unimodal functions, in the first stage, the exploitation is facilitated, which speeds up local search like in hill-climbing method. In the last stage, the exploration capability is enhanced to refine the solution, especially when the bottom of the search space is relatively flat. For multimodal functions, in the first stage, local search can be speeded too, although it is not enough to find the best solution, particles can locate some local optima quickly. In the second stage, particles are helped explore wider space, under certain directions found so far. In the third stage, the global search is facilitated again and the solutions can be refined. In each stage, the swarm has its own preference, and the staged search strategy plays an important role when solving different problems.

5 Conclusion

In this paper, we proposed an adaptive staged particle swarm optimization algorithm based on the analysis about the exploitation capability of the swarm during the search process. The $P(ita)$ sequences of SPSO and LDWPSO were analyzed to conclude that the search process of PSO can be divided into three stages. The mapping between w-c and the first principal component of the $P(ita)$ sequences was given, which can be used to guide the settings of the parameters in different stages. Finally, an ASPSO was proposed, which outperformed SPSO and LDWPSO according to experimental results.

Acknowledgement

This work is supported by National Natural Science Foundation of China (NSFC), under grant number 60875080 and 60673020, and partially financially supported by the Research Fund for the Doctoral Program of Higher Education (RFDP) in China. This work is also in part supported by the National High Technology Research and Development Program of China (863 Program), with grant number 2007AA01Z453.

References

1. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proc. of the IEEE International Conference on Neural Networks, pp. 1942–1948 (1995)
2. Poli, R.: Analysis of the publications on the applications of particle swarm optimization. Journal of Artificial Evolution and Applications 2008, Article No. 4 (2008)
3. Shi, Y.H., Eberhart, R.C.: Parameter selection in particle swarm optimization. In: Porto, V.W., Waagen, D. (eds.) EP 1998. LNCS, vol. 1447, pp. 591–600. Springer, Heidelberg (1998)
4. Chatterjee, A., Siarry, P.: Nonlinear inertia weight variation for dynamic adaptation particle swarm optimization. Computers and Operations Research 33, 859–871 (2006)
5. Feng, Y., Teng, G.F., Wang, A.X., Yao, Y.M.: Chaotic Inertia Weight in Particle Swarm Optimization. In: Second International Conference on Innovative Computing, Information and Control, pp. 475–478 (2007)
6. Bratton, D., Kennedy, J.: Defining a standard for particle swarm optimization. In: Proc. of the IEEE Swarm Intelligence Symposlum(SIS), pp. 120–127 (2007)
7. Clerc, M., Kennedy, J.: The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space. IEEE Trans. on Evolutionary Computation 6(1), 58–73 (2002)
8. Zhang, J.Q., Liu, K., Tan, Y., He, X.G.: Allocation of Local and Global Search Capabilities of Particle in Canonical PSO. In: GECCO 2008, Atlanta, Georgia, USA, pp. 165–166 (2008)
9. Suganthan, P.N., Hansen, N., Liang, J.J., Deb, K., Chen, Y.P., Auger, A., Tiwari, S.: Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization (2005), `http://www.ntu.edu.sg/home/EPNSugan`

A New Particle Swarm Optimization Algorithm and Its Numerical Analysis*

Yuelin Gao, Fanfan Lei, and Miaomiao Wang

Institute of Information & System Science,
North Ethnic University, Yinchuan, 750021
gaoyuelin@263.net

Abstract. The speed equation of particle swarm optimization is improved by using a convex combination of the current best position of a particle and the current best position which the whole particle swarm as well as the current position of the particle, so as to enhance global search capability of basic particle swarm optimization. Thus a new particle swarm optimization algorithm is proposed. Numerical experiments show that its computing time is short and its global search capability is powerful as well as its computing accuracy is high in compared with the basic PSO.

Keywords: particle swarm optimization, velocity equation, numerical analysis.

1 Introduction

Particle swarm optimization (PSO) was proposed by Eberhart and Kennedy in 1995, it is a kind of swarm intelligence-based computational method [1], which comes from the study of birds foraging behavior. Since the convergence speed is fast and the setting parameters is less and the implementation is easy, PSO has been widespread attention in recent years. PSO have been widely applied in function optimization, neural network, pattern classification, fuzzy system control, and other engineering fields [2-4]. Currently, the research on PSO is mainly improving its shortcomings for being easy to fall into premature convergence. The main method to overcome the defect is to increase population diversity and to merge with other methods [5-6]. But the methods improved itself from PSO are limited.

Taking better account the impact on the current particles with the other particles, the speed equation of the basic PSO is improved by using a convex combination of the current best position of a particle and the current best position which the whole particle swarm as well as the current position of the particle, so as to enhance global search capability of basic particle swarm optimization. Thus a new particle swarm optimization algorithm is proposed. It was shown by using ten classical test functions that the proposed new PSO algorithm is better than the basic PSO in computational time and global optimization and computational accuracy.

* The work is supported by the National Natural Science Foundation of China under Grant No.60962006 and the Natural Science Foundation of Ningxia under Grant No.NZ0848.

2 Basic PSO

Like other evolution algorithms, Particle swarm optimization (PSO) [7] is able to obtain global optimal solution in complex space via collaboration and competition of individual particle. Firstly, PSO initializes particles to produce original particle swam, each particle corresponds with a feasible solution of the optimization problem and a fitness value given by the objective function. Each particle moves in solution space at a velocity to decide its distance and direction. Usually, each particle tracks current best particle to move and obtains optimal solution by each generation search finally. In each generation, the particle will track two extreme values: one is the best solution of each particle gained so far, which represents the cognition level of each particle; the other is the overall best solution gained so far by any particle in the population, which represents society cognition level.

Let n be the dimension of search space, $x_i = (x_{i1}, x_{i2}, \cdots, x_{in})$ denotes the current position of the i^{th} particle in swarm, and $p_i = (p_{i1}, p_{i2}, \cdots, p_{in})$ denotes the best position that it has ever visited. The index of the best particle among the particles in the population is represented by the symbol g ,i.e $p_g = (p_{g^1}, p_{g^2}, \cdots, p_{g^n})$ denotes the best position that the swarm have ever visited. The rate of the velocity for the i^{th} particle in the swarm is represented as $v_i = (v_{i1}, v_{i2}, \cdots v_{in})$. In simple PSO model, the particles are manipulated according to the equations

$$v_{id}(t+1) = wv_{id}(t) + c_1 r_1 (p_{id}(t) - x_{id}(t)) + c_2 r_2 ((p_{gd}(t) - x_{id}(t)), \tag{1}$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t), \tag{2}$$

where the superscript t denotes the t^{th} iteration; c_1 and c_2 are positive constants, called the cognitive and social parameter respectively, r_1 and r_2 are random numbers uniformly distributed in the range $(0,1)$; $d = 1, \cdots, n$. Let the upper limit of the rate of the velocity $v_{\max}$. When $|v_{id}| > V_{\max}$, then $|v_{id}| = V_{\max}$, and w is called inertia weight which has the ability of balancing global search and local search.

For inertia weigh, Shi Y. [8] adopt the linearly decreasing strategy , i.e.

$$w(t) = (w_{ini} - w_{end})(T_{\max} - t) / T_{\max} + w_{end}, \tag{3}$$

where the superscript t denotes the t^{th} iteration, $T_{\max}$ denotes the most iteration, w_{ini} denotes the original inertia weigh; w_{end} denotes the inertia weigh value when the algorithm process has been run the most iterations. The experiments indicate that w will impact on global search ability and local search ability, when w is higher, the global search ability is strong, but the local search ability is low; whereas the local search ability is strong, but global search ability is low. The linearly decreasing inertia weight can make PSO adjust global search and local search, but it has two defects: the first of them is that local search ability is lower in iteration forepart, even if particles have been closed to the global optimization point, they sometimes miss it, and get in local extremum. However, global search ability is lower in iteration anaphase; the second of them is that the most iteration $T_{\max}$ of equation (3) is forecasted hardly, thereby impacting on adjusting ability of PSO.

3 Improved PSO and Numerical Analysis

3.1 Improved PSO

In order to improve PSO's global search capabilities and make particle swarm have diversity, we improve the basic PSO as

$$v_{id}(t+1) = wv_{id}(t) + cr[\varphi(\lambda_1 p_{id}(t) + \lambda_2 p_{gd}(t) + \lambda_3 x_{jd}(t)) - x_{id}(t)], \tag{4}$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1), \tag{5}$$

where $\lambda_1, \lambda_2, \lambda_3 \in [0,1]$ and $\lambda_1 + \lambda_2 + \lambda_3 = 1$, In this paper ,we set $\lambda_1 = \lambda_2 = \lambda_3 = \frac{1}{3}$ i.e. take $p_{id}(t)$, $p_{gd}(t)$, $x_{jd}(t)$ position of the center of gravity. c is non-negative constant, r is random number in $[0,1]$; φ is the control factor, $x_{jd}(t)$,which is a different particle with $x_i(t)$, is a randomly generated the t^{th} generation of particles in the d^{th} dimensional coordinates of the j^{th}. The improved PSO is denoted as IPSO.

In the formula (4) , $x_j(t)$ is a randomly generated groups of different particles with $x_i(t)$, Therefore, the formula (4) takes more consideration to the group the position of other particles search performance of the algorithm, which can further enhance the PSO's global search capabilities.

As $x_{jd}(t)$ is a randomly generated particle from the particle swarm which does not contain i^{th} particle, Equation (4) gives more consideration to the impact of the current particle movement on other particles. It is seen that the formula (4) is more simply than the formula (1).

3.2 Numerical Analysis

3.2.1 Test Function

The following 10 test functions come from [9]:

Ten test functions were used to test PSO and IPSO, Iterations $T_{\max} = 1000$, population size $N = 60$,the inertia weight w is taken to be a linear decline, $w_{\max} = 0.9, w_{\min} = 0.4$. Numerical test shows that the optimization performance of PSO is best at $c_1 = c_2 = 2.0$ and the optimization performance of IPSO is best at $c = 2.5$.

Table 1. Characteristics of the various functions and parameter settings

Function	Name	Charac.	Dim.	Range	Min
$f_1(x)=\sum_{i=1}^{D} x_i^2$	Sphere	Uni-modal	30	$\lvert x_i\rvert \le 100$	0
$f_2(x)=\sum_{i=1}^{D}(\sum_{j=1}^{i} x_j)^2$	Schwefel's Problem 1.2	Uni-modal	30	$\lvert x_i\rvert \le 100$	0
$f_3(x)=\sum_{i=1}^{D}[x_i^2-10\cos(2\pi x_i)+10]$	Rastrigrin	Multi-modal	30	$\lvert x_i\rvert \le 5.12$	0
$f_4(x)=\frac{1}{4000}\sum_{i=1}^{D} x_i^2-\prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})+1$	Griewank	Multi-modal	30	$\lvert x_i\rvert \le 600$	0
$f_5(x)=-20\exp[-0.2\sqrt{\sum_{i=1}^{D} x_i^2/D}]$ $-\exp(\sum_{i=1}^{D}\cos(2\prod x_i)/D)+20+e$	Ackley	Multi-modal	30	$\lvert x_i\rvert \le 32$	0
$f_6(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	Schwefel's Problem2.22	Multi-modal	30	$\lvert x_i\rvert \le 100$	0
$f_7(x)=0.5+\frac{(\sin\sqrt{x_1^2+x_2^2})^2-0.5}{[1+0.001(x_1^2+x_2^2)]^2}$	Schafferf6	Multi-modal	2	$\lvert x_i\rvert \le 100$	0
$f_8(x)=\frac{\pi}{D}\{10\sin^2(\pi y_i)+$ $\sum_{i=1}^{D-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]$ $+(y_D-1)^2\}+\sum_{i=1}^{D}u(x_i,10,100,4)$ $y_i=1+\frac{1}{4}(x_i+1),$ $u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, x_i>a,\\ 0,-a\le x_i\le a,\\ k(-x_i-a)^m, x_i<-a\end{cases}$	——	Multi-modal	30	$\lvert x_i\rvert \le 50$	0
$f_9(x)=0.1\{\sin^2(3\pi x_1)+$ $\sum_{i=1}^{D-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]$ $+(x_D-1)^2[1+\sin^2(2\pi x_D)]\}+$ $\sum_{i=1}^{D}u(x_i,5,100,4)$	——	Multi-modal	30	$\lvert x_i\rvert \le 50$	0
$f_{10}(x)=\sum_{i=1}^{D-1}[100(x_{i+1}-x_i^2)^2+(1-x_i)^2]$	Rosenbrock	Uni-modal	30	$\lvert x_i\rvert \le 30$	0

3.2.2 The Test on Control Factor φ

It is studied how the control factor φ affects the global optimization performance of IPSO. The value range of φ is set to [0.1,1], the step size is 0.1. For each fixed control factors, IPSO runs independently 20 times, we record the average optimal fitness value, the average running time, variance for each test function. The test results are seen in Table 2 - Table 6.

Table 2. Control Factor analysis on the examples F1 and F2

Function	F1			F2		
CF	Mean	Var	CPU(s)	Mean	Var	CPU(s)
0.1	0	0	1.7375	28.8330	0.0069	1.7422
0.2	0	0	1.8086	57.6724	0.0081	1.7859
0.3	0	0	1.7992	86.5024	0.0071	1.7805
0.4	0	0	1.7852	115.3576	0.0069	1.7797
0.5	0	0	1.7820	144.2090	0.0069	1.8148
0.6	0	0	1.7703	173.0364	0.0073	1.8078
0.7	0	0	1.7656	201.8428	0.0077	1.7906
0.8	0	0	1.7547	230.5661	0.0109	1.7742
0.9	0	0	1.7406	259.1726	0.0163	1.8062
1.0	0.0101	2.894E-05	1.7375	297.5437	70.1936	1.8445

Table 3. Control Factor analysis on the examples F3 and F4

Function	F3			F4		
CF	Mean	Var	CPU(s)	Mean	Var	CPU(s)
0.1	0	0	2.0332	0	0	3.0266
0.2	0	0	2.0625	0	0	3.0375
0.3	0	0	2.0336	0	0	3.0453
0.4	0	0	2.0844	0	0	3.1250
0.5	0	0	2.0938	0	0	3.1133
0.6	0	0	2.0922	0	0	3.1656
0.7	0	0	2.1437	0	0	3.1773
0.8	0.0047	0.0001	2.1367	0	0	3.2211
0.9	2.5140	2.7198	2.1867	0.0919	0.0023	3.2539
1.0	19.7320	30.6687	2.3672	0.1706	0.0029	3.3477

Table 4. Control Factor analysis on the examples F5 and F6

Function	F5			F6		
FC	Mean	Var	CPU(s)	Mean	Var	CPU(s)
0.1	0	0	2.1938	0	0	2.1031
0.2	0	0	2.1852	0	0	2.0758
0.3	0	0	2.1945	0	0	2.1039
0.4	0	0	2.1992	0	0	2.0383
0.5	0	0	2.1914	0	0	2.0273
0.6	0	0	2.2211	0	0	2.0438
0.7	0	0	2.2141	0	0	2.0063
0.8	0	0	2.2008	41.3350	1.059E+03	1.9883
0.9	0	0	2.1922	102.5982	2.242+E03	1.9398
1.0	0.0320	1.750E-04	2.1930	251.3467	5.574E+03	1.9266

Table 5. Control Factor analysis on the examples F7 and F8

Function	F7			F8		
FC	Mean	Var	CPU(s)	Mean	Var	CPU(s)
0.1	0	0	1.0891	0.7190	0.0062	2.8289
0.2	0	0	1.1875	1.3791	0.0069	2.8289
0.3	0	0	1.1242	1.9471	0.0090	2.8312
0.4	0	0	1.1203	2.4281	0.0128	2.8547
0.5	2.00E-06	0	1.1305	2.8334	0.0174	2.8242
0.6	1.30E-05	0	1.1211	3.1142	0.0264	2.8602
0.7	4.00E-05	3.0e-11	1.1461	3.339 1	0.0335	2.9438
0.8	1.760E-04	3.700e-10	1.1398	3.4895	0.0411	2.8297
0.9	5.280E-04	2.270e-09	1.1383	3.5483	0.0507	2.8992
1.0	1.750E-04	2.939e-09	1.1289	3.5536	0.0593	2.9992

Table 6. Control Factor analysis on the examples F9 and F10

Function	F9			F10		
FC	Mean	Var	CPU(s)	Mean	Var	CPU(s)
0.1	2.8768	0.0032	2.3641	0	0	3.5430
0.2	5.7603	0.0048	2.4688	0	0	3.4930
0.3	8.4710	0.0167	2.4531	0	0	3.4789
0.4	10.9601	0.0380	2.5133	0	0	3.4531
0.5	13.2250	0.0773	2.5703	0	0	3.4258
0.6	15.2416	0.1226	2.6008	0	0	3.3977
0.7	16.7837	0.2310	2.6578	0	0	3.3641
0.8	17.9838	0.3611	2.7039	0	0	3.3289
0.9	18.6531	0.5692	2.7398	2.1E+05	0	3.3156
1.0	18.6551	0.8998	2.8531	2.284E+03	6.311E+05	3.2969

By Table 2 - Table 6, we can see that the optimal value of each function as the control factor φ decreases. When $\varphi = 0.1$, the average optimal value of each function is best, the average variance is smallest, and the average run time is shortest. Therefore, we let $\varphi = 0.1$ after using IPSO.

3.2.3 Comparison of IPSO with PSO

PSO and IPSO are tested by using above ten test functions. For each test function, each algorithm is run independently 20 times, we record the average optimal fitness value, the variance of the best fitness, and the running time. The result is seen in Table 7.

Form Table7, the average global optimal value obtained by using IPSO is better than the one obtained by using PSO for each test function in $f_1 \sim f_7$ and f_{10}, the average global optimal value obtained by using IPSO is not better than the one obtained by using PSO for each test function in f_8, f_9. For each test function, the variance obtained by using IPSO and the computing time is less than the variance obtained by using PSO and computing time.

Table 7. Two kinds of algorithms seek optimal fitness value, time and var

Algorithm Function	PSO			IPSO		
	Mean	Var	CPU(s)	Mean	Var	CPU(s)
f_1	1.6888E-06	6.5883E-12	1.8766	2.8726E-62	9.5527E-123	1.7391
f_2	54.0371	2.2173E+03	1.7867	28.8088	0.0069	1.7625
f_3	29.7990	50.4843	2.3508	0	0	2.0305
f_4	0.0200	5.1690E-04	3.2656	0	0	3.0133
f_5	2.2335E-04	1.0796E-08	2.2817	8.8818E-16	0	2.2719
f_6	96.5792	2.9249E+04	1.9773	6.9815E31	6.4159E-60	2.1484
f_7	0	0	1.0555	0	0	0.9172
f_8	0.0052	5.3737E-04	2.9375	0.7386	0.0055	2.8445
f_9	0.0011	1.1437E-05	2.8484	2.8768	0.0032	2.3641
f_{10}	4.5335E+03	1.6537E+06	4.5523	2.1497E-60	8.0085E-120	3.7531

4 Conclusion and Remarks

The speed equation of particle swarm optimization is improved by using a convex combination of the current best position of a particle and the current best position which the whole particle swarm as well as the current position of the particle, so as to enhance global search capability of basic particle swarm optimization. Thus a new particle swarm optimization algorithm is proposed. It was shown by using ten classical test functions that the new IPSO algorithm is better than the basic PSO algorithm in computational time and global optimization and computational accuracy.

References

1. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: IEEE Int. Conf. Neural Networks, Perth, Australia, vol. 11, pp. 1942–1948 (1995)
2. Zhu, J., Gu, X.S., Jiao, B.: A novel particle swarm optimization algorithm for short- term scheduling of batch plants with parallel units. International Journal of Computational Intelligence Research 4 (2008)
3. Izui, K., Nishiwaki, S., Yoshimura, M.: Enhanced multiobjective particle swarm optimization in combination with adaptive weighted gradient-based searching. Engineering Optimization 40(9), 789–804 (2008)
4. Po-Hung, C., Cheng-Chien, K., Fu-Hsien, C., Cheng-Chuan, C.: Refined Binary Particle Swarm Optimization and Application in Power System. Wseas transaction on system 8(2), 169–178 (2009)
5. Yi, T.K., Erwie, Z.: A hybrid genetic algorithm and particle swarm optimization for multimodal functions. Applied Soft Computing Journal 8, 849–857 (2008)
6. Yuelin, G., Zihui, R.: Adaptive particle swarm optimization algorithm with genetic mutation operation. In: Proceedings of the Third International Conference on Natural Computation, vol. 2, p. 211–215(2007)

7. Maurice, C.: Particle swarm optimization. Great Britain and the United States, ISTE Ltd. (2006)
8. Shi, Y., Eberhart, R.: Empirical study of particle swarm optimization. In: Proceedings of The 1999 Congress of Evolutionary Computation, pp. 1945–1950. IEEE Press, Los Alamitos (1999)
9. Yao, X., Liu, Y., Lin, G.: Evolutionary programming made faster. IEEE Trans. Evol. Comput. 3, 82–102 (1999)

A New PSO Model Mimicking Bio-parasitic Behavior

Quande Qin[1], Rongjun Li[1], Ben Niu[2], and Li Li[2]

[1] School of Business Administration, South China University of Technology, Guangzhou, China
[2] College of Management, Shenzhen University, Shenzhen, China
{qinquande,drniuben}@gmail.com, lirongjun1016@hotmail.com, lii318@163.com

Abstract. Based on the analysis of biological symbiotic relationship, the mechanism of facultative parasitic behaviour is embedded into the particle swarm optimization (PSO) to construct a two-population PSO model called PSOPB, composed of the host and the parasites population. In this model, the two populations exchange particles according to the fitness sorted in a certain number of iterations. In order to embody the law of "survival of the fittest" in biological evolution, the poor fitness particles in the host population are eliminated, replaced by the re-initialization of the particles in order to maintain constant population size. The results of experiments of a set of 6 benchmark functions show that presented algorithm model has faster convergence rate and higher search accuracy compared with CPSO, PSOPC and PSO-LIW.

Keywords: Swarm Intelligence, Particle Swarm Optimization, Parasitic behaviour, PSOPB.

1 Introduction

Particle swarm optimization (PSO) is a kind of stochastic optimization algorithm, originally motivated from the sociological behaviors associated with birds flocking [1, 2]. In the original version of PSO algorithm, the trajectory of each particle in the search space is adjusted by dynamically altering its velocity, according to two factors: each individual's best position ever found and its informants' best position ever found. [3]. Comparing with other stochastic optimization methods, the PSO has comparable or superior search performance for many hard optimization problems with faster convergence rate. It has already been widely used as a problem-solving method in many areas, such as power systems [4], artificial neural network training [5], fuzzy system control [6] and computational finance [7]. However, Angeline [8] showed that the PSO has difficulties in keeping balance between exploration and exploitation. This indicates that PSO algorithm sharply converges in the early stages of the search process, but saturates or even stagnates in the later stages. In the past decades, numerous researches provided some improved methods to overcome the drawback of trapping in the local optima. The most part of improved methods can be summarized into the following categories: tuning the parameters in the velocity and position update equations of PSO [9], designing different population topologies [10], combining

PSO with other evolutionary optimization operators [11] and adopting new learning strategies [12].

As PSO algorithm is derived from mimicking sociological behaviors of animals, incorporating the bio-inspired mechanisms into the canonical PSO may be a viable way to increase the algorithm's performance. There are some related papers concerning this topic. Based on the life cycle of biological phenomenon, Krink [13] proposed a new PSO algorithm, in which each iteration is considered as a complete life-cycle process of evolutionary individual. Inspired by co-evolutionary process between predator and prey, Silva [14] constructed a predator-prey PSO model, in which particles are divided into two categories: predator and prey, the former are to force local optima of the particles to escape in the search process, whereas the latter are subject to predator exclusion and gradually close to the global optimal solution. He [15] provided a new type of PSO (PSOPC) according to passive congregation behavior in animals. In PSOPC, information can be transferred among individuals of the whole swarm. In accordance with the bacterial chemotactic behavior, Niu [16] presents an improved PSO. In the algorithm each particle is not only attracted by its personal best position and the group's best position, but also repulsed by the worst position of itself and the whole group. From the existing literatures, incorporating the biological mechanism into PSO model has showed its efficacy and is worth studying deeply.

This paper proposed a two-population PSO Algorithm, which mimicked the bio-parasitic behavior between the host and the parasitic organism. In the present paper, we called the proposed algorithm as PSOPB. Through some benchmarks, the experimental results show that the proposed PSO model has faster convergence rate and increase the search accuracy significantly.

The rest of this paper is organized as follows. Section 2 introduces the canonical PSO models. In section 3, the bio-parasitic behavior and PSOPB model are presented. We describe the experimental settings and results in section 4. The paper is concluded in section 5.

2 Canonical PSO

In PSO, a swarm of particles are represented as potential solutions, and each particle i in the tth iterations is associated with two factors, i.e., the velocity vector $V_i^t = [v_{i1}, v_{i2}, \cdots, v_{iD}]$ and the position vector $X_i^t = \left[x_{i1}^t, x_{i2}^t, \cdots, x_{iD}^t\right]$, where D stands for the dimensions of the solution space. $x_{id} \in [l_d, u_d], d = 1, 2, \cdots D$, where l_d, u_d are the lower and upper bounds of the dth dimension, respectively. The velocity and the position of each particle are initialized by random vectors within corresponding ranges. During the evolutionary process, the swarm is manipulated according to the following equations:

$$v_{id}^{t+1} = \omega v_{id}^t + c_1 r_1 \left(pB_i^t - x_{id}^t\right) + c_2 r_2 \left(nB_i^t - x_{id}^t\right) \tag{1}$$

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

where ω, called inertia weight, is to assist with the balance between exploration and exploitation, r_1 and r_2 are random numbers, uniformly distributed with the interval $[0,1]$, and c_1 and c_2, usually set to 2.0, which determine the balance between the influence of the individual's experience and that of the swarm, are acceleration coefficients. In (1), pB_i^t, denoted as $pB_i^t = \left[pB_{i1}^t, pB_{i2}^t, \cdots, pB_{id}^t\right]$, is the best previous position with best fitness found so far by the ith particle, and nB_i^t .is the best position in the neighborhood. In the previous literatures, instead of using nB_i^t, gB^t may be used in the global-version PSO, whereas lB in the local version PSO. Generally, a maximum velocity, $V_{\max}$, is specified to control excessive roaming of particles outside the user defined search space.

The value of ω is linearly decreasing with the iterative generations as

$$\omega = \left(\omega_{start} - \omega_{end}\right) \times \left(\frac{iter_{\max} - iter}{iter_{\max}}\right) + \omega_{end} \tag{3}$$

where $iter_{\max}$ is the maximum number of allowable iterations, and $iter$ is the current iteration number, and ω_{start} and ω_{end}, usually are set to 0.9 and 0.4 [17], denote the initial and final values of the inertia weight, respectively. Hereafter, in this paper, this version of PSO is referred to as linearly decreasing inertia weigh method (PSO-LIW).

Another important variant of PSO is proposed in form of "constriction factor"(CPSO), which is an alternative method for controlling the behavior of particles in the swarm [9]. In CPSO, the velocity is updated by the following equation:

$$v_{id}^{t+1} = \lambda\left(v_{id}^t + c_1 r_1\left(pB_i^t - x_{id}^t\right) + c_1 r_2\left(nB_i^t - x_{id}^t\right)\right) \tag{4}$$

Where λ is called a constriction factor, given by:

$$\lambda = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|} \tag{5}$$

Usually the value of λ is set to 0.729 with $\varphi = c_1 + c_2 = 4.1$. c_1 and c_2 are usually both set to 2.05 . In this paper, the global-version PSO is used. Therefore, nB_i^t is replaced by gB_i^t in (1) and (4).

3 A Two – Population PSO Mimicking Bio-parasitic Behavior

3.1 Bio-parasitic Behavior

The term symbiosis, first proposed in 1879 by the German mycologist Anton de Bary [18], commonly describes close and often long-lasting physical relationships between different biological species. There are three different categories of symbiotic relationships: mutualism, commensalism and parasitism [19]. Mutualism describes a kind of symbiotic relationship that is actually beneficial to both species involved. Commensalism describes a relationship between two living organisms where one benefits and the other is not significantly harmed. Parasitism is a relationship in which one organism, known as the parasite, lives in or on another organism, known as the host, from which it derives nourishment and the host is harmed [19]. Some organisms which mostly live independent of a host but seldom hold the charge of a parasite are referred to as facultative parasites. By contrast, obligate parasites are that cannot live elsewhere except on the living protoplasm of its host.

In nature, once hosts are infected by parasites, it will deploy a set of immune mechanisms. Vertebrates infected by parasites will produce a strong immune response in the case of subjecting to the second same parasitic infection. Plants and inferior animals also can improve their immunity after infected. The co-evolutionary process between parasites and hosts commonly reduce the "negative effect" of the parasitic harm behavior, or even parasitism evolves into mutualism [20].

3.2 PSOPB Model

In the proposed PSO model (PSOPB), particles are divided into two populations: the parasites population $(Swarm^P)$, in which the number of particles denoted as N^P, and the host population $(Swarm^H)$, the size of which denoted as N^H. We believe that facultative parasitism relationship between two populations is suitable to be incorporated into PSO. It means that $Swarm^P$ obtains nourishment from $Swarm^H$ in a certain number of iterations, denoted as k. Simulation of the parasitic relationship between the two populations is described as the exchange of particles. All particles in the two populations are sorted according to their fitness value. The particles with fitness are greater than or equal to the particle with fitness of the order of $0.5(N^P + N^H)$ are classified into $Swarm^P$, and the remaining particles belong to $Swarm^H$. When $Swarm^H$ is infected by $Swarm^P$, it will produce immune response, which embodied in the proposed model is as follows: when the best particle's fitness in $Swarm^H$ is worse than that in $Swarm^P$, the *i*th particles in $Swarm^H$ fly in accordance with three directions: pB_i^H, gB^H and gB^P, where pB_i^H is the best previous position in $Swarm^H$, gB^H and gB^P represent the best position in $Swarm^H$ and $Swarm^P$, respectively. Otherwise, each particle in $Swarm^H$ evolves according the canonical PSO.

The $Swarm^H$ is harmed after parasitic behavior. In order to embody the law of "survival of the fittest" in biological evolution, the poor fitness particles with the number is set to equate $\gamma * N^H$, are removed and replaced by the re-initialization of the particles in order to maintain constant population size in . $Swarm^H$.

In this paper, PSOPB adopts the form of "constriction factor". Through the analysis above, the velocity of $Swarm^P$ is updated by the equation (4) and the velocity of $Swarm^H$ is set to update the following equations:

$$v_{id}^{t+1} = \begin{cases} \lambda\left(v_{id}^{t} + c_{11}r_1\left(pB_i^t - x_{id}^t\right) + c_{12}r_2\left(gB^H - x_{id}^t\right) + c_{13}r_3\left(gB^P - x_{id}^t\right)\right) & fgB^H < fgB^P \\ \lambda\left(v_{id}^{t} + c_1r_1\left(pB_i^t - x_{id}^t\right) + c_2r_2\left(gB^H - x_{id}^t\right)\right) & fgB^H \geq fgB^P \end{cases} \tag{6}$$

where $c_{11}, c_{12}, c_{13,} c_1$ and c_2 are acceleration coefficients, fgB^H and fgB^P are the best particle's fitness in $Swarm^H$ and $Swarm^P$, respectively, r_1, r_2 and r_3 .are random numbers uniformly distributed with range [0, 1], the meaning of other parameters in (6) is the same as (4). The pseudocode for PSOPB is listed in Table 1.

Table 1. Pseudocode for the PSOPB algorithm

```
Algorithm PSOPB
  Begin
     Randomly initialize positions and velocities of all particles
     Parameter initialization
     Set t =0
     While (the termination conditions are not met)
        Do in parallel
           For each population ( $Swarm^H$ , $Swarm^P$ )
           Evaluate the fitness value of each particle
           Update the velocity of each population
           $V_{id}^t = \min(\max(V_{id}^t, V_{\min}), V_{\max})$
           Update the position of each population
           Update $pB_{id}^t$ of each population
           Update $gB^H$ , $gB^P$ , $fgB^H$ and $fgB^P$
        End Do in parallel
           If $\mathrm{mod}(t,k) = 0$ & $t \neq 0$
              Regroup the particles according to their fitness
              Remove $\left(\gamma * N^H\right)$ particles of $Swarm^H$ with poor fitness
              Re-initialization of particles to substitute for the removed in $Swarm^H$
           End If
         Set t=t+1
      End While
   End
```

4 Experimental Studies

4.1 Benchmark Functions

A set of 6 benchmark functions are used to evaluate the performance of PSOPB with others. All benchmarks used and their parameters setting are given in Table 2.

Table 2. Benchmarks for simulation and parameter setting

Function	Mathematical representation	Search range	Range of Initialization
Sphere	$f_1(x)=\sum_{i=1}^{n} x_i^2$	$(-100,100)^n$	$(50,100)^n$
Schwefel's Problem	$f_2(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i} x_j\right)^2$	$(-100,100)^n$	$(50,100)^n$
Rosenbrock	$f_3(x)=\sum_{i=1}^{n}\left(100(x_{i+1}-x_i)^2+(x_i-1)^2\right)$	$(-30,30)^n$	$(10,30)^n$
Ackley	$f_4(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i\right)+20+e$	$(-32,32)^n$	$(10,20)^n$
Rastrigin	$f_5(x)=\sum_{i=1}^{n}\left(x_i^2-10\cos(2\pi x_i)+10\right)$	$(-10,10)^n$	$(2.56,5.12)^n$
Griewank	$f_6(x)=\frac{1}{4000}\sum_{i=1}^{n} x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	$(-600,600)^n$	$(300,600)^n$

4.2 Experimental Settings

To evaluate the performance of PSOPB, three PSO variants were used for comparisons: PSO-LIW, CPSO and PSOPC. In all PSO variants, asymmetric initialization method is adopted here, in which the population is initialized only in a portion of the search space. This is to prevent a center-seeking optimizer from ''accidentally'' finding the global optimum [11].

The parameters used for PSO-LIW, CPSO and PSOPC were recommended in [17,21,22,15].The maximum velocity $V_{\max}$ and minimum velocity $V_{\min}$ for all algorithms were set the upper bound and lower bound, respectively. The acceleration coefficients c_1 and c_2 for PSO-LIW were both 2.0. For CPSO, $c_1 = c_2 = 2.05$ was used. $c_1 = c_2 = 0.5$, $c_3 = 0.4 - 0.6$ were adopted in PSOPC. For PSOPB, $c_{11} = c_{12} = c_{13} = 1.367, c_1 = c_2 = 2.05$ in (6) were used. The inertia weight is critical for the convergence behavior of PSO-LIW and PSOPC. It is varied to follow (3), but for PSO-LIW, $\omega_{start} = 0.9$, $\omega_{end} = 0.4$ is used and whereas for PSOPC, $\omega_{start} = 0.9$,

$\omega_{end} = 0.7$. For CPSO and PSOPB, the constriction factor $\lambda = 0.729$ was adopted. In PSOPB, after the parasitic behavior occurs, the half particles of the host population were removed, that is $\gamma = 0.5$. The population size for all algorithms was set to 80. As for PSOPB, $N^H = N^P = 40$ were employed. Dimensions of all benchmark functions were set to 30. In order to prevent particles moving out the bounds, the method was used in this paper is derived from the reference [23]. To be specific, after particle *i* updates its position, its velocity is set zero and the position is limited to the bounds if the updated position is out of the search range.

In order to investigate the performance of PSOPB is sensitive to the number of iterations, k, or not, we test PSOPB with different value of k. on f_3 and f_5. The maxim iteration is set at 3,000. The average and standard deviation of the optimum solution for 20 trials are presented in Table 3. From the results; it is easy to find that the value of k counts much for the performance of POSPB. When k was equal 20, a typical unimodal function, f_3, get good performance. While $k=100$ was used, f_5 obtain good results. In the following experiments, $k = 20$ and $k=100$ were adopted to optimize the unimodal and multimodal functions, respectively.

Table 3. Results achieved with different value of k

Function	PSOPB			
k	20	50	100	150
f_3	3.8933 (4.1803)	5.0825 (5.5621)	5.3037 (4.2131)	7.5096 (4.9382)
f_5	39.8859 (8.7009)	37.5434 (9.9023)	20.2368 (7.0445)	20.9635 (7.4685)

4.3 Numerical Results and Comparisons

The experiment runs 30 times independently for all PSO variants on the set of 6 benchmark functions and the maximum iteration is set at 6,000. The mean values and standard deviation of the results are presented in the Table 4, in which numbers in bold represent are the comparatively best values. The graphs of the average best fitness value of the base-10 logarithm with the evolution of iterations are presented in Figs. 1–6. From Table 4 and Figs 1-6, we can observe that PSOPB obtains the remarkable performance. It is clear that PSOPB has a faster convergence rate and higher search accuracy than CPSO, PSO-LIW and PSOPC for both unimodal and multimodal problems. The most concerning thing is that PSOPB has sustainable searching ability for f_3 and f_5 which is prone to lead to canonical PSO trap in the local optima.

Table 4. Numerical results of all PSO algorithms

`Function	Indicators	variants of PSO compared			
		PSO-LIW	CPSO	PSOPB	PSOPC
f_1	Mean	2.0797e-045	2.5647e-145	**1.1543e-147**	9.9896e-101
	Std	7.2323e-045	3.8777e-145	**3.5598e-147**	2.2337e-100
f_2	Mean	0.58971	1.0991e-017	**6.4512e-020**	0.66863
	Std	0.44475	1.8367e-017	**1.0711e-019**	0.53359
f_3	Mean	30.7252	6.1971	**0.4785**	23.5664
	Std	26.1358	3.8022	**1.2452**	4.4037
f_4	Mean	6.2172e-015	0.4765	**5.8624e-015**	3.1086e-014
	Std	5.4153e-015	0.7744	**1.1235e-015**	4.8643e-014
f_5	Mean	19.8333	43.6796	**11.7372**	27.6621
	Std	**5.3515**	11.4073	7.9732	6.7261
f_6	Mean	0.0150	0.0118	**0.0049**	0.0098
	Std	0.0221	0.0107	**0.0059**	0.0071

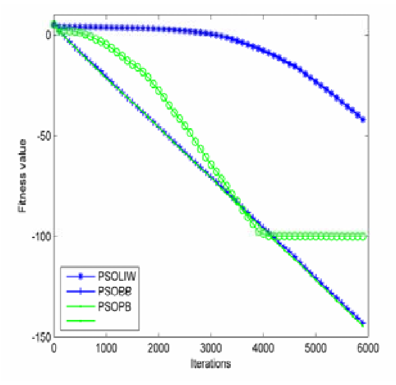


Fig. 1. f_1 Sphere function

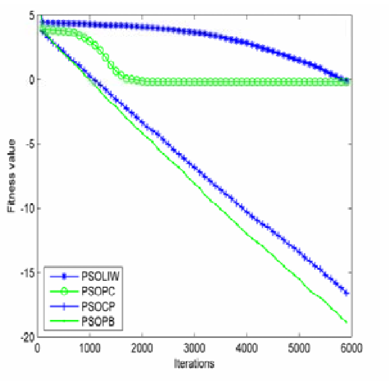


Fig. 2. f_2 Schwefel's Problem

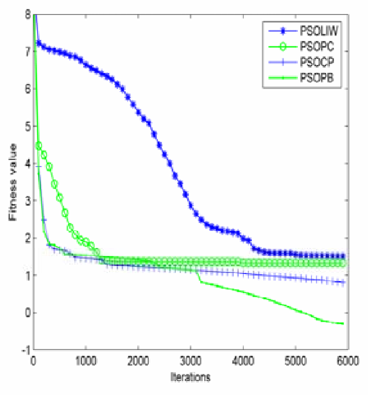


Fig. 3. f_3 Rosenbrock function

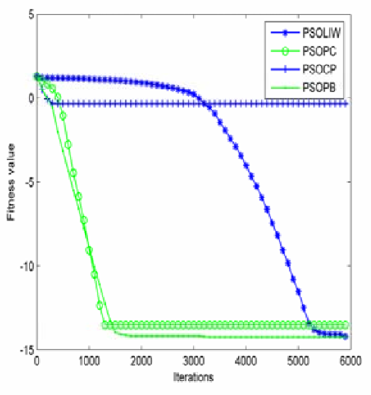


Fig. 4. f_4 Ackley function

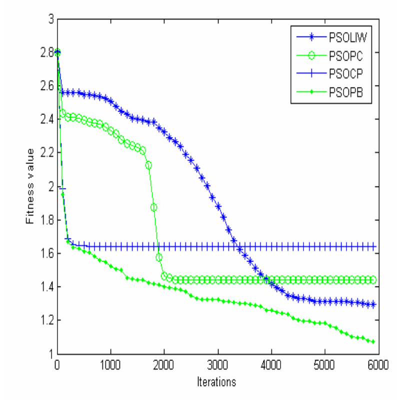


Fig. 5. f_5 Rastrigin function

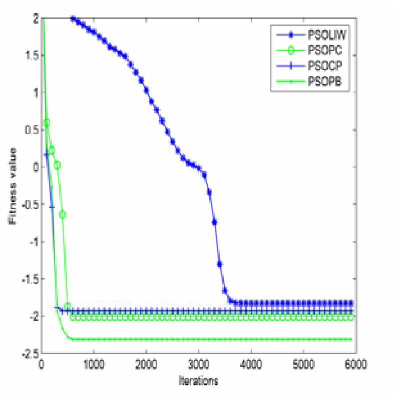


Fig. 6. f_6 Griewank function

5 Conclusions

In this paper, a novel PSO mimicking bio-parasitic behaviour was proposed. By mimicking facultative parasitic behaviour between the host population and the parasites population, the two populations exchange particles according to fitness. On the basis of analysis parasitic mechanism, the immune react of the host population infected is embedded into the PSO model. The law of "survival of the fittest" in biological evolution is also demonstrated in the host population.

A set of 6 benchmark functions have been tested PSOPB in comparison with PSO-LIW, CPSO and PSOPC. The experimental results show that PSOPB has a faster convergence rate and higher search accuracy for all the benchmark functions. Through simulating the co-evolutionary process between the host and the parasites population, the results demonstrate the efficacy of incorporating the parasitic behavior into the canonical PSO model. In the future, we will study how to use PSOPB to optimize practical engineering optimization problems.

Acknowledgements. This work is partly supported by Shenzhen/Hong Kong innovation circle project under Grant SG200810220137A and Natural Science Foundation of Guangdong Province under Grant 9451806001002294.

References

1. Eberchart, R.C., Kennedy, J.: Particle Swarm Optimization. In: IEEE International Conference on Neural Networks, Perth, Australia (1995)
2. Eberchart, R.C., Kennedy, J.: A New Optimizer Using Particle Swarm Theory. In: Proc. of the 6thInternational Symposium on Micromachine and Human Science, Nagoya, Japan, pp. 39–43 (1995)
3. Shi, Y., Eberchart, R.C.: A Modified Particle Swarm Optimizer. In: IEEE Congress on Evolutionary Computation, Anchorage, AK, NJ, pp. 69–73 (1998)

4. Park, J.B., Lee, K.S., Shin, J.R., Lee, K.Y.: A Particle Swarm Optimization for Economic Dispatch with Nonsmooth Cost Functions. IEEE Transaction on Power System 20, 34–42 (2005)
5. Mendes, R., Cortez, P., Rocha, M., Neves, J.: Particle Swarms for Feedforward Neural Network Training. In: International Joint Conference on Neural Networks, pp. 1895–1899 (2002)
6. Parsopoulos, K.E., Papageorgiou, E.I., et al.: A first Study of Fuzzy Cognitive Maps Learning Using Particle Swarm Optimization. In: Proc. of IEEE Congress on Evolutionary Computation, Canberra, Australia, pp. 1440–1447 (2003)
7. Cura, T.: Particle Swarm Optimization Approach to Portfolio Optimization. Nonlinear Analysis: Real World Applications 10, 2396–2406 (2009)
8. Angeline, P.J.: Evolutionary Optimization versus Particle Swarm Optimization and Philosophy and Performance Difference. In: Proc. of 7th Annual Conference on Evolutionary Programming, San Diego, USA, pp. 601–610 (1998)
9. Clerc, M., Kennedy, J.: The particle swarm: explosion, stability, and convergence in multi dimensional complex space. IEEE transaction on Evolutionary Computation 6, 58–73 (2002)
10. Suganthan, P.N.: Particle Swarm Optimizer with Neighborhood Operator. In: Proc. of the IEEE Congress of Evolutionary Computation, pp. 1958–1961 (1999)
11. Angeline, P.J.: Using Selection to Improve Particle Swarm Optimization. In: Proc. of IEEE World Congress on Computational Intelligence, Anchorage, Alaska, pp. 84–89 (1998)
12. Mendes, R., Kennedy, J., Neves, J.: The Fully Informed Particle Swarm: Simpler, be Better. IEEE transaction on Evolutionary Computation 8, 204–210 (2004)
13. Krink, T., Løvbjerg, M.: The Life Cycle Model: Combining Particle Swarm Optimisation, Genetic Algorithms and Hillclimbers. In: Guervós, J.J.M., Adamidis, P.A., Beyer, H.-G., Fernández-Villacañas, J.-L., Schwefel, H.-P. (eds.) PPSN 2002. LNCS, vol. 2439, pp. 621–630. Springer, Heidelberg (2002)
14. Silva, A., Neves, A., Costa, E.: An Empirical Comparison of Particle Swarm and Predator Prey Optimisation. In: O'Neill, M., Sutcliffe, R.F.E., Ryan, C., Eaton, M., Griffith, N.J.L. (eds.) AICS 2002. LNCS (LNAI), vol. 2464, pp. 103–110. Springer, Heidelberg (2002)
15. He, S., Wu, Q.H., Wen, J.Y., et al.: A Particle Swarm Optimizer with Passive Congregation. BioSystems 78, 135–147 (2004)
16. Niu, B., Zhu, Y.L., et al.: An Improved Particle Swarm Optimization Based on Bacterial Chemotaxis. In: Proc. of the 6th World Congress on Intelligent Control and Automation, Dalian, China, pp. 3193–3197 (2006)
17. Shi, Y., Eberhart, R.C.: Parameter Selection in Particle Swarm Optimization. In: Porto, V.W., Waagen, D. (eds.) EP 1998. LNCS, vol. 1447, pp. 591–600. Springer, Heidelberg (1998)
18. Ahmadjian, V., Paracer, S.: Symbiosis: An Introduction to Biological Associations. Oxford University Press, Oxford (2000)
19. Douglas, A.E.: Symbiotic Interactions. Oxford University Press, Oxford (1994)
20. Li, W.X., Wang, G.T.: Regulation of Parasites on Host Populations: A Brief. Acta Hydrobiologica Sinica 26(5), 550–554 (2002)
21. Kennedy, J., Eberchart, R.C.: Empirical Study of Particle Swarm Optimization. In: Proc. of Congress on Evolutionary Computation, Washington, DC, pp. 1945–1949 (1999)
22. Eberhart, R.C., Shi, Y.: Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization. In: Proc. of IEEE International Congress on Evolutionary Computation, San Diego, CA, pp. 84–88 (2000)
23. Carlisle, A., Dozier, G.: An Off-The-Shelf PSO. In: Proc. of the workshop on Particle Swarm Optimization, Indianapolis, USA (2001)

KNOB Particle Swarm Optimizer

Junqi Zhang[1], Kun Liu[2], and Ying Tan[2,⋆]

[1] Key Laboratory of Embedded System and Service Computing, Ministry of Education, Department of Computer Science and Technology, Tongji University, Shanghai, 200092, China
[2] Key Laboratory of Machine Perception, Ministry of Education, Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, Beijing, 100871, China
ytan@pku.edu.cn

Abstract. It is not trivial to tune the swarm behavior just by parameter setting because of the randomness, complexity and dynamic involved in particle swarm optimizer (PSO). Hundreds of variants in the literature of last decade, brought various mechanism or ideas, sometimes also from outside of the traditional meta-heuristics field, to tune the swarm behavior. While, in the same time, additional parameters have to be afforded. This paper proposes a new mechanism, named KNOB, to directly tune the swarm behavior through parameter setting of PSO. KNOB is defined as the first principal component of the statistical probability sequence of exploration and exploitation allocation along the search process. The using of the KNOB to tune PSO by parameter setting is realized through a statistical mapping, between the parameter set and the KNOB, learned by a radial basis function neural network (RBFNN) simulation model. In this way, KNOB provides an easy way to tune PSO directly by its parameter setting. A simple application of KNOB to promote is presented to verify the mechanism of KNOB.

Keywords: Particle Swarm Optimizer, Exploitation, Exploration, search Strategy, KNOB.

1 Introduction

Particle swarm optimizer is a stochastic global optimization technique based on a social interaction metaphor [1,2]. Because of the complexity, dynamic nature and randomness involved in the PSO, it is hard to directly balance exploitation and exploration by parameter selection. Many heuristic methods are introduced into PSO to tune the balance of search strategy. However, more parameters are usually led into PSO simultaneously with the introduction of the heuristic methods.

This paper proposes a novel KNOB PSO (KPSO) to directly tune the balance between exploitation and exploration in particle swarm just by parameter selection. KNOB is defined as the first principal component of the statistical probability sequence of exploration and exploitation allocation along the search process. The using of the KNOB to tune PSO by parameter setting is realized by a statistical mapping, between the parameter set and the KNOB, learned by a radial basis function neural network

⋆ Corresponding author.

(RBFNN) simulation model. In this way, KNOB provides an easy way to tune PSO directly by its parameter setting, which is verified through a simple application of KNOB.

The remainder of this paper is organized as follows. Section 2 briefly review the PSO. Section 3 defines the KONB. Section 4 elaborates the radial basis function neural network (RBFNN) simulation model to learn the statistical mapping between the parameter set and the KNOB. A simple use of KNOB is verified in Section 5. Concluding remarks are drawn in Section 6.

2 PSO

In each iteration, the velocity V and the position X of each particle are updated according to its own previous best position and the best position of its neighbors in the swarm so far. The constricted update rules are

$$\begin{aligned} V_{id}(t+1) &= \chi(V_{id}(t) + c_1 r_1 (P_{iBd}(t) - X_{id}(t)) \\ &\quad + c_2 r_2 (P_{gBd}(t) - X_{id}(t))), \qquad (1) \\ X_{id}(t+1) &= X_{id}(t) + V_{id}(t+1). \qquad (2) \end{aligned}$$

where $i = 1, 2, \cdots, n$, i is the number of particles in the swarm, $d = 1, 2, \cdots, D$, and d is the dimension of solution space. The learning factors c_1 and c_2 are nonnegative constants, r_1 and r_2 are random numbers uniformly drawn from the interval $[0, 1]$, which are all scalar quantities for each particle in each dimension. P_{iBd} and P_{gBd} are the locations of the best positions found so far by particle i and its neighbors in dimension d, respectively. The constriction coefficient χ is defined as

$$\chi = \frac{2}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|}, \varphi = c_1 + c_2. \qquad (3)$$

It is suggested by Bratton and Kennedy that the values of these constants should be set as: $c_1 = c_2 = 2.05$, $\varphi = 4.1$ and $\chi \approx 0.72984$, so that the convergence of the model can be ensured. Our analyses are based on this standard model, except we have substituted the update rules as in Eqs. (4) and (5) for convenience. This is because a PSO with constriction [3] is algebraically equivalent to a PSO with inertia weight [4].

$$\begin{aligned} V_{id}(t+1) &= wV_{id}(t) + c_1 r_1 (P_{iBd}(t) - X_{id}(t)) \\ &\quad + c_2 r_2 (P_{gBd}(t) - X_{id}(t)), \qquad (4) \\ X_{id}(t+1) &= X_{id}(t) + V_{id}(t+1). \qquad (5) \end{aligned}$$

In Eq. (4), the parameter $w \in [-1, 1]$ is the inertia weight for the velocity $V_{id}(t)$. Set $w \approx 0.72984$ and $c_1 = c_2 = 1.496172$. The other denotations are with the same meanings as in Eq. (1).

3 KNOB

In this section, the KNOB is defined based on the definitions of the probabilities of the exploitation and exploration allocated in particle swarm. To quantify the search strategy in Particle Swam, in [5], we give the following intuitive definition of the exploitation probability where f_X is the probability density of the particle position sampling and $A(ita)$ is the exploitation area.

Definition 1 (Exploitation Probability). The probability lying in *A(ita)* in the next iteration is defined as

$$P(ita) = \int_{A(ita)} f_X(x)dx. \tag{6}$$

Based on the definition of $P(ita)$, one has the probability of exploration $P(ra)$

$$P(ra) = 1 - P(ita) \tag{7}$$

which enables our calculations to concentrate only on the $P(ita)$ sequences as the search strategy in particle swarm in the following analyses and discussions. In order to further show how the parameter sets of PSO affect the search strategy, we use the $P(ita)$ sequence which will be defined in Definition 2 to elaborate how PSO allocate exploitation and exploration along the search process with different parameter sets.

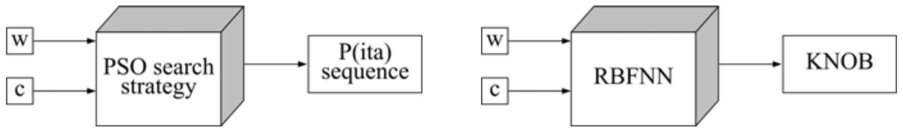
Definition 2 ($P(ita)$ sequence). $P(ita)$ sequence is a set of statistical $P(ita)$s along the search process. Each statistical $P(ita)$ is calculated based on all $P(ita)$s in every dimensions of all particles in the swarm at each iteration, where the $P(ita)$ is defined in Definition 1.

We attempt to seek a simple and feasible statistic value which can represent the $P(ita)$ sequence with different parameter sets to the utmost extent. We speculate on that the *first principal component* [6] of a $P(ita)$ sequence may be help. In fact, we find in the experiments that the contribution rate of *first principal component* is 98.17%, which represents almost all the information content of the $P(ita)$ sequences. Besides, it provides a simple statistical value to represent the $P(ita)$ sequences with different parameter sets. Therefore, we give the following definition. Under this definition, the KNOB could be used as a balance mechanism for the search strategy in particle swarm by parameter selection.

Definition 3 (KNOB). The *first principal component* of a $P(ita)$ sequence is defined as a KNOB to indicate the search strategy in PSO by the synthesis of the exploitation capability along the whole search process.

4 RBFNN Model

In the PSO model conceptualized in Fig. 1(a), the inertia weight (w) and the acceleration coefficient (c, set $c = c_1 = c_2$) are two input factors, each parameter set derives a $P(ita)$ sequence along search process according to the unknown inner search strategy. In order to directly use parameter selection to tune the KNOB, we propose a simulation model based on RBFNN shown in Fig. 1(b). The RBFNN aims to discover the unknown inner search strategy of the PSO model and results in a mapping from the parameters w and c to the KNOB by statistical learning. This simulation model can then guide users to tune the KNOB straight forward by parameter selection.



(a) PSO Model (b) RBFNN Simulation Model

Fig. 1. PSO model (a) and our proposed RBFNN simulation model (b)

Table 1. Setup of parameters for samples

Parameters	*Values*
p	50
d	30
w	$-1 : 0.05 : 1$
c	$0.1 : 0.1 : 2.5$

In the following, first, we run constricted PSO with different parameter setups of w-c on serval typical benchmark functions. The results of corresponding $P(ita)$ sequences are saved. Then, we employ the principal component analysis (PCA) [6] method to extract the *first principal component* (KNOB) from each $P(ita)$ sequence. Finally, the RBFNN learns from the samples composed by these parameter setups and their corresponding KNOBs to simulate the PSO search strategy in Fig. 1(a) and gives the predictive results.

According to [7], we set $p = 50$ for the PSO and $d = 30$ for every function. In order to have enough samples to train the RBFNN in Figure 1(b), we choose four benchmark functions from the $CEC'05$ [8], which are *Shifted Schwefel's Problem 1.2 with Noise in Fitness, Shifted Rastrigin's Function, Shifted Rotated Rastrigin, Shifted Rotated Expanded Scaffer's F6*. Different w and c values are chosen. The parameter setups for samples are shown in Table 1. There are 4,100 samples in total, each of which is the averaged value over 20 independent runs. The results of the $P(ita)$ sequences are recorded for 1,000 iterations, because most searches enter the stagnation phase or reached convergence in about 1,000 iterations on the selected benchmark functions. The data samples are divided into two parts, $Cross - Validation$ and $test$ data sets. $Cross - Validation$ data is further divided into a training set and a validation set to train the RBFNN through cross validation with 2,575 training samples and 500 validation samples randomly drawn from these 3,075 samples each time. The $Test$ data derived from *Shifted Rotated Rastrigin*, which is not used in the cross validation, is chosen to test the generalization performance of the trained RBFNN.

As a result, we can use the trained RBFNN to predict the mapping of KNOB in w-c plane. This will give users a straight-forward guide to choose suitable parameter setups that correspond to an appropriate balance between exploitation and exploration for a specific problem, and allow for a dramatic improvement on the performance of PSO. In order to obtain this landscape, parameter samples are set as: $p = 50$, $d = 30$, $c \in (0, 2.5]$ and $w \in [-1, 1]$. The averaged prediction results of 30 trained RBFNNs

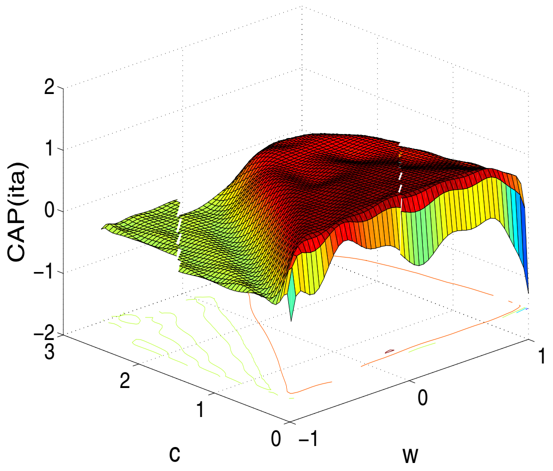


Fig. 2. The KNOB for w-c plane when $p = 50$ and $d = 30$

are shown in Fig. 2, which shows the surface of the KNOB over different w and c, and gives the overall effect of the parameter setups.

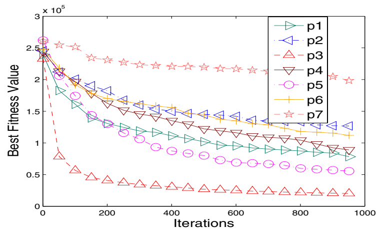
5 A Simple Application of KNOB

In this application, we aim at using the proposed KPSO to search for a parameter set that performs better than the constriction based parameter set suggested by Clerc [3]) (denoted as SPSO) on the benchmark functions *Shifted Schwefel's Problem 1.2 with Noise in Fitness, Shifted Rastrigin's Function, Shifted Rotated Rastrigin, Shifted Rotated Expanded Scaffer's F6.*

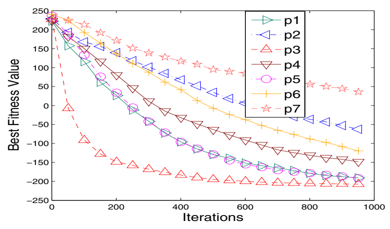
We first select the constriction based parameter set suggested by Clerc [3]) as the parameter set whose KNOB equal to 21.55 predicted by the mapping shown in Fig. 2 and is used as the current KNOB. The KNOB granularity is set as 1. Given the KNOB and the KNOB granularity, six KNOBs are evenly chosen around 21.55 as shown in Table 2. The parameter set 4 is the constriction based parameter set suggested by Clerc [3]).

Table 2. The mapped parameter sets according to the tuned KNOBs and their KNOB

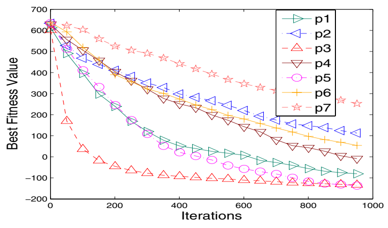
$ParameterSets$	w	c	KNOB
1	0.15	1.8	24.55
2	0.61	1.79	23.55
3	0.47	1.42	22.55
4	0.72984	1.496172	21.55
5	0.84	1	20.55
6	0.885	0.9	19.55
7	0.93	0.66	18.55



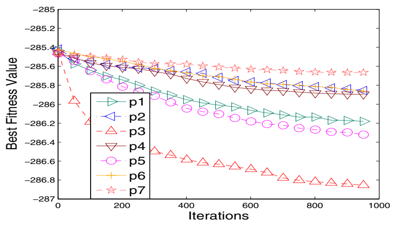
(a) Shifted Schwefel's Problem 1.2 with Noise in Fitness



(b) Shifted Rastrigin's Function



(c) Shifted Rotated Rastrigin



(d) Shifted Rotated Expanded Scaffer's F6

Fig. 3. The averaged best fitness sequences of the PSO with fifty particles in thirty dimensions using the parameter sets listed in Table 2 when optimizing the *Shifted Schwefel's Problem 1.2 with Noise in Fitness, Shifted Rastrigin's Function, Shifted Rotated Rastrigin and Shifted Rotated Expanded Scaffer's F6* over 50 independent runs. The swarm evolves for 1000 iterations in each run.

The learned mapping in Fig. 2 is then used to select the corresponding parameter sets of selected KNOBs.

The results of the averaged convergence curves using the parameter sets in Table 2 are recorded in 1,000 iterations over 50 runs on the benchmark test functions and shown in Fig. 3. These functions are all minimization problems and a lower value represents a better solution.

As shown in Fig. 3, the PSO with parameter sets 1, 3 and 5 converge fastest on all of the benchmark functions. The PSO with parameter set 3 converges faster than the parameter set 4 used in the SPSO on all the benchmark functions. Therefore, parameter set 3 is selected as the parameters of the KPSO.

Furthermore, in order to verify the effectiveness and efficiency of the KPSO, the statistical means and standard deviations of the obtained solutions of the benchmark functions are provided in Table 3 for the comparison between the KPSO and SPSO over 50 independent runs. It can be seen from the averaged solutions that the proposed KPSO outperforms the SPSO on all of the benchmark functions.

It can be concluded from the comparisons that the parameter set 3 not only has faster convergence speed, but also has more accurate optimal solutions on all of the benchmark functions than the SPSO. Because we have found a parameter set that performs significantly better than the constriction based parameter set suggested by Clerc [3]) on all benchmark functions, the KPSO algorithm terminates.

Table 3. Statistical means and standard deviations of the solutions of benchmark functions given by the KPSO and the SPSO over 50 independent runs

$Func.$	**KPSO's M ± S**	**SPSO's M ± S**
Shifted Schwefel's Problem 1.2 with Noise in Fitness with Noise in Fitness	**1.9659e+004± 6.5447e+003**	8.7721e+004± 6.6138e+004
Shifted Rastrigin's Function	**-207.9337 ± 30.8512**	-153.5789± 131.6026
Shifted Rotated Rastrigin	**-135.4692± 47.2030**	-20.0018± 215.6167
Shifted Rotated Expanded Scaffer's F6	**-286.8861± 0.4103**	-285.9057 ± 0.4530

6 Conclusion

This paper proposed a new mechanism, named KNOB, to directly tune the swarm behavior through parameter setting of PSO. First, the KNOB was defined as the first principal component of the statistical probability sequence of exploration and exploitation allocation along the search process. The using of the KNOB to tune PSO by parameter setting has been realized by a statistical mapping, between the parameter set and the KNOB, learned by a radial basis function neural network (RBFNN) simulation model. In this way, KNOB provided a simple statistical value to represent the $P(ita)$ sequences with different parameter sets. A simple use of KNOB verified that the KNOB could be used as a easy balance mechanism for the search strategy in particle swarm through its parameter setting.

Acknowledgment. This work is supported by the National Natural Science Foundation of China (NSFC), under Grants No. 60803065, 60875080, 60673020, 60803032, 60970061 and 60804042, and partially supported by the Spacial Grade of the Financial Support from China Postdoctor Science foundation under Grant No. 200801029. This program is also financially supported by the National High Technology Research and Development Program of China (863 Program) under Grants No. 2007AA01Z453 and 2009AA01Z141, the National Basic Research Program of China(973 Program) under Grant No. 2010CB328100, the Program for Changjiang Scholars and Innovative Research Team in University, and the Research Fund for the Doctoral Program of Higher Education (RFDP) in China, Program for Young Excellent Talents in Tongji University under Grant No. 2009KJ032 and Shanghai Educational Development Foundation under Project 2008CG23.

References

1. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proc. of the IEEE International Conference on Neural Networks, Perth, Australia, pp. 1942–1948 (1995)
2. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: Proc. of the 6th Int. Symp. Mcro Machine Human Science, Nagoya, Japan, pp. 39–43 (1995)
3. Clerc, M., Kennedy, J.: The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. on Evolutionary Computation. 6, 58–73 (2002)

4. Poli, R., Broomhead, D.: Exact analysis of the sampling distribution for canonical particle swarm optimiser and its convergence during stagnation. In: Proc. of the IEEE International Conference on Genetic And Evolutionary Computation Conference, London, England, pp. 134–141 (2007)
5. Zhang, J., Liu, K., Tan, Y., He, X.G.: Allocation of local and global search capabilities of particle in canonical pso. In: Proc. of Genetic and Evolutionary Computation Conference, pp. 165–166 (2008)
6. Jolliffe, I.T.: Principal component analysis. Springer, Berlin (1986)
7. Bratton, D., Kennedy, J.: Defining a standard for particle swarm optimization. In: Proc. of the IEEE Swarm Intelligence Symposium (SIS), pp. 120–127 (2007)
8. Suganthan, P.N., Hansen, N., Liang, J.J., Deb, K., Chen, Y.P., Auger, A., Tiwari, S.: Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization (2005), http://www.ntu.edu.sg/home/EPNSugan

Grouping-Shuffling Particle Swarm Optimization: An Improved PSO for Continuous Optimization

Yinghai Li*, Xiaohua Dong, and Ji Liu

College of Hydraulic & Environmental Engineering, China Three Gorges University, Yichang, Hubei, 443002, China
lyh307@tom.com, {xhdong,liuji}@ctgu.edu.cn

Abstract. This paper proposes a novel population-based evolution algorithm named grouping-shuffling particle swarm optimization (GSPSO) by hybridizing particle swarm optimization (PSO) and shuffled frog leaping algorithm (SFLA) for continuous optimization problems. In the proposed algorithm, each particle automatically and periodically executes grouping and shuffling operations in its flight learning evolutionary process. By testing on 4 benchmark functions, the numerical results demonstrate that, the optimization performance of the proposed GSPSO is much better than PSO and SFLA. The GSPSO can both avoid the PSO's shortage that easy to get rid of the local optimal solution and has faster convergence speed and higher convergence precision than the PSO and SFLA.

Keywords: Particle swarm optimization, Shuffled frog leaping algorithm, Evolution strategy, Continuous optimization.

1 Introduction

In the past two decades, with the development of optimization technology, swarm intelligence meta-heuristics are proved to be successful approaches to solve complex continuous global optimization problems [1]. As a typical swarm intelligence algorithm, particle swarm optimization algorithm (PSO) has been widely research an application in the academia and engineering [2,3]. In the algorithm, the individual adjusts its flight path efficiently and flexibly according to both its own and the population historical best position. However, PSO has the shortages of premature convergence, poor accuracy and easy to fall into local optimal solution, which prompt us to explore methods to weaken or avoid the shortcomings and improve its performance.

Shuffled Frog Leaping Algorithm (SFLA) is a newly proposed meta-heuristic computing technology based on the memetic algorithm which is firstly used to solve the water distribution system design problem [4] and has been verified that efficient to many global optimization problems [5,6]. SFLA is based on grouping evolution of memes carried by individuals and global exchange of information within the population. The grouping evolution allows individuals to deep search in different direction in the feasible space and the shuffling process allows the full exchange of information

* Corresponding author.

between individuals to avoid the algorithm fall into a local optimum. Emphasis on both global and local search strategies, which is one of the SFLA's major advantages [7].

Inspired by the grouping-shuffling evolution mechanism of SFLA greatly help to improve individual's convergence and global search abilities, we combine the grouping-shuffling evolution mechanism with PSO's efficient and flexible individual flight learning strategy to develop a novel grouping-shuffling particle swarm optimization algorithm (GSPSO). In the proposed algorithm, the whole particles repeated carry out grouping evolution and shuffling operation. In each group, the worst particle is evolution with the PSO's flight learning strategy. The grouping-shuffling evolution mechanism can prevent the particle's search process from falling into the local optimal solution and becoming premature convergence.

The rest of this paper is organized as follows: In Section 2, we briefly explain the basic principles of PSO and SFLA. In Section 3, we propose the new modified PSO based on SFLA evolution framework. In Section 4, through numerical experiments and comparison, we verify the performance of the proposed GSPSO. Finally, we conclude this paper in Section 5.

2 The Standard PSO and SFLA Algorithms

Before actually proposing the improved particle swarm optimization based on SFLA evolution framework, to make the paper self-explanatory, the principle of standard PSO and SFLA are briefly explained in the following.

2.1 Standard Particle Swarm Optimization

The PSO algorithm, proposed by James Kennedy and Russell Eberhart in 1995, is a swarm intelligent algorithm inspired by the social behavior of a flock of birds find food [2,3]. PSO has been originally used to handle continuous optimization problems, with the advantages of simple concept, high performance and easy programming.

In PSO, each solution is a 'bird', which is referred to as a 'particle'. Suppose in the *D*-dimensional search space, the population of particle is *P*, the *i*-th particle's current position vector and velocity vector are respectively expressed as $X_i = (x_{i1}, x_{i2}, \cdots, x_{iD})$ and $V_i = (v_{i1}, v_{i2}, \cdots, v_{iD})$, the best historical positions of itself and swarm are expressed as $P_i = (p_{i1}, p_{i2}, \cdots, p_{iD})$ and $P_g = (p_{g1}, p_{g2}, \cdots, p_{gD})$. The velocity V_i and the position X_i of the *i*-th particle are updated according to the experience of itself and neighboring particles as the following equations(1)(2). At the end of defined iterations, the best particle *Pg* in swarm is the best solution of the problem.

$$v_{id}(g+1) = w \times v_{id}(g) + c_1 \times r_1 \times [p_{id}(g) - x_{id}(g)] + c_2 \times r_1 \times [p_{gd}(g) - x_{id}(g)] \tag{1}$$

$$x_{id}(g+1) = x_{id}(g) + v_{id}(g+1) \tag{2}$$

Where c_1 and c_2 are the two positive constants, called acceleration coefficients, r_1 and r_2 are the two random numbers in the range [0,1], *w* is called inertia weight which can be automatically adjusted according to the iteration number such as $w(g) = 0.9 - (g / G) / 2$, *g* is the iteration number, *G* is the max iteration number.

2.2 Standard Shuffled Frog Leaping Algorithm

The SFLA algorithm, proposed by Eusuff and Lansey in 2000, is a memetic intelligent algorithm inspired by natural memetics [8]. The SFLA consists of a population of frogs partitioned into different group which called memeplexes. The frogs act as hosts of memes where a meme is a unit of cultural evolution. The algorithm performs simultaneously an independent local search in each memeplex [9]. To ensure global exploration, the frogs are periodically shuffled and regrouped into new memplexes. The local search and the shuffling processes continue until defined convergence criteria or iterations are satisfied. As a novel bio-evolutionary algorithm, the SFLA combines the benefits of the genetic-based algorithms and the social behavior-based algorithms which have the advantages of few parameters(less than PSO) and emphasis on both wide scan of a large solution space and deep search of promising locations for a global optimum [5].

In SFLA, each solution is a "frog", an initial population of *P* frogs is created randomly within the *D*-dimensional feasible space. The *i*-th frog's position is expressed as $X_i = (x_{i1}, x_{i2}, \cdots, x_{iD})$. After that, the frogs are sorted in descending order by their fitness. Then, the entire population is grouped into *m* memeplexes with the special strategy: the first frog goes to the first memeplex, the second frog goes to the second memeplex, *m*-th frog goes to the *m*-th memeplex, and $m+1$-th frog goes back to the first memeplex, etc. Each memeplex containing *n* frogs, which satisfy $P = m \times n$. The strategy of grouping is as follows:

$$M^k = \{y^k[j] \mid y^k[j] = X_{k+m(j-1)}, k = 1, \cdots, m, j = 1, \cdots, n\} \tag{3}$$

In each memeplex, record the frogs's position with the best and the worst fitness as *Xb* and *Xw* respectively. Also, record the frog's postion with the global best fitness as *Xg*. Then, the position of the frog with the worst fitness is adjusted as follows:

$$D_d = rand \times (Xb_d - Xw_d) \tag{4}$$

$$new\ Xw_d = old\ Xw_d + D_d \quad (D_{d\max} \geq D_d \geq -D_{d\max}) \tag{5}$$

Where *rand* is a random number in the range [0,1], $D_{i\max}$ is the maximum step size allowed for a frog in the *i*-th dimensional. If the evolution produces a better solution, it replaces the worst frog. Otherwise, the calculations in (4) and (5) are repeated but with *Xb* replaced by *Xg*. If no improvement becomes possible in this case, a new solution within the feasible space is randomly generated to replace the worst frog. The calculations then continue for a specific number of iterations [7].

3 The Proposed Grouping-Shuffling Particle Swarm Optimization

The PSO and SFLA algorithms have their own distinct characteristics. PSO has highly efficient and flexible individual flight learning strategy. SFLA can balance the grouping evolution and global information exchange. By Combining the advantages of two algorithms, embed the individual flight strategy of PSO into SFLA's grouping-shuffling evolution framework, we proposes a simple and efficient global search algorithm:

grouping-shuffling particle swarm optimization(GSPSO). In the algorithm, the whole particles periodically carry out grouping evolution and shuffling operation. In each group, the worst particle is continuously updated its flight velocity and position according to its own velocity inertia and the optimal positions of its located group and the swarm. The basic framework of the GSPSO is as the following fig.1.

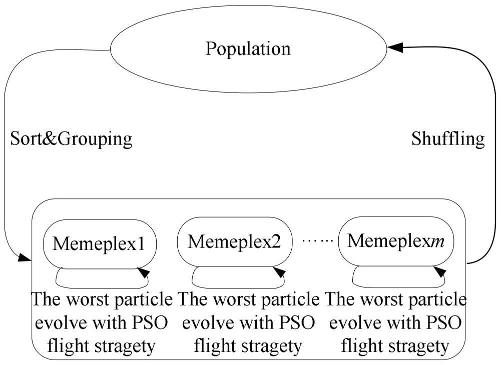

Population
Sort&Grouping
Shuffling
Memeplex1
Memeplex2
Memeplexm
The worst particle evolve with PSO flight stragety
The worst particle evolve with PSO flight stragety
The worst particle evolve with PSO flight stragety

Fig. 1. Framework of GSPSO

For the maximum optimization function *f*(*x*), the procedures of GSPSO can be described as follows:

Step1:Initialize algorithm parameters, including: population of particles *P*, memeplexes number *m*, number of evolution generation in each memeplex G_{meme}, max number of shuffling iterations G_{shuf};

Step2:Randomly initialize all particles' positions and velocities within the feasible space, and calculate their fitness;

Step3:Sort all particles in order of descending fitness, then divided them into *m* memeplexes according to(3), each containing *n* particles($n=P/m$);

Step4:Within each memeplex, the particles experience PSO's individual flight evolution strategy:

1) lg=1, set Xb^k and Xw^k as the first paricle $y^k[1]$ and last paricle $y^k[n]$ of *k*-th memplex respectively, also set *xg* as the particle with global best fitness;
2) Adjust the position and velocity of the worst particle Xw^k according to PSO's individual flight learning strategy as (6)(7), where use Xb^k to replace the individual history best position;

$$\text{new } v_{Xw^k d} = w \times v_{Xw^k d} + c_1 \times r_1 \times [Xb_d^k - Xw_d^k] + c_2 \times r_2 \times [Xg_d - Xw_d^k] \tag{6}$$

$$\text{new } Xw_d^k = Xw_d^k + \text{new } v_{xw^k d} \tag{7}$$

3) Calculate the fitness of new Xw^k and compare the relationship between new Xw^k and $y^k[n]$:

I If $f(\text{new } Xw^k) > f(y^k[n])$, let $y^k[n]$= new Xw^k and update the $y^k[n]$'s velocity as new V_{Xw^k} ;

II If $f(\text{new } Xw^k) < f(y^k[n])$, update $y^k[n]$'s position by implementing the crossover operation between Xb^k and Xw^k with a probability of 50%, and calculate the $y^k[n]$'s velocity according to it's new position. The crossover operation is as (8), where λ randomly choose as 0 or 1.

$$y_d^k[n] = \lambda * Xb_d^k + (1-\lambda) * Xw_d^k \tag{8}$$

4) After change for the worst particle, resort the k-th memeplex in order of decreasing fitness.

5) lg=lg+1, Repeat 1)-4) for a desired number of iterations lg= G_{meme}.

Step5:Re-shuffling the entire swarm after each memeplex complete evolution;

Step6:Repeat step3-step5 until satisfy the desired number of shuffling iterations G_{shuf}.

4 Numerical Experiments and Results

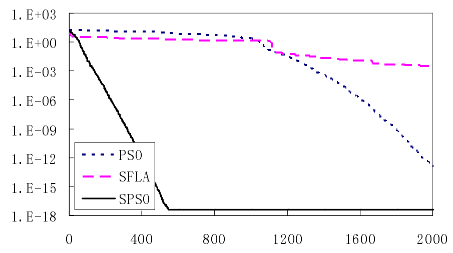
The optimization performance of algorithm, not only depends on its intrinsic characteristics, but also closely related to the complexity of the optimization problems. In this study, we use four well-known continuous benchmark functions (the Ackley function, Rosenbrock function, Griewangk function and Schaffer's f7 function) for numerical experiment to test the effect of proposed GSPSO algorithm and compared it with standard PSO and SFLA algorithms. The global optimum solution of all functions is known to be zero when all variables equal zero. These benchmark functions have different characteristics: Ackley and Rosenbrock functions are single-modal, Griewangk and Schaffer's f7 functions are multi-modal.

In order to obtain the best optimization performance for three algorithms, according to experiences and through repeat testing, PSO parameters are set as [10]: population P=100, acceleration coefficients c_1=c_2=2, inertia weight $w(g)$=0.9-(g/G)/2; SFLA Parameters are set as [5]: population P=100, number of memeplexes m=10, memetic evolution iterations G_{meme}=10; GSPSO parameters are set as: population P=100, number of memeplexes m=10, memetic evolution iterations G_{meme}=10, acceleration coefficients c_1=c_2=1, inertia weight w(g)=0.9-(g/G_{shuf})/2. For different variable dimension, the global iterations (or shuffling iterations) are set as 500 and 2000 respectively. In order to eliminate the influence of randomicity, 20 runs are performed for each function. The test results are shown in table1. Where Opt and Ave mean the optimal and average solution of 20 runs respectively, Var means the variance.

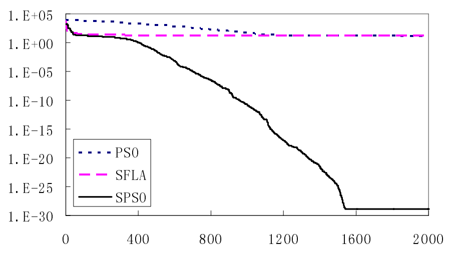
From table1 we can see that, with the finite global evolution iterations, PSO and SFLA are difficult to find the optimal solution area in solving the single-modal function Rosenbrock and the multi-modal function Schaffer's f7. However, GSPSO can easily find the optimal solution area and solve these two functions better. By comparing with PSO and SFLA in the aspects of optimal and average solution, results show that GSPSO has much better optimizing accuracy and stability then PSO and SFLA.

Table 1. Results of the benchmark functions tests

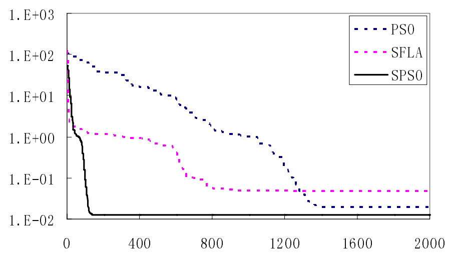
Function	Algorithm	20 variables,500 global iterations			50 variables, 2000 global iterations		
		Opt	Ave	Var	Opt	Ave	Var
Ackley	PSO	2.142E-04	7.587E-04	2.300E-07	5.599E-04	2.841E-03	5.800E-06
	SFLA	1.407E-01	1.487E+00	6.900E-01	9.060E-01	1.929E+00	3.964E-01
	GSPSO	0.000E-08	0.000E-08	0.000E-08	0.000E-08	0.000E-08	0.000E-08
Rosenbrock	PSO	5.855E+00	2.449E+01	4.413E+02	4.126E+01	7.854E+01	8.790E+02
	SFLA	1.781E+01	1.979E+01	1.669E+00	4.705E+01	4.936E+01	2.903E+00
	GSPSO	0.000E-08	4.000E-08	0.000E-08	0.000E-08	3.980E-01	1.026E+00
Griewangk	PSO	1.369E-04	2.914E-02	6.475E-04	9.460E-06	6.839E-03	1.679E-04
	SFLA	1.458E-01	6.972E-01	7.171E-02	1.395E-01	4.317E-01	3.699E-02
	GSPSO	0.000E-08	1.489E-02	1.516E-04	0.000E-08	3.529E-03	2.051E-05
Schaffer's f7	PSO	3.751E+00	8.461E+00	2.527E+01	4.407E+01	8.299E+01	8.671E+02
	SFLA	1.495E+00	6.834E+00	1.282E+01	5.964E+01	8.510E+01	1.012E+02
	GSPSO	0.000E-08	6.987E-02	7.046E-04	3.030E-02	6.994E-01	9.255E-01



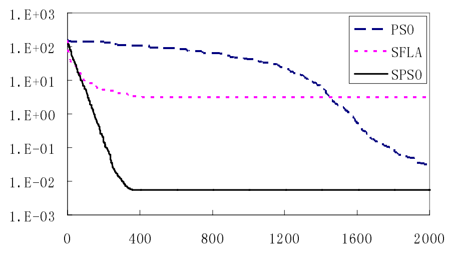
(a)



(b)



(c)



(d)

Fig. 2. Typical convergence curves of four functions: (a)Ackley, (b)Rosenbrock, (c) Griewangk, (d)Schaffer's f7

We further compare and analyze the convergence of these three algorithms. In the experiment, the parameters of three algorithms are as above. Fig 2 shows the typical convergence curves of these functions. The benchmark functions are all 20 dimensions. The figures illustrate that, the convergence speed of PSO is relatively slower but the convergence precision is slightly better than SFLA. The proposed GSPSO not only inherits the advantage of SFLA with fast convergence speed, but also has higher convergence precision than the PSO and SFLA. Therefore, according to the above two kinds of tests, it can be demonstrated that the GSPSO can easily escape the local optimal and avoid individual's premature convergence, and outperforms standard PSO and SFLA in solving all these test functions.

5 Conclusions

This paper proposes an improved PSO algorithm named GSPSO and uses it as a search engine to solve continuous optimization problem. By embedding the individual learning strategy of PSO into the grouping and shuffling evolution framework of SFLA, the GSPSO combines the advantages of two algorithms. Numerical experiments have shown that: in the one hand, GSPSO can avoid the PSO's shortage that easy to get rid of the local optimal solution; in the other hand, GSPSO have faster convergence speed and higher convergence precision than the PSO and SFLA. The results indicate that the GSPSO outperforms standard PSO and SFLA and is an efficient approach for solving the complex continuous optimization problems.

References

1. Shelokar, P.S., Siarry, P., Jayaraman, V.K., Kulkarni, B.D.: Particle swarm and ant colony algorithms hybridized for improved continuous optimization. Applied Mathematics and Computation 188(1), 129–142 (2007)
2. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: IEEE International Conference Neural Networks, pp. 1942–1948 (1995)
3. Ebehtart, R.C., Kennedy, J.: A new optimizer using Particle swarm theory. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Seienee, Nagoya, Japan, pp. 39–43 (1995)
4. Eusuff, M.M., Lansey, K.E.: Optimization of water distribution network design using the shuffled frog leaping algorithm. Journal of Water Resources Planning and Management 129(3), 210–225 (2003)
5. Elbeltagi, E., Hegazy, T., Grierson, D.: Comparison among five evolutionary-based optimization algorithms. Advanced Engineering Informatics 19(1), 43–53 (2005)
6. Alireza, R.V., Ali, H.M.: Solving a bi-criteria permutation flow-shop problem using shuffled frog-leaping algorithm. Soft Comput. 12, 435–452 (2008)
7. Elbehairy, H., Elbeltagi, E., Hegazy, T., Soudki, K.: Comparison of Two Evolutionary Algorithms for Optimization of Bridge Deck Repairs. Computer-Aided Civil and Infrastructure Engineering 21, 561–572 (2006)
8. Li, Y., Zhou, J., Yang, J., Liu, L., Qin, H., Yang, L.: The Chaos-based Shuffled Frog Leaping Algorithm and Its Application. In: Fourth International Conference on Natural Computation, vol. 1, pp. 481–485 (2008)

9. Eusuff, M.M., Lansey, K.E., Pasha, F.: Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization. Engineering Optimization 38(2), 129–154 (2006)
10. Shi, Y., Eberhart, R.C.: Parameter Selection in Particle Swarm Optimization. In: Porto, V.W., Waagen, D. (eds.) EP 1998. LNCS, vol. 1447, pp. 591–600. Springer, Heidelberg (1998)

Gender-Hierarchy Particle Swarm Optimizer Based on Punishment*

Jiaquan Gao[1], Hao Li[1], and Luoke Hu[2]

[1] Zhijiang College, Zhejiang University of Technology, Hangzhou 310024, China
[2] College of Mechanical Engineering, Zhejiang University of Technology, Hangzhou 310014, China
springf12@163.com

Abstract. The paper presents a novel particle swarm optimizer (PSO), called gender-hierarchy particle swarm optimizer based on punishment (GH-PSO). In the proposed algorithm, the social part and recognition part of PSO both are modified in order to accelerate the convergence and improve the accuracy of the optimal solution. Especially, a novel recognition approach, called general recognition, is presented to furthermore improve the performance of PSO. Experimental results show that the proposed algorithm shows better behaviors as compared to the standard PSO, tribes-based PSO and GH-PSO with tribes.

Keywords: gender, hierarchy, recognition, particle swarm optimizer.

1 Introduction

In 1995, a new optimizer, particle swarm optimizer (PSO), was introduced by Kennedy and Eberhart, which was inspired by bird flocking, fish schooling, bee swarm, etc. [1]. As one of excellent paradigms of swarm intelligence (SI), PSO shows its excellent ability in continuous none-linear function optimization, and has more less parameters involved as compared to other evolutionary algorithms (EAs). Hence, it has been used in many practical applications, such as pattern recognition, image match, medical image process, and so on. Especially, when applied to function optimization, most of PSO was designed to accelerate the convergence and improve the accuracy of optima, such as LBEST PSO, GBEST PSO [1], linear decreasing weight PSO [2], constricted PSO [3], tribe PSO [4], [5], et al. The GBSET PSO performed better in terms of convergence, but it was at cost of loss of diversity comparing with LBEST PSO. Therefore, the GBEST PSO has less possibility to obtain optima in problem space than LBEST PSO. For improving the convergence and the accuracy of the optimal solution, recently, it is found that the hybrid versions of PSO, such as tribe PSO etc., shows good performance.

In this paper, a new hybrid version of PSO was proposed. In the proposed algorithm, the particle has two attributes, gender and hierarchy. With the two

* The research is funded by National Natural Science Foundation of China (10926198).

attributes, the particles show the ability of the exploitation and exploration in search space, respectively. In addition, besides modifying the social part and recognition part of PSO, a novel recognition approach, called general recognition, is presented to accelerate the convergence and improve the accuracy of the optimal solution. Our proposed algorithm is tested on five benchmark problems, and compared with the standard PSO, tribes-based PSO and GH-PSO with tribes.

2 Gender-Hierarchy Particle Swarm Optimizer Based on Punishment

2.1 A Standard GBEST Model of Particle Swarm Optimizer

The GBEST PSO was proposed 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 $\phi_1 = \gamma_1 * a_g, \phi_2 = \gamma_2 * a_t, \gamma_1, \gamma_2 \rightarrow U(0,1) \in R$ and $V = [\nu_1, \nu_2, \ldots, \nu_n], X = [x_1, x_2, \ldots, x_n]$. X and V denote a vector of a particle's position and velocity on n dimensions, respectively. v_i^k and x_i^k respectively stand for the velocity and position of a particle on the ith dimension in the kth iteration step.ω is an inertia weight introduced firstly by Shi and Eberhart [6]. g^k represents the global best position among all swarm particles until the kth iteration step. The best position of the ith particle before the kth iteration is denoted by l_i^k. γ_1 and γ_2 are two random numbers with a uniform distribution at the interval [0,1]. $a_g$and a_t are two positive constant values. Usually, they both are set to 2.

For GBEST PSO, it can be obviously seen that it is consisted 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. However, it is a self-study procedure in the recognition part. That is, each particle learns from its previous experienced best particle, and furthermore is moved to a fittest position.

2.2 The Proposed Algorithm

Before presenting our algorithm, some concepts or terms are firstly introduced.

- Gender: Gender usually is used to describe the sexual character, which is easily found in human beings and mammal animals. Generally, male is advantageous over female in exploring the unknown world. Hence, inspired by this natural character, the gender concept is introduced to particle agents. In the initial phase, each particle is randomly assigned a sexual character. If a particle is male, it will be helpful to facilitate the exploration of the search space. On the contrary, the female particle can improve the exploitation of the search space.

– Hierarchy: In human being and other sophisticated societies, they all have a hierarchy. if an individual is in relative high hierarchical level, he will has more powerful in society. Inspired by this concept, the hierarchy is also introduced to particle agents. Each particle agent has a hierarchical level, also called grade attribute, which is formulated as follows:

$$\omega_{grade\_i\_k} \triangleq \{fitness(x_{i\_k}) - fitness(g_{i\_k})\}/fitness(g_{i\_k}) \tag{3}$$

where $\omega_{grade\_i\_k}$ represents the ith particle's grade value in kth iteration. The fitness value of objective functions is denoted by $fitness$. g_{i_k}describes the global best position among all swarm particles before the kth iteration. x_{i_k} denotes the position of the ith particle in the kth iteration. From the definition of grade mentioned in (3), we can observe that for a particle, the smaller its value of grade is, the better its position is.

Next, based on the above definitions and concepts, our proposed algorithm, GH-PSO, is listed as follows:

$$v_i^{k+1} = \omega * v_i^k + \phi_1 * (g^k - \omega_{particle\_i} * x_i^k) + \phi_2 * (l_i^k - \omega_{particle\_i} * x_i^k) + \omega_{si} * (X_{average\_i} - x_i^k) \tag{4}$$

$$x_i^{k+1} = x_i^k + \omega * v_i^{k+1} \tag{5}$$

with $\phi_1 = \gamma_1 * a_g, \phi_2 = \gamma_2 * a_t$, $\gamma_1, \gamma_2 \rightarrow U(0,1) \in R$ and $V = [\nu_1, \nu_2, \ldots, \nu_n], X = [x_1, x_2, \ldots, x_n]$. Where $\omega, \phi_1, \phi_2, g^k, x_i^k$, and l_i^k have the same definitions as in Sect. 2.1. The parameters $\omega_{si}, X_{average\_i}, \omega_{particle\_i}$ are defined as $\omega_{si} \triangleq (\sum_{i=1}^{i=N} fitness(x_i))/(\sum_{i=1}^{i=N} fitness(x_{optima}))$, $X_{average\_i} \triangleq \sum_{j=i}^{j=N} x_{ij}/N$ and $\omega_{particle\_i} \triangleq (c_1 * \omega_{grade\_i\_k} + c_2 * \omega_{gender\_i\_k})/(c_1 + c_2)$, where x_i is the position of the ith particle, the number of particles is denoted as N. x_{optima} represents the best position of all particles. $fitness(x_i)$ and $fitness(x_{optima})$ is the fitness value of the ith particle and the optimal solution, respectively. The mean value of position of all particles on the ith dimension is described as $X_{average\_i}$. $\omega_{particle\_i}$ is the weight of the ith particle, which is constructed with two parts: gender's weight and grade's weight. 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. ω_{si} is a recognition weight.

In our proposed algorithm, it can be observed that the social part and the recognition part both are modified in order to accelerate the convergence by multiplying a new coefficient $\omega_{particle}$*i*. From $(g^k - x_i^k)$, $(l_i^k - x_i^k)$ and (1), we can observe that the velocity of a particle is not only directly proportional to distance between current position and the best position, but is also directly proportional to distance between its current position and its previous best position. Therefore, by amplifying the distances, the particles will fly faster towards the best position so that the convergence of the proposed algorithm is accelerated. The third term of (4) is called "general recognition" because each particle can adjust itself to behaviors of most particles based on it, and thus improve the diversity of the population. After the definitions and concepts mentioned above are given, the pseudo-code of GH-PSO algorithm is listed as follows:

```
 1. Initialize the agents of particles with gender randomly
 2. While not finished
 3.    For each particle in swarm
 4.       Calculate the fitness value.
 5.       Update the optimal value, best particle $g^k$ and local best particle $l_i^k$.
 6.    End of For loop
 7.    Calculate the grade of each particle, $\omega_{particle\_i}$.
 8.    Calculate the weight of swarm, $\omega_{si}$.
 9.    Calculate $X_{average}$.
10.    Update the velocity and position of particle , $x_i^k$ and $v_i^k$.
11.    Compute the punishment, $V_{max}$.
12.    If the terminal conditions are satisfied, then terminate the procedure, and
    output the optimal solution.
13. End of While.
```

2.3 Parameter Selection of GH-PSO

In [3,7], the analysis on trajectory of simplified PSO was described, and some guidelines of parameter selection have also been given. It have proved that the trajectory of a particle agent does not "fly" in its search space but "surfs" it on sine waves. At early stage of PSO, Shi and Eberhart have drawn a conclusion that it will be better if the inertia weight is within the range of [0.9,1.2] by some numerical experiments [2]. Generally, the PSO algorithms will take the least iterations to obtain the optima with a inertia weight in this range. In addition, the two parameters, a_g and a_t, are usually set to 2. In order to avoid the problem that the particles fly out of the search space, the upper limitation of positions and velocities of particles is respectively set, and denote them by X_{max} and V_{max}, respectively. For each particle, its position must be within the interval $[-X_{max}, X_{max}]$, and its velocity cannot also be out of the interval $[-V_{max}, V_{max}]$. Also, it is observed that it is a good choice if X_{max} is equal to V_{max}. In our experiments, we will use the same scheme of parameter selection as in the above literature. Especially, the parameter $\omega_{particle\_i}$ is confined within the interval [1.09, 2.89].

2.4 Punishment Policy in GH-PSO

For PSO, the particles are often out of control and fly out of the search space, which will result in a bad convergence. Therefore, in order to avoid the difficulty, an approach is suggested with setting the upper limitation of velocities of particles [8,9]. Furthermore, in the literature [10], Braendler and Hendtlass present a different approach. the velocity of particles is adjusted using hyperbolic tangent function rather than their upper limitation as compared to the previous method in runs. In GH-PSO, besides adopting Braendler and Hendtlass's method, a new punishment policy is proposed to better control the particles' behaviors. The policy is described as follows:

Step 1: Calculate the velocity of particles on each dimension. For each particle, it will be punished if its velocity is out of the interval $[-V_{\max}, V_{\max}]$. A stochastic number at the interval [0,1] is generated. Then, the velocity is multiplied by the stochastic number as a punishment.
Step 2: Calculate the new velocity of particles. For each particle, If its new velocity is still out of the interval $[-V_{\max}, V_{\max}]$, we set the velocity of particle to V_{max} (-V_{max}) if its new velocity is more than V_{max} (less than $-V_{max}$).
Step 3: End the punishment procedure.

In fact, this punishment policy is similar to the selection operator and mutation operator, which are used in genetic algorithms (GAs). Here, the test operation described in Step 1 is identical to GAs' selection operation, and the punishment operation described in Step 2 is similar to GAs' mutation operation. The two operations will be helpful to facilitate the diversity of the population.

3 Experiments and Discussions

3.1 Experimental Settings

For showing the performance of GH-PSO, GH-PSO is compared with the standard PSO [1,6], tribes-based POS [5]. Especially, GH-PSO is also compared with GH-PSO with tribes in order to investigate the effect of the tribes. The test problems are selected from the benchmark functions, which include Sphere function, Rosenbrock function, Rastrigin function, Griewank function and Schaffer's f6 function [7].

The boundaries of velocity of particles, V_{max}, are set to 100, 30, 5.12, 600, 100 corresponding to each function mentioned above, respectively. For each function, all algorithms will be conducted on the following three different dimension sizes: 30, 20 and 10. For the three dimension sizes, the population of particles is respectively set to 160 ,100, 60 and the maximum number of generations is set to 2000, 1500, 100 for each algorithm. The inertia weight ω is decreased linearly from 0.9 to 0.4 with the increasing generation number, and its cut-off is set to 1×10^{-8}.

All algorithms are implemented under FreeBSD 7.0 system platform with GCC 4.2.1 compiler in C++.

3.2 Experimental Results

All algorithms are randomly run 50 times, and the change curve of the mean value of the optimal solution is shown in Figs. 1- 3 with the increasing generation number for each algorithm, respectively. The mean value of the optimal solution obtained by all algorithms on three different dimension sizes is respectively shown in Tables 1-3.

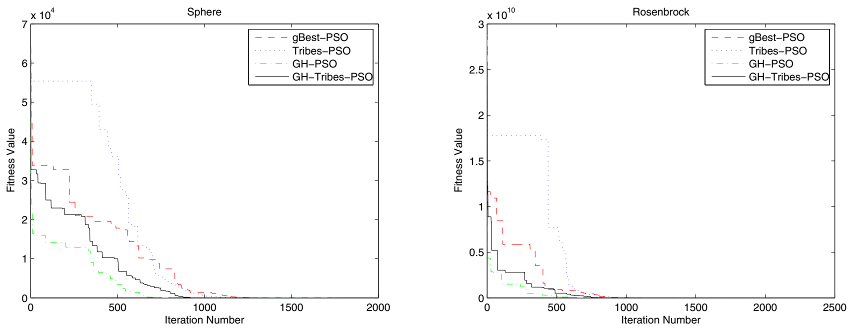


Fig. 1. Convergence diagram of Sphere and Rosenbrock

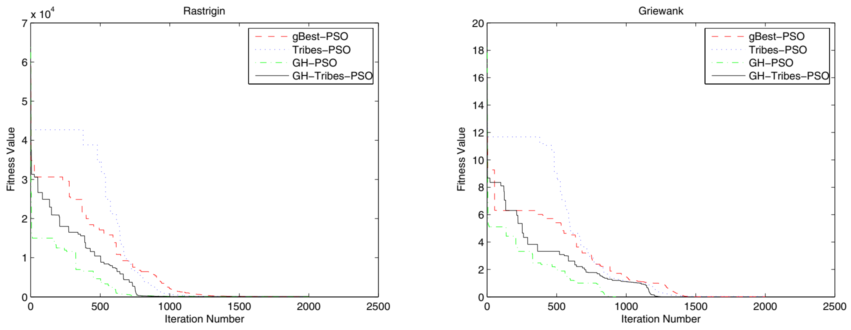


Fig. 2. Convergence diagram of Rastrigin and Griewank

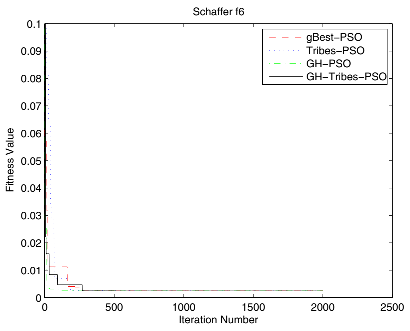


Fig. 3. Convergence diagram of Schaffer's f6

From Figs. 1 to 3, we can observe that GH-PSO has a better convergence than all other algorithms. In fact by analyzing the values in Tables 1-3, our observation is affirmed. GH-PSO has a smaller optimal value than all other algorithms. Therefore, a conclusion can be drawn that our algorithm GH-PSO outperforms all other algorithms for each test problem.

Table 1. The mean value of optima on 30 dimensions

Function	gBest-PSO	Tribes-PSO	GH-PSO	GH-Tribes PSO
Sphere (f_0)	9.250000×10^{-8}	1.131000×10^{-7}	8.650000×10^{-8}	8.420000×10^{-8}
Rosenbrock (f_1)	1.211339×10^{2}	1.462742×10^{2}	2.748598×10^{1}	2.795591×10^{1}
Rastrigin (f_2)	3.200599×10^{1}	4.565054×10^{1}	1.335908×10^{1}	1.373763×10^{1}
Griewank (f_3)	1.210836×10^{-2}	2.680860×10^{-2}	1.012459×10^{-2}	1.180661×10^{-2}
Schaffer f6 (f_6)	2.455858×10^{-3}	2.455858×10^{-3}	2.455858×10^{-3}	2.455858×10^{-3}

Table 2. The mean value of optima on 20 dimensions

Function	gBest-PSO	Tribes-PSO	GH-PSO	GH-Tribes PSO
Sphere (f_0)	8.730000×10^{-8}	8.730000×10^{-8}	7.840000×10^{-8}	8.220000×10^{-8}
Rosenbrock (f_1)	5.706853×10^{1}	6.654097×10^{1}	1.773444×10^{1}	1.800886×10^{1}
Rastrigin (f_2)	1.767705×10^{1}	2.228704×10^{1}	1.335908×10^{1}	1.373763×10^{1}
Griewank (f_3)	1.210836×10^{-2}	2.680860×10^{-2}	1.012459×10^{-2}	1.180661×10^{-2}
Schaffer f6 (f_6)	2.455858×10^{-3}	2.455858×10^{-3}	2.455858×10^{-3}	2.455858×10^{-3}

Table 3. The mean value of optima on 10 dimensions

Function	gBest-PSO	Tribes-PSO	GH-PSO	GH-Tribes PSO
Sphere (f_0)	8.090000×10^{-8}	8.630000×10^{-8}	7.840000×10^{-8}	7.960000×10^{-8}
Rosenbrock (f_1)	3.687350×10^{1}	3.031101×10^{1}	0.602891×10^{1}	0.782953×10^{1}
Rastrigin (f_2)	0.292719×10^{1}	1.134251×10^{1}	0.936522×10^{1}	1.096791×10^{1}
Griewank (f_3)	1.010836×10^{-2}	1.680860×10^{-2}	0.812459×10^{-2}	1.000661×10^{-2}
Schaffer f6 (f_6)	2.455858×10^{-3}	2.455858×10^{-3}	2.455858×10^{-3}	2.455858×10^{-3}

3.3 Discussions

For GH-PSO, two concepts, gender and hierarchy, are introduced to particle agents. At the initial stage of runs, each particle is accelerated by its inertia weight, $\omega_{particle\_i}$, and thus the agents fly faster towards the better positions than those without the inertia weight. When the particle agents are closer to the optimal solution, they slow down the search and fine-tune to the optima by applying the lower particle's weight. Through employing the general recognition, the particle agents can adjust its position by learning from the popular behaviors.

From (4), we know that if a particle's weight, $\omega_{particle\_i}$, and the general recognition weight, ω_{si}, are set to 1 and 0, respectively, then GH-PSO becomes GBEST PSO. And, from the definition of ω_{si}, the value of general recognition weight is inclined to 1. Therefore, this tendency is used as an indication to identify whether the particle agents are close to optimal solution or not.

The experimental results verify that the performance of GH-PSO is improved by employing gender and hierarchy (see Figs. 1- 3 and Tables 1-3).

4 Conclusions

In this paper, we propose a novel algorithm, called GH-PSO. By revising the original version of PSO with particles' inertia weights and general recognition, the performance of PSO is improved. The experimental results show that GH-PSO is an effective approach as compared to the standard PSO, tribes-based PSO and GH-PSO with tribes.

The further works will focus on the multiobjective function optimization, and the experiments are already underway.

References

1. Kennedy, J., Eberhart, R.C.: A new optimizer using particle swarm theory. In: Proc. 6th International Symposium on Micro Machine and Human Science, Nagoya, Japan, pp. 39–43 (1995)
2. Shi, Y.H., Eberhart, R.C.: Empirical study of particle swarm optimization. In: Proc. 1999 Congress on Evolutionary Computation, pp. 1945–1950. IEEE Press, Piscataway (1999)
3. Clerc, M., Kennedy, J.: The particle swarmexplosion, stability, and convergence in a multidimensional complex space. IEEE Trans. Evol. Comput. 6, 58–73 (2002)
4. Chen, K., Li, T.H., Cao, T.C.: Tribe-PSO: A novel global optimization algorithm and its application in molecular docking. Chemometrics Intell. Lab. Syst. 82, 248–259 (2006)
5. Cooren, Y., Clerc, M., Siarry, P.: Performance evaluation of TRIBES, an adaptive particle swarm optimization algorithm. Swarm Intelligence 3, 1935–3820 (2009)
6. Shi, Y.H., Eberhart, R.C.: A modified particle swarm optimizer. In: Proc.1998 IEEE International Conference on Computational Intelligence, Anchorage, Alaska, pp. 69–73. IEEE Press, Los Alamitos (1998)
7. Trelea, I.C.: The particle swarm optimization algrorithm:convergence analysis and parameter selection. Inform. Proc. Lett. 85, 317–325 (2003)
8. Fan, H.Y.: A modification to particle swarm optimization algorithm. Eng. Comput. 19, 970–989 (2002)
9. Schutte, J.F., Groenwold, A.A.: A study of global optimization using particle swarms. Struct. Multidis. Optim. 25, 261–269 (2003)
10. Braendler, D., Hendtlass, T.: Improving particle swarm optimization using the collective movement of the swarm. IEEE Trans. Evol. Comput. (to appear)

An Improved Probability Particle Swarm Optimization Algorithm

Qiang Lu[1] and Xuena Qiu[2]

[1] School of Automation, Hangzhou Dianzi University,
Hangzhou, 310018, China
lvqiang@hdu.edu.cn

[2] School of Telecommunication, Ningbo University of Technology,
Ningbo, 310018, China
qiuxn26@hotmail.com

Abstract. This paper deals with the problem of unconstrained optimization. An improved probability particle swarm optimization algorithm is proposed. Firstly, two normal distributions are used to describe the distributions of particle positions, respectively. One is the normal distribution with the global best position as mean value and the difference between the current fitness and the global best fitness as standard deviation while another is the distribution with the previous best position as mean value and the difference between the current fitness and the previous best fitness as standard deviation. Secondly, a disturbance on the mean values is introduced into the proposed algorithm. Thirdly, the final position of particles is determined by employing a linear weighted method to cope with the sampled information from the two normal distributions. Finally, benchmark functions are used to illustrate the effectiveness of the proposed algorithm.

Keywords: Normal distribution, probability particle swarm optimization, evolutionary computation.

1 Introduction

The particle swarm optimization (PSO) algorithm, originally proposed by Kennedy and Eberhart (1995) [1,2], has become one of the fascinating branches of evolutionary computation. The underlying motivation for the development of the PSO algorithm is the social behavior of animals such as bird flocking and fish schooling. In the last decade, the PSO algorithm has been well studied [3,4,5]. For example, in [3], a quantum-behaved particle swarm optimization algorithm was proposed for a class of unconstrained functions by introducing several simple quantum concepts. In [4,5], the authors designed a class of hybrid optimization approaches by combining the PSO algorithm with other methods such as an ant colony algorithm in order to improve the efficiency of the PSO algorithm. These improved PSO algorithms [3,4,5] have a common feature, i.e., the particle's velocity part is considered. However, in 2003, Kennedy [6] (2003) proposed a bare bones particle swarm algorithm where the particle's velocity part is dropped and the position of any particle on each dimension satisfies a normal distribution with mean $\frac{p_{id}+p_{gd}}{2}$ and standard deviation $|p_{id} - p_{gd}|$ where p_{gd} and p_{id} are the global best position and the previous best position, respectively. From the bare bones particle swarm algorithm, one can see that this algorithm provides another way, i.e., probability method to improve the search performance of the PSO algorithm. Recently, the study on probability particle swarm algorithm has received increasing interest from scholars [7,8,9,10,11]. For instance, in [7,8], a particle swarm optimization algorithm based on pheromone mechanism was proposed for unconstrained functions by introducing an information-shared matrix, which is used to store useful information. In [9,10], based on Gaussian or Cauchy method, the bare bones particle swarm algorithm was further improved. Moreover, in order to correct the weakness of the PSO algorithm such as linearization of the curve attained in steady-state, Secrest [11] (2003) presented a Gaussian particle swarm optimization algorithm. However, only a few results on the study of the probability particle swarm algorithm are reported. Therefore, how to further improve the search performance of the probability particle swarm algorithm for the problem of unconstrained optimization is the motivation of the current study.

In this paper, in order to deal with the problem of unconstrained optimization, we will propose an improved probability particle swarm optimization (IPPSO) algorithm. The proposed algorithm uses two normal distributions to update the position of each particle. The means of two normal distributions are the global best position and the previous best positions, respectively. Moreover, we will introduce a disturbance to the means of two normal distributions. And the information about fitness will be used as the standard deviation of the two normal distributions. We will use twelve benchmark unconstrained functions to illustrate the effectiveness of the proposed algorithm.

2 An Improved Probability Particle Swarm Optimization Algorithm

2.1 Algorithm Details

By sampling two normal distributions $N((randn()\eta + 1)p_{gd}(k), u_1(k+1))$ and $N((randn()\eta+1)p_{id}(k), u_2(k+1))$, two temporal positions $\xi^g_{id}(k+1)$ and $\xi^i_{id}(k+1)$ are obtained by using (1) and (2), respectively.

$$\xi^g_{id}(k+1) = \frac{randn()}{u_1(k+1)} + (randn()\eta + 1)p_{gd}(k) \tag{1}$$

$$\xi^i_{id}(k+1) = \frac{randn()}{u_2(k+1)} + (randn()\eta + 1)p_{id}(k) \tag{2}$$

where $randn()$ is a random number satisfying a normal distribution with mean 0 and variance 1; η is a heuristic parameter; p_{id} denotes the previous best position of the i^{th} particle; p_{gd} is the global best position among particles; $u_1(k+1)$ and $u_2(k+1)$ are adjustment parameters, which can be calculated by

$$u_1(k+1) = u_1(k) + f(x_i(k)) - f(p_g(k)) \tag{3}$$

$$u_2(k+1) = u_2(k) + f(x_i(k)) - f(p_j(k)) \tag{4}$$

where $x_i(k)$ denotes the position of the i^{th} particle; $f : R^n \rightarrow R$ denotes the fitness function; $u_1(k)$ and $u_2(k)$ can be evaporated as

$$u_1(k) = evp \cdot u_1(k) \tag{5}$$

$$u_2(k) = evp \cdot u_2(k) \tag{6}$$

A threshold l_{th} is defined. If one random number in [0,1] is less than the threshold l_{th}, the position of the particle will be updated based on (8). Otherwise, it will be adjusted according to (7).

$$x_{id}(k+1) = c_1\xi_{id}^{g}(k+1) + c_2\xi_{id}^{i}(k+1) \tag{7}$$

$$x_{id}(k+1) = rand() \tag{8}$$

where $rand()$ denotes a random value in the search range; c_1 and c_2 are adjustment parameters and satisfy $c_1 + c_2 = 1$; $x_{id}(k+1)$ is a new position of the i^{th} particle.

2.2 Algorithm Procedure

The procedure of the IPPSO algorithm is summarized as

Algorithm 1 ***(IPPSO Algorithm)***

```
1. Initialization.
   (a) Initialize the threshold $l_{th}$, adjustment parameters $c_1$ and $c_2$, and
       heuristic parameter $\eta$.
   (b) Initialize the adjustment parameters $u_1$ and $u_2$ with 0.
2. Repeat until a given maximal number of iteration is achieved.
   (a) Evaluate the fitness of each particle.
   (b) Store the global best particle and fitness.
   (c) Store the previous best particle of the $i^{th}$ ($i = 1, 2, \cdots n$) particle and its
       fitness.
   (d) Perform (5) and (6).
   (e) Each particle performs the following steps.
       i. Perform (3) and (4).
       ii. Perform (1) and (2).
       iii. If a random number is lesser than the threshold $l_{th}$, perform (8) and
            (7) otherwise.
```

Table 1. Parameters of the IPPSO algorithm

Population	l_{th}	c_1	c_2	$(\eta, evp) d < 10$	$(\eta, evp) 10 \leq d < 600$	Max iteration
20	0.01	0.8	0.2	(0,0.01)	(0.5,0.9)	2000

3 Simulated Experiments

In order to evaluate the performance of the IPPSO algorithm, the experiments will be conducted from two aspects: the fixed number of function evaluations and the fixed convergence precision of functions. Several algorithms published in recent years such as BBPSO [6], BBPSO-GJ1 [9], BBPSO-CJ2 [10] and GPSO [11] are used as comparison examples. The parameters of the IPPSO algorithm are listed in Table 1. Moreover, twelve benchmark unconstrained functions can be seen in [5].

3.1 Fixed Number of Function Evaluations

The objective of this subsection is to evaluate the performance of the IPPSO algorithm based on the fixed number of function evaluations, which is set as 40000. The average objective function values of the BBPSO, GPSO, BBPSO-GJ1 and BBPSO-GJ2 algorithms over 50 runs on each benchmark function $(f_1 - f_{12})$ are shown in Table 2 and 3.

From Table 2 and 3, one can see that the better average values on functions f_1, f_2, f_3, f_4, f_8 and f_9 are obtained by using the IPPSO algorithm in contrast to other algorithms. Moreover, for functions f_5 and f_6, the BBPSO algorithm can obtain better function values while the BBPSO-GJ2 algorithm can gain the better function values for functions f_7, f_{10}, f_{11} and f_{12}.

3.2 Fixed Convergence Precision of Functions

In this subsection, success rate, the number of function evaluations and standard deviation are used as indexes to evaluate the performance of the IPPSO algorithm under the fixed convergence precision of functions. The results of these algorithms on functions $f_1 - f_{12}$ (each algorithm runs 50 times on each function) are reported in Table 4 and 5.

From Table 4 and 5, one can see that on functions $f_1 - f_8$ and f_{12}, 100% success rate is gained by the IPPSO algorithm while on functions $f_9 - f_{10}$ 100% success rate is only obtained by the BBPSO-GJ2 algorithm. For functions f_8 and f_{11}, function values obtained by the BBPSO, BBPSO-GJ1 and GPSO algorithms cannot reach the given convergence precision within the max iteration. For the number of function evaluations, compared with other algorithms, the IPPSO algorithm also gains the less number of function evaluations on all functions except for f_3, f_{10}, f_{11} and f_{12}.

Table 2. Results of the IPPSO, GPSO and BBPSO algorithms based on the fixed number of function evaluations

Function	Average objective function value (Std.)		
	$IPPSO$	$GPSO$	$BBPSO$
f_1	3.0000(0)	4.08(5.3446)	3.0000(2.57E-15)
f_2	0.3979(0)	0.3979(4.59E-16)	0.3979(3.36E-16)
f_3	-2.0000(0)	-1.9128(6.94E-2)	-1.9976(1.71E-2)
f_4	-186.7309(0)	-186.7309(1.02E-13)	-186.7309(1.785E-13)
f_5	-3.8627(3.94E-5)	-3.8628(2.57E-15)	-3.8628(4.93E-15)
f_6	-3.2798(5.60E-2)	-3.2651(6.02E-2)	-3.2842(5.62E-2)
f_7	8.44E-11(3.37E-11)	2.4632E+4(6.87E+3)	3.69E-24(7.31E-24)
f_8	2.22E-12(5.67E-13)	3.4661(2.1744)	174.805(147.1863)
f_9	8.824(0.99)	40.0857(92.2618)	100.557(318.9685)
f_{10}	3.4E-3(6.8E-3)	9.9E-3(9.8E-3)	1.22E-2(1.64E-2)
f_{11}	1.67(6.22)	124.1306(27.52)	73.6069(23.6241)
f_{12}	5.41E-7(4.33E-7)	2.71E-2(2.14E-2)	1.4E-2(2.06E-2)

Table 3. Results of the IPPSO, BBPSO-GJ1 and BBPSO-GJ2 algorithms based on the fixed number of function evaluations

Function	Average objective function value (Std.)		
	$IPPSO$	$BBPSO - GJ2$	$BBPSO - GJ1$
f_1	3.0000(0)	3.0000(2.68E-15)	3.0136(5.12E-2)
f_2	0.3979(0)	1.2059(1.0143)	0.3979(6.08E-5)
f_3	-2.0000(0)	-2.0000(0)	-1.9882(2.23E-2)
f_4	-186.7309(0)	-186.5531(2.08E-1)	-186.6962(1.538E-1)
f_5	-3.8627(3.94E-5)	-3.8595(3.2E-3)	-3.8413(2.92E-2)
f_6	-3.2798(5.60E-2)	-3.0103(7.18E-2)	-2.9864(1.55E-1)
f_7	8.44E-11(3.37E-11)	0(0)	6.52E+4(5.49E+4)
f_8	2.22E-12(5.67E-13)	2.2614E-6(1.01E-5)	1.25E+4(4.78E+3)
f_9	8.824(0.99)	9.5279(0.6023)	1.44E+5(4.14E+5)
f_{10}	3.4E-3(6.8E-3)	0(6.34E-3)	2.63E-2(3E-2)
f_{11}	1.67(6.22)	1.1689(4.006)	27.343(8.288)
f_{12}	5.41E-7(4.33E-7)	0(0)	5.2E-3(1.6E-2)

Table 4. Results of the IPPSO, GPSO and BBPSO algorithms based on the fixed convergence precision of functions

Function	Required accuracy	Success rate			Function evaluations(Std.)		
		IPPSO	*GPSO*	*BBPSO*	*IPPSO*	*GPSO*	*BBPSO*
f_1	1E-3	100	100	100	329 (136.91)	304 (87.27)	329 (71.85)
f_2	1E-3	100	100	100	152 (76.45)	287 (133.75)	275 (126.07)
f_3	1E-3	100	46	94	1028 (632.75)	21842 (1.98E+4)	3026 (9.45E+3)
f_4	1E-2	100	100	98	373 (148.87)	668 (262.14)	2368 (6.45E+3)
f_5	1E-4	100	100	100	538 (342.27)	1736 (1352)	356.8 (86.125)
f_6	1E-3	100	44	68	402 (79.61)	22709 (1.97E+4)	1317 (1.85E+4)
f_7	1E-5	100	0	100	1810 (101.67)	40000 (0)	14172 (897.80)
f_8	1E-5	100	0	0	9077 (3.11E+3)	40000 (0)	40000 (0)
f_9	10	98	0	62	1614 (5.54E+3)	40000 (0)	19488 (1.77E+4))
f_{10}	1E-2	84	56	58	11043 (1.56E+4)	26188 (1.25E+4)	22424 (1.51E+4)
f_{11}	1E-2	90	0	0	14362 (1.25E+4)	40000 (0)	40000 (0)
f_{12}	1E-5	100	30	74	5156 (6.32E+3)	30137 (1.57E+4)	13479 (1.7E+4)

Table 5. Results of the IPPSO, BBPSO-GJ1 and BBPSO-GJ2 algorithms based on the fixed convergence precision of functions

Function	Required	Success rate			Function evaluations		
	accuracy	IPPSO	BBPSO -GJ1	BBPSO -GJ2	IPPSO	BBPSO -GJ1	BBPSO -GJ2
f_1	1E-3	100	82	100	329 (136.91)	8845 (1.57E+4)	698 (349.44)
f_2	1E-3	100	100	96	152 (76.45)	457 (708.70)	4795 (9.12E+3)
f_3	1E-3	100	36	100	1028 (632.75)	25980 (1.89E+4)	628 (182.23)
f_4	1E-2	100	82	18	373 (148.87)	8936 (1.55E+4)	37066 (7.86E+3)
f_5	1E-4	100	10	0	538 (342.27)	36027 (1.2E+4)	40000 (0)
f_6	1E-3	100	0	0	402 (79.61)	40000 (0)	40000 (0)
f_7	1E-5	100	0	100	1810 (101.67)	40000 (0)	11498 (604.88)
f_8	1E-5	100	0	86	9077 (3.11E+3)	40000 (0)	32108 (5.98E+3)
f_9	10	98	0	100	1614 (5.54E+3)	40000 (0)	4523 (528.48)
f_{10}	1E-2	84	0	100	11043 (1.56E+4)	40000 (0)	9103 (804.56)
f_{11}	1E-2	90	0	100	14362 (1.25E+4)	40000 (0)	9041 (925.17)
f_{12}	1E-5	100	10	100	5156 (6.32E+3)	36495 (1.07E+4)	1554 (412.36)

4 Conclusion

An improved probability particle swarm optimization (IPPSO) algorithm has been proposed. We have also illustrated the effectiveness of the proposed algorithm through twelve benchmark functions from two aspects: the fixed number of function evaluations and the fixed convergence precision of functions. It is worth mentioning that BBPSO-GJ2 can also generate much better results, especially for functions with many local maxima.

Acknowledgments. This work was supported in part by the Natural Science Foundation of Zhejiang under Grant Y1090426, and by the Technical Project of Zhejiang under Grant 2009C33045.

References

1. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: IEEE International Conference on Neural Network, pp. 1942–1948. IEEE Press, New York (1995)
2. Eberhart, R.C., Kennedy, J.: A New Optimizer Using Particle Swarm Theory. In: 6th International Symposium on Micro Machine and Human Science, pp. 39–43. IEEE Press, New York (1995)
3. Lu, Q., Chen, R.-Q., Yu, J.-S.: Quantum Continuous Particle Swarm Optimization Algorithm and Its Application. System Engineering-Theory and Practice 28, 122–130 (2008)
4. Liu, B., Wang, L., Jin, Y.-H.: Improved Particle Swarm Optimization Combined with Chaos. Chaos, Solitons and Fractals 25, 1261–1271 (2005)
5. Shelokar, P.S., Siarry, P., Jayaraman, V.K.: Particle Swarm and Ant Colony Algorithms Hybridized for Improved Continuous Optimization. Applied Mathematics and Computation 188, 129–142 (2005)
6. Kennedy, J.: Bare Bones Particle Swarms. In: Swarm Intelligence Symposium, pp. 80–87. IEEE Press, New York (2003)
7. Lu, Q., Qiu, X., Liu, S.: A Discrete Particle Swarm Optimization Algorithm with Fully Communicated Information. In: ACM/SIGEVO Summit on Genetic and Evolutionary Computation, pp. 393–400. ACM Press, New York (2009)
8. Lu, Q., Liu, S., Qiu, X.: Design and Realization of Particle Swarm Optimization Based on Pheromone Mechanism. Acta Automatica Sinica 35, 1410–1419 (2009)
9. Krohiling, R.A.: Gaussian Particle Swarm with Jumps. In: IEEE Congress on Evolutionary Computation, pp. 1226–1231. IEEE Press, New York (2005)
10. Krohiling, R.A., Mendel, E.: Bare Bones Particle Swarm Optimization with Gaussian or Cauchy jumps. In: IEEE Congress on Evolutionary Computation, pp. 3285–3291. IEEE Press, New York (2009)
11. Secrest, B.R., Lamont, G.B.: Visualizing Particle Swarm Optimization-Gaussian Particle Swarm Optimization. In: Swarm Intelligence Symposium, pp. 198–204. IEEE Press, New York (2003)

An Automatic Niching Particle Swarm for Multimodal Function Optimization

Yu Liu[1,2], Zhaofa Yan[1,2], Wentao Li[1,2], Mingwei Lv[1,2], and Yuan Yao[3]

[1] School of Software, Dalian University of Technology, Dalian 116024, P.R. China
[2] Institute of IT Service Engineering and Management, Dalian 116024, P.R. China
[3] Shanghai Key Laboratory of Machine Automation and Robotics
yuliu@tsinghua.org.cn

Abstract. Niching is an important technique for mutlimodal optimization. This paper proposed an improved niching technique based on particle swarm optimizer to locate multiple optima. In the proposed algorithm, the algorithm inspired from natural ecosystem form niches automatically without any prespecified problem dependent parameters. Experiment results demonstrated that the proposed niching method is superior to the classic niching methods which are with or without niching parameters.

1 Introduction

Particle swarm optimization [1,2] like evolutionary algorithms (EAs) is effective and robust to solve difficult optimization problems. However, the original algorithm as well as most of variants were designed to obtain only one global optimum. Those variants can only locate a single optimum, because they were designed to converge to only one optimum by the information sharing mechanism. However, there are many real world problems which have multiple optimum solutions. Those solutions are equally good and the practitioner may need several solutions to provide several options. Optimization methods which can get more than one optimum solutions are desired to satisfy those needs. At the same time, locating multiple optima can reduce the chance of getting trapped in local optima.

Mutlimodal Function is the functions which have many optima, or exist one global optimum and several useful local optima in the feasible solution space. There are many such problems in the real world, such as rule discovery in Data Mining, neural network ensemble and so on. Mutlimodal Optimization (MMO) is used to handle mutlimodal problems, and can obtain several solutions. Evolutionary Algorithm (EA) and Swarm Intelligence (SI) based on population search have a great potential to locate multiple optima simultaneously in the search space.

In recent years, numerous techniques have been proposed to handle mutlimodal optimization problems. These methods are referred as niching techniques. Niching techniques are incorporated into the traditional EA to locate multiple solutions, and sometimes, they also incorporated into EA aiming to avoid trapping in local optima as well as multi-objective optimization to maintain diversity. Recently, there are several classic niching techniques which have been proposed, such as, deterministic crowding [3], fitness sharing [4], clustering [5], clearing [6] and speciation [7]. Those techniques are proposed in the literature of EAs, some of which were introduced to PSO to enable PSO to handle mutlimodal optimization problems, such as SPSO [7] and MSPSO [8], [9]. Those methods have some deficiencies, such as some methods need optima distribution to set the radius to a optimal value.

To overcome the existing deficiencies especially the prespecified problem dependent parameters, in this paper, an algorithm based on PSO without any parameters was proposed. This method was inspired from the natural ecosystem, the mechanism that can form niches automatically. Under this natural ecosystem mechanism, every particle learns from its nearest neighbor and impacts its neighbor.

2 Related Works

2.1 Particle Swarm Optimization

PSO was proposed by emulating the social behavior. Each individual represents a point in the feasible solution space. The individual also called particle represents a potential solution. This kind of information sharing mechanism guides the population moving toward the optimum area. In the original PSO, the velocity V_i^d ad position X_i^d of the dth dimension of the ith particle are updated as equation 1, 2.

$$\begin{aligned} V_i^d =& V_i^d + c_1 \times rand1_i^d \times \left(pbest_i^d - X_i^d\right) \\ &+ c_2 \times rand2_i^d \times \left(gbest^d - X_i^d\right) \end{aligned} \quad (1)$$

$$X_i^d = X_i^d + V_i^d \quad (2)$$

Where $X_i = \left(X_i^1, X_i^2, \ldots, X_i^d\right)$ is the position of the ith particle; $V_i = \left(V_i^1, V_i^2, \ldots, V_i^d\right)$ represents velocity of the ith particle. $pbest_i$ is the best postion that the ith particle experienced; $gbest$ is the best position that the whole population experienced. c_1 and c_2 are two acceleration constants representing the attraction weight of *pbest* and *gbest*. $rand1_i^d$ and $rand2_i^d$ are two random numbers between 0 and 1. In those two equations, we can find that each particle was guided by the *pbest* and *gbest*.

PSO has attracted a high level of interest, and many variants were proposed. One of the most famous variants called constriction-factor variant [10]. It is basically the same as the original version, except that the velocity is updated by equation 3 instead of equation 1.

$$\begin{aligned} V_i^d =& \chi \left(V_i^d + c_1 \times rand1_i^d \times \left(pbest_i^d - X_i^d\right)\right. \\ &\left. + c_2 \times rand2_i^d \times \left(gbest^d - X_i^d\right)\right) \end{aligned} \quad (3)$$

Where

$$\chi = \frac{2}{2 - \varphi - \sqrt{\varphi^2 - 4\varphi}} \quad and \quad \varphi = c_1 + c_2, \quad \varphi > 4.0.$$

There are many variants based on choosing *gbest*, two well-known of which are *gbest* and *lbest* model. The methods above are global PSO, which are a fully connected and informed topology, every particle communicates with the whole population. While in *lbest* model, particles only interact with the particles in their neighborhood. The best one in their neighborhood called *lbest* is chosen to take the place of *gbest* in global model. PSOs with various population topologies [11] were studied. In the *lbest* model, the velocity is updated by the equation 4 instead of 1.

$$\begin{aligned} V_i^d =&\chi \left(V_i^d + c_1 \times rand1_i^d \times \left(pbest_i^d - X_i^d\right)\right. \\ &\left.+c_2 \times rand2_i^d \times \left(lbest^d - X_i^d\right)\right) \end{aligned} \tag{4}$$

In this equation, *lbest* is the best particle in the i-th neighborhood. In this version, particles can do a local search rather than converge to a area quickly which may lead to premature. This can somehow avoid premature and maintain population diversity.

In fact, niching is a technique used to form some species, which is essentially a local model of neighborhood. To some degree, neighborhood topologies can lead to niching methods.

2.2 Automatic Niching Particle Swarm (ANPSO)

Elite strategy was used to keep diversity when dealing with multi-objective optimization [12]. When dealing with mutlimodal optimization, diversity is also important. In this paper, elite was introduced to cooperate with PSO to handle multimodal optimization problems. Roulette Wheel Selection algorithm was used to choose the individual into elite. In this method, the individual with high fitness is more likely to be selected into the elites.

When dealing with mutlimodal optimization problems, many variants divide the swarm into some small subswarms aiming to search for different optima. In our research, individual and its closest neighbor are more likely to be in the same group. Figure 1 and 2 give an example of 30 indiviual distribute in the landscape of Himmelblau function [13]. It can be concluded that individual and its closest neighbor are more likely to locate in the same subswarm, we call it mechnism *nearest neighbor*. In this paper, elite an *nearest neighbor* was introduced to cooperate with PSO. Elite individuals were chosen from pbests, the pbests with high fitness were more likely to be chosen into elites, and they were considered as exemplar to be learned from. When diciding which individual in elite to be learned from, nearest neighbor mechanism was implimented.

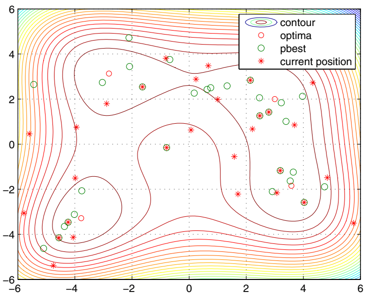


Fig. 1. Iteration 5

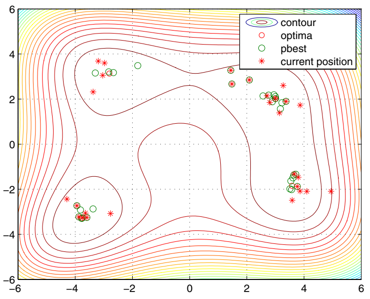


Fig. 2. Iteration 15

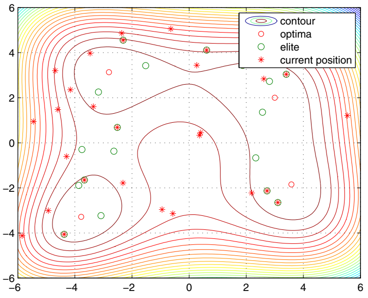


Fig. 3. Iteration 5

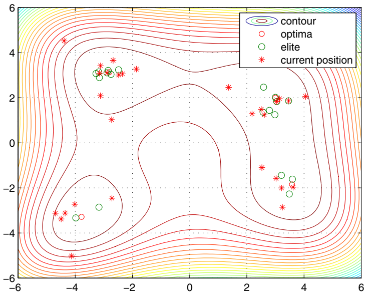


Fig. 4. Iteration 15

In Figure.3 and 4, we can see that individual and its nearest neighbor in elite are more likely to be in the same subswarm. The proposed method is given in Algorithm 1.

The mechanism can also be viewed in the aspect of natural ecosystems and social system as PSO. In natural ecosystems or social system, individuals get information and learn from elite individuals around as well as from their experience. Every individual can be elited, but the individuals with high fitness are more likely to be treated as elite. This mechnism is emulated by Roulette Wheel Selection algorithm. After identifying the elite, individuals are more likely to learn from the elite that share most common with itself. Since the characteristic of particle in swarm is identified by its location, it can be considered that the closer two particle, the more common they share. This can be emulated by learning from the closest elite. Roulette Wheel Selection algorithm was used to simulate the chosen elites. In this algorithm, *pbest*s have high fitness are more likely to be chosen into elite. After choosing the elite, the nearest elite of i-th was chosen as *lbest* to be learned from.

Algorithm 1. Pseudo Code of ANPSO

```
for iteration $\leq$ maxGeneration do
    for all particle in the swarm do
        Calculate the elite using Roulette Wheel Selection algorithm;
        Choose the nearest one of elite as the lbest for $i$-th particle;
        Update velocity and position by equation 4 and 2;
    end for
end for
```

3 Experiment

Some widely used classic functions were used to evaluate the ability of our proposed algorithm, including simple one dimensional and complex two dimensional functions. The primary aims of our experiments are to demonstrate that the proposed algorithm can form niches automatically without any prespecified parameters, to demonstrate that our method can locate multiple solutions, to compare with the niching algorithms with and without parameters, and to demonstrate that our niching method without parameter superior to the classic niching method without parameters.

The proposed algorithm were compared with two classic PSO based niching algorithms.

- SPSO: This algorithm was a classic niching algorithm, which came from the classical GA based on niching algorithm (SCGA) [7] and relied on a prespecified speciation radius. It was proved superior to species conserving genetic algorithm (SCGA) [14].
- RPSO [15]: This algorithm was recently proposed classical niching algorithm without any niching parameters.

In the following of this section, we will describe the test functions used to conduct those experiments and the performance measurements used to estimate the performance of those algorithms.

3.1 Test Functions

The test functions used in those experiments were showed in the Table 1, those functions were categorized to 2 groups including simple and single dimensional functions and more complex two dimensional functions. Those classic functions are widely used.

3.2 Parameters

For function f_1 to f_8, we used 50 particles which are enough to handle those problem. All the mutlimodal algorithms run for 10000 function evaluations.

Table 1. Test Functions

Name	Test Function
Central Two-Peak Trap [16]	$f_1(x) = \begin{cases} 16x & \text{for } 0 \le x < 10, \\ 32(15-x) & \text{for } 10 \le x < 15, \\ 40(x-15) & \text{for } 15 \le x \le 20. \end{cases}$
Five-Uneven-Peak Trap [7]	$f_2(x) = \begin{cases} 80(2.5-x) & \text{for } 0 \le x < 2.5, \\ 64(x-2.5) & \text{for } 2.5 \le x < 5.0, \\ 64(7.5-x) & \text{for } 5.0 \le x < 7.5, \\ 28(x-7.5) & \text{for } 7.5 \le x < 12.5, \\ 28(17.5-x) & \text{for } 12.5 \le x < 17.5, \\ 32(x-17.5) & \text{for } 17.5 \le x < 22.5, \\ 32(27.5-x) & \text{for } 22.5 \le x < 27.5, \\ 80(x-27.5) & \text{for } 27.5 \le x \le 30. \end{cases}$
Equal Maxima [13]	$f_3(x) = \sin^6(5\pi x)$
Decreasing Maxima [13]	$f_4(x) = \left(e^{-2\log(2)\times\left(\frac{x-0.1}{0.8}\right)^2}\right) \times \sin^6(5\pi x)$
Uneven Maxima [13]	$f_5(x) = \sin^6\left(5\pi\left(x^{3/4} - 0.05\right)\right)$
Uneven Decreasing Maxima [13]	$f_6(x) = \left(e^{-2\log(2)\times\left(\frac{x-0.08}{0.854}\right)^2}\right) \times \sin^6\left(5\pi\left(x^{3/4} - 0.05\right)\right)$
Himmelblau function [13]	$f_7(x,y) = (x^2+y-11)^2 - (x+y^2-7)^2$
Six-Hump Camel Back [17]	$f_8(x,y) = \left(4 - 2.1x^2 + \frac{x^4}{3}\right)x^2 + xy + \left(-4+4y^2\right)y^2$

All the variants of PSO were compared in this paper by using the standard constricted version. If any particle with a position x_i exceeding the boundary of the variable range, its position is reset to a value that is twice of the boundary subtracting x_i, this method is also be used by RPSO [15]. For SPSO, we set the species radius r to a value that is less than the distance between the two closest particles.

3.3 Performance Measure

The performance of the algorithms is estimated in two aspects, the success probability of a run and the computation each method spent to locate all the optima.

To identify a located optimum, we prespecify a *accept value* ε (typically $\varepsilon > 0$), representing how close the fitness of the computed solution to that of the known global optima that the solution can be accepted as a optimum. It can be formally presented in equation 5.

$$|f(solution) - goal| \leq \varepsilon \tag{5}$$

In equation 5, the *solution* is the fitness of located optimum and the *goal* is the fitness of global optimum. If the difference between fitness of the solution and goal is not larger than ε, then the solution was acceptted as a optimum.

The success probability in this paper is also called *success rate* in other papers. which is the percentage of runs in which all the global optima are successfully located. The *success rate* relies on the prespecified ε, for a large value of ε, which means relaxed acceptted condition lead to a higher success rate, while low value means harsh acceptted condition lead to a lower success rate.

In this paper, evaluation is considered to be the only source of computation, so computation is represented by the number of evaluations spent to reach the goal. In this paper, it was represented by the average number of evaluations of all the runs.

4 Result

4.1 Ability to form Niching Automatiocally

Fig.5(a)-(c) shows a simulation run of the ANPSO on Himmelblau function using 50 particles. In those figures, red circle represent the optimum solution, red '*' represent *current position*, green circle represent *pbest*, the blue lines connect the *current position* and *pbest* togother.We can see in the figure that, at the beginning, all the distribute in the search space randomly, and after 10 generation, they begin to cluster into subswarms. At the time of generation 20, there are subswarms in the area of global optima. This shows ANPSO having the ability to forming niches automatically.

4.2 1-Dimensional Function

For f_1, when ε set to 1, all of the three methods are able to locate all of the global optima with a success rate 100%. However, when raise the threshold, such as set ε to 0.001, only ANPSO have a success rate of 100%, and ANPSO is also faster than the other two methods.

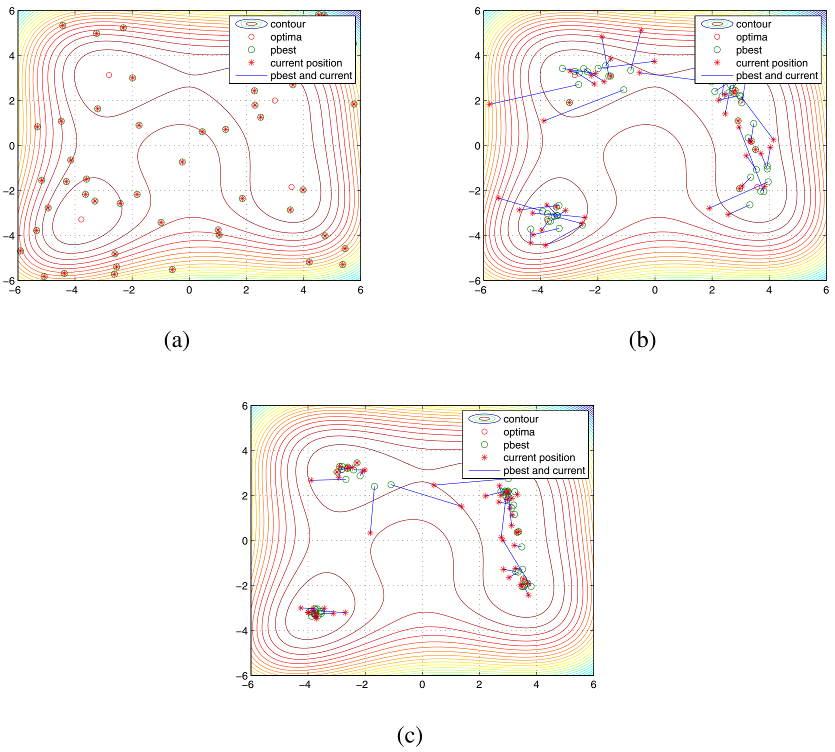


Fig. 5. A simulation run of the ANPSO on Himmelblau function using 50 particles: (a) Generation 1 (b) Generation 10 (c) Generation 20

There are 2 global optima, and 3 local optima in f_2, those give some challenges to the optimization algorithms. All of the three methods are unable to locate all optima with high success rate, while ANPSO have the highest success rate and lowest computation.

f_3, f_4, f_5 and f_6 were classic one dimensional test functions, introduced by Deb [13]. Both of f_3 and f_5 have five global optima, but the optima of f_3 are evenly distributed, while optima of f_5 are unevenly distributed. This gives a great challenge to the prespecified radius needed method to specify optimal radius. A too large or too small radius will either unable to distinguish two niches, or to create too many niches, and lead to lose some optima or cost large computation. f_4 and f_6 are functions with a single global optimum with 4 local optima. Those two functions are used to test whether the optimization algorithms can avoid local optima or not. SPSO and ANPSO have an equivalent performance superior to RPSO, while SPSO need a specified radius.

4.3 2-Dimensional Function

Most of existing methods have good performance on single-dimensional functions, 2-dimensional functions are challenges to the mutlimodal optimization algorithms. f_7 named Himmelblau have 4 equally global optima, RPSO missed some optima in some runs, have a lower success rate of 56%, ANPSO is superior to RPSO the one also doesn't need any parameters. However, thanks to the prespecified radius, SPSO also gets a 100% success rate and the computation is as low as ANPSO.

4.4 Success Probability and Computation

Table 2 summarizes the success rates on function f_1 to f_8, A population size of 50 was used. The PSO niching algorithms were run until all known global optima were found, or reach the maximum of 10000 evaluations were reached. Both of ANPSO and RPSO do not need any parameters, while SPSO needs r as radius of niche. Those parameters are used for measuring performance. In all of the run, ANPSO have a highest success rate, though SPSO takes the advantage of radius. Compared with the RPSO, which doesn't need any parameters, ANPSO have a much better performance.

Table 2. Success Rates

Function	ε	r	ANPSO	RPSO	SPSO
f_1	0.1	5	100%	98%	46%
f_2	0.1	10	96%	46%	44%
f_3	0.001	0.05	100%	100%	100%
f_4	0.001	0.05	100%	100%	100%
f_5	0.001	0.05	100%	88%	100%
f_6	0.001	0.05	100%	100%	100%
f_7	0.001	2	100%	56%	100%
f_8	0.00025	0.5	100%	100%	100%

Table 3. Averaged Evaluation times Over 50 Runs (Mean and One Standard Error) for Results Presented in Table 2

Function	ANPSO	RPSO	SPSO
f_1	**896**	1450	6461
	(361.10)	(1409)	(4222.1)
f_2	**2628**	6737	7285
	(2124.4)	(3813.9)	(3833.6)
f_3	670	1424	**446**
	(325.45)	(1202.9)	(190.82)
f_4	282	285	**248**
	(158.04)	(168.80)	(144.26)
f_5	637	2334	**467**
	(235.56)	(3154.4)	(170.12)
f_6	346	367	**338**
	(214.24)	(181.72)	(277.85)
f_7	2635	6076	**2598**
	(994.9)	(3594.20)	(447.32)
f_8	1238	1319	**1199**
	(211.54)	(246.38)	(289)

Table 3 gives the number of evaluations for each experiment to achieve the success rates presented in Table 2. From the table, we can see that, thanks to the prespecified radius, the computation of ANPSO is a little larger than SPSO, while compared the parameter independent RPSO, the computation is much lower, while SPSO needs a prespecified and problem dependent radius.

5 Conclusion and Future

This paper proposed a new niching method inspired from natural ecosystem, and incorporating with Particle Swarm Optimization, implement ANPSO. The proposed method does not rely on any prespecified parameters, and can form niches automatically. Compared with the other classic niching methods, it is superior to those methods in terms of success rate and computation time without any prespecified parameters. It have a high success rate and low computation as SPSO, and parameters independent as RPSO.

Acknowledgements

This work was under Grand by the Natural Science Foundation of China (No.60803074, 60673024), the Natural Science Foundation of Liaoning Province (No.20082172) and open research issue from Shanghai Key Laboratory of Machine Automation and Robotics.

References

1. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the sixth international symposium on micro machine and human science, vol. 43. IEEE, New York (1995)
2. Kennedy, J., Eberhart, R., et al.: Particle swarm optimization. In: Proceedings of IEEE international conference on neural networks, vol. 4, pp. 1942–1948. IEEE, Piscataway (1995)
3. Mahfoud, S.: Crowding and preselection revisited. Urbana (51) 61801
4. Goldberg, D., Richardson, J.: Genetic algorithms with sharing for multimodal function optimization. In: Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application table of contents, pp. 41–49. L. Erlbaum Associates Inc., Hillsdale (1987)
5. Yin, X., Germany, N.: A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization. In: Artificial neural nets and genetic algorithms: proceedings of the international conference, Innsbruck, Austria, vol. 6, p. 450. Springer, Heidelberg (1993)
6. Petrowski, A.: A clearing procedure as a niching method for genetic algorithms. In: Proceedings of the 1996 IEEE International Conference on Evolutionary Computation, Citeseer, pp. 798–803 (1996)
7. Li, J., Balazs, M., Parks, G., Clarkson, P.: A species conserving genetic algorithm for multimodal function optimization. Evolutionary computation 10, 207–234 (2002)
8. Iwamatsu, M.: Locating all the global minima using multi-species particle swarm optimizer: The inertia weight and the constriction factor variants. In: IEEE Congress on Evolutionary Computation, CEC 2006, pp. 816–822 (2006)

9. Iwamatsu, M.: Multi-species particle swarm optimizer for multimodal function optimization. IEICE Transactions on Information and Systems 89, 1181–1187 (2006)
10. Clerc, M., Kennedy, J.: The particle swarm - explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation 6, 58–73 (2002)
11. Kennedy, J., Mendes, R.: Population structure and particle swarm performance, pp. 1671–1676 (2002)
12. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: NSGA-II. IEEE Transactions on Evolutionary Computation 6 (2002)
13. Deb, K.: Genetic algorithms in multimodal function optimization. Master's thesis, The University of Alabama (1989)
14. Li, X.: Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization. In: Deb, K., et al. (eds.) GECCO 2004. LNCS, vol. 3102, pp. 105–116. Springer, Heidelberg (2004)
15. Li, X.: Niching without niching parameters: Particle swarm optimization using a ring topology. IEEE Transactions on Evolutionary Computation 14, 150–169 (2010)
16. Ackley, D.: An empirical study of bit vector function optimization. Genetic algorithms and simulated annealing 1, 170–204 (1987)
17. Michalewicz, Z.: Genetic algorithms+ data structures. Springer, Heidelberg (1996)

An Availability-Aware Task Scheduling for Heterogeneous Systems Using Quantum-behaved Particle Swarm Optimization

Hao Yuan[1], Yong Wang[1], and Long Chen[2]

[1] Electronic Commerce & Modern Logisties Key Laboratory, Chongqing University of Posts and Telecommunications, Chongqin, 400065
[2] School of Computer Science, Chongqing University of Posts and Telecommunications, Chongqin, 400065
{yln2446@21cn.com}

Abstract. A major challenge in task scheduling is the availability of resources. In a heterogeneous environment, where processors operate at different speeds and are not continuously available for computation, achieving a better make-span is a key issue. The existing algorithm SSAC has proved to be a good trade-off between availability and responsiveness while maintaining a good performance in the average response time of multiclass tasks. But the makespan may be influenced due to load imbalance. In this paper we proposed approach try to further optimize this scheduling strategy by using quantum-behaved particle swarm optimization. And compared with SSAC and MINMIN in the simulation experiment; results indicate that our proposed technique is a better solution for reducing the makespan considerably.

Keywords: Quantum-behaved Particle Swarm Optimization, Task Scheduling, Heterogeneous Systems.

1 Introduction

P2P networks, Clusters and grids are a popular paradigm for parallel and distributed computing [1]. In a grid environment, the resources are geographically distributed, managed and owned by different organizations with their own policies and interconnected via the Internet [2]. This introduces a number of resource management issues and scheduling strategies in the domain of security, resources and heterogeneity [3]. The resource management and scheduling systems for grid computing need to manage resources depending on several factors like consumers or the owners and hence continuously adapt to changes in resource availability [4]. A major challenge in task scheduling is the availability of resources. In a heterogeneous environment, where processors operate at different speeds and are not continuously available for computation, achieving a better make-span is a key issue [5,6].

Quantum Particle Swarm Optimization algorithm [7,8] (QPSO) is a new optimum method that combines quantum computation with Particle Swarm Optimization (PSO). It appears strong life-force and be valuable for research. Quantum computation absorbed many essential characters of quantum mechanics, which improved the computation efficiency, and become a brand new model of computation. Hence, QPSO has greatly enhanced the efficiency of search and can compensate for the lack of PSO and it has a wide research foreground.

The existing algorithm SSAC has proved to be a good trade-off between availability and responsiveness while maintaining a good performance in the average response time of multiclass tasks. But the makespan may be influenced due to load imbalance [9]. In this paper we proposed approach try to further optimize this scheduling strategy by using quantum-behaved particle swarm optimization. And compared with SSAC and MINMIN in the simulation experiment; results indicate that our proposed technique is a better solution for reducing the makespan considerably.

The rest of this paper is organized as follows: section 2 describes the background of QPSO and model of task scheduling in heterogeneous systems; in section 3, QPSO for task scheduling in heterogeneous systems is introduced; in section 4, Experimental results and analyses are given. At last, the conclusions and future work are given in section 5.[1]

2 Background

2.1 Quantum-behaved Particle Swarm Optimization

PSO was originally developed by Kennedy and Eberhart [10]. In PSO, a swarm consists of a set of particles, where each particle represents a potential solution. Currently, Jun sun [7,8] etc proposed quantum-behaved particle swarm optimization (QPSO) based on quantum delta potential well. The basic idea is assumed that all particles moving in quantum field will converge to a minimal area where potential power is infinite. The QPSO framework presented as follows.

The QPSO eliminate velocity's notion, because particles' state is represented by wave function $\psi(x,t)$ in the quantum time-space framework. Example, in 3-dimensional wave function denoted as [11]

$$|\psi|^2\, dxdydz = Qdxdydz \tag{1}$$

Where Qdxdydz the probability that measurement of the particle's position at the time t finds it in the volume element about the point (x, y, z). Particles have not fixed position in quantum time space. We can consider that particle may appear in anywhere simultaneously, and it collapses to an assured position as soon as macroscopically measuring by corresponding probability. $|\psi|^2$ is the probability density function satisfying

$$\int_{-\infty}^{+\infty} |\psi|^2\, dxdydz = \int_{-\infty}^{+\infty} Qdxdydz = 1 \tag{2}$$

[1] Yuan Hao (1970-), male (Han), lecturer, master, mainly engaged in computer networks, information security research. Wang Yong(1978 -), male (Han), Associate Professor, Ph.D., mainly engaged in computer networks, information security research. Chen Long(1970-), male (Han), Professor, Ph.D., mainly engaged in information security research.

Namely, any particle's probability of appearing in whole space is 1.
At first, algorithm generate a position vector P which locate between Pi and Pg, and P denoted as

$$\boldsymbol{P} = (\varphi_1 \boldsymbol{P}_i + \varphi_2 \boldsymbol{P}_g)/(\varphi_1 + \varphi_2) \tag{3}$$

Then update position vector according to probability density function [14].

$$\boldsymbol{x}_i(t+1) = \boldsymbol{P} \pm (1/g) \times [\ln(1/u)] \times |\boldsymbol{x}_i(t) - \boldsymbol{P}| \tag{4}$$

Where parameter g is the variable needs to be adjusted in algorithm, constrained by $g > \ln\sqrt{2}$. Another parameter u is a random number which range between (0, 1) and showed normal distribution. The probability of plus and minus in equation (4) is 50% respectively.

2.2 Model of Task Scheduling for Heterogeneous Systems

Now, we formulate the scheduling problem as a trade-off problem between availability and the mean response time. Thus, the proposed scheduling algorithm aims at improving system availability and maintaining an ideal response time of submitted tasks [12]. More formally, the problem of maximizing the availability of a heterogeneous system can be formulated as follows [13]:

$$\text{Maximize } A = \sum_{i=1}^{m} \left\{ \frac{\lambda_i}{\lambda} \exp\left[-\sum_{j=1}^{n} \left(p_{ij} \frac{\theta_j}{\mu_{ij}} \right) \right] \right\} \tag{5}$$

Subject to the following response time constraints:

$$\forall 1 \leq i \leq m, 1 \leq j \leq n, a_i \leq \xi_j : \text{ Miminize } TC_i \tag{6}$$

The above constraint is the response time constraints, each of which means that, among nodes whose availability shortage factor for class i equals zero, a node is chosen for the ith class in such a way as to minimize the mean response time of class i. Notice that the response time constraints can be satisfied by estimating the mean response times of class i on all candidate nodes whose availability shortage factor for class i is zero.

Computational heterogeneity captures the nature of heterogeneous computing platforms where the execution times of each task on different nodes are distinctive. Although each multiclass task has an availability requirement, computational nodes exhibit a variety of availability levels. For simplicity and without loss of generality, the availability levels and availability requirements are normalized in the range from 0 to 1.0. We introduce the concepts of computational heterogeneity and availability heterogeneity. The computational weight of class i on node j is defined as the ratio between its service rate on node j and the fastest service rate in the system. That is, the computational weight is expressed by:

$$w_{ij} = \mu_{ij} / \max_{k=1}^{n} (\mu_{ik}) \tag{7}$$

The computational heterogeneity of the ith class, that is, HCi, can be measured by the standard deviation of the computational weights. Thus, we have:

$$\mathrm{HC_i} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(w_i - w_{ij}\right)^2} \tag{8}$$

The computational heterogeneity can be expressed as follows:

$$\mathrm{HC} = \frac{1}{m}\sum_{i=1}^{m} HC_i \tag{9}$$

The heterogeneity of availability HA in a heterogeneous system is measured by the standard deviation of the availability offered by all the nodes in the system. Hence, HA is written as:

$$\mathrm{HA} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(\overline{\xi} - \xi_j\right)^2}\,, where\ \overline{\xi} = \left(\sum_{j=1}^{n}\xi_j\right)/\,n \tag{10}$$

3 Task Scheduling for Heterogeneous Systems Based on QPSO

If the model is m-jobs and n-machines, suppose that the searching space is m-dimensional and S particles form the colony. The ith particle represents an m-dimensional vector Xi (i =1,2, ..., s). It means that the ith particle locates at Xi which is one sequence in the searching space. The position of each particle is a potential result. We could calculate the particle's fitness by putting its position into a designated objective function. When the fitness is lower, the corresponding Xi is better. The ith particle's "flying" velocity is also an m-dimensional vector, denoted as Vi. Denote the best position of the ith particle as pi, and the best position of the colony as pg, respectively. In QPSO, each particle of the swarm shares mutual information globally and benefits from discoveries and previous experiences of all other colleagues during the search process.

Steps of QPSO to minimizing the makespan:

Step 1: Let population size be psize, the termination generation number Maxgen. Iterative number be k = 0, Give birth to psize initializing particles. Calculate each particle's fitness value of initialization population, and let first generation pi be initialization particles, and choose the particle with the best fitness value of all the particles as the pg(gBest).

Step 2: The parameter gbests and pbests are evaluated on functions. Choose the current nondominated from these variables and the solutions in archive file. Then update the archive set by deleting the solutions which dominated by new solutions and insert the nondominated into the archive file. On this step, if the capacity of A is overflowing, the solution will be deleted where in the densest area, which is implemented by Partition algorithm with O(nlogn) time complexity in this paper. If the fitness value is better than the best fitness value pi(pBest) in history, let the current value be the new pi(pBest). Choose the particle with the best fitness value of all the particles as the pg(gBest). If k = = Maxgen, go to Step 3, or else let k = k + 1, go to Step 2.

Step 3: Put out the pg. We see that there are two key steps when applying QPSO to the algorithm: the representation of the solution and the fitness function. The stop condition depends on the problem to be optimized.

Step 4: According to gbest and pbest generated by step 4 update the particle's position by formulae (3) and (4)

Step 5: Repeat from step 2, and add the parameter t to t + 1.

4 Experimental Results and Analysis

We compare our strategy with SSAC and MINMIN. The parameters used in experiment are shown in Table 1. These parameters are selected based on those used in the literature or represent real world heterogeneous systems. The performance metrics used to evaluate the performance of algorithm are described as:

(1)Makespan: It is the time difference between the start and finish of a sequence of jobs or tasks.

(2)Availability. It is the probability that the system is continuously performing at any random period of time.

Table 1. Parameters

Parameter	Values
Number of nodes	20
Number of tasks	32(32,48,64,128)
Mean task arrival rate	1.0
Node availability	0.1-1.0
Population size	50
Max. Generations	350
Intertia (w)	0.6
Constants (c1, c2)	2.5

(1)Experimental I

In this experiment, we study the behavior of our algorithm against the arrival rate of tasks in the ith class. We vary the mean arrival rate from 0.2 to 1.0 with an increment of 0.2 as in SSAC algorithm. Figure 1 and Figure 2 shows the results of three algorithms.

From Figure 1 it can be seen that our algorithm provides a better makespan than the other two algorithms. It is considerably a lesser than SSAC and is a bit higher than MINMIN. The reason is MINMIN algorithm schedules its tasks depending on those nodes that provide the earliest completion time. Whereas our new algorithm takes the nodes availability into account.

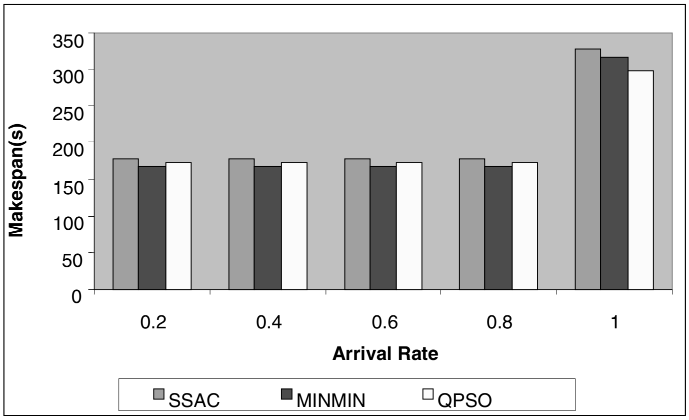


Fig. 1. Performance impact of arrival rate-Makespan

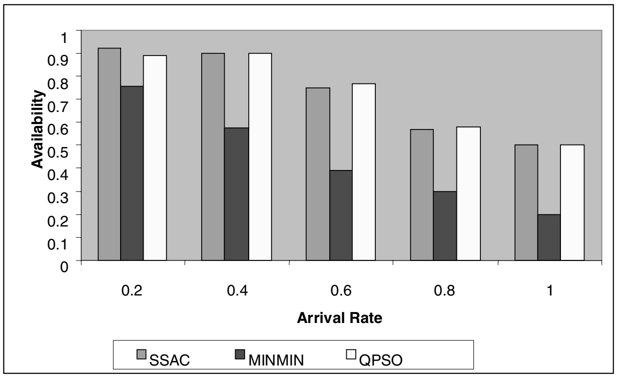


Fig. 2. Performance impact of arrival rate-Availability

(2)Experimental II
In this experiment the number of nodes was varied. This part focuses on the scalability of the algorithm. We vary the number of nodes in the simulated heterogeneous environment from 32 to 128. Figure 3 and Figure 4 are the resulting performance graphs. In Figure 3, it could be observed that as the number of nodes is increased the makespan also improves as now our algorithm can schedule the tasks appropriately. With more nodes in the environment, the nodes tend to get less workload to execute. Hence a noticeable improvement against SSAC.

MINMIN algorithm performs badly as opposed to SSAC and our algorithm. This result can be depicted in Figure 4. Since the algorithm proposed in availability-aware it makes best use of its resources.

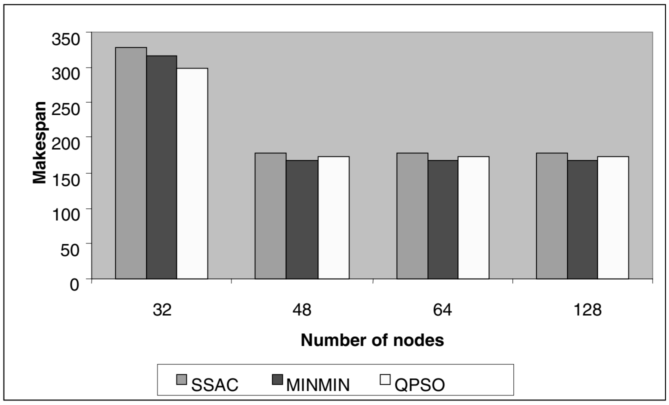


Fig. 3. Performance impact of Number of nodes-Makespan

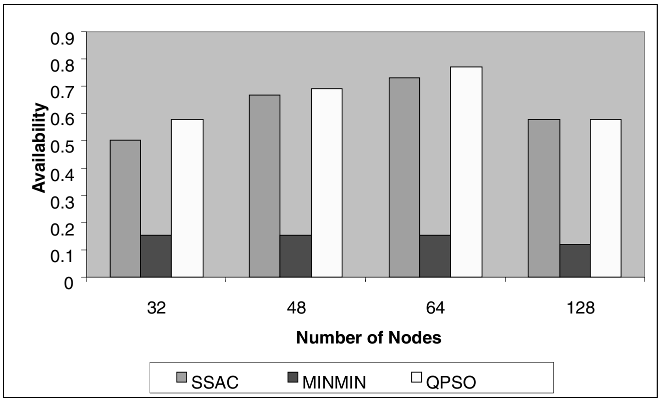


Fig. 4. Performance impact of Number of nodes-Availability

5 Conclusions

In this paper we proposed approach try to further optimize this scheduling strategy by using quantum-behaved particle swarm optimization. And compared with SSAC and MINMIN in the simulation experiment; results indicate that our proposed technique is a better solution for reducing the makespan considerably.

Acknowledgment

This research is supported by the National Natural Science Foundation of China (Grant No.60703035), Natural Science Foundation of Chongqing Science and Technology Commission (CSTC No.2009BB2288).

References

1. Srinivasan, S., Jha, N.K.: Safety and Reliability Driven Task Allocation in Distributed Systems. IEEE Transactions on Parallel and Distributed Systems 10(3), 238–251 (1999)
2. Lee, C.-Y.: Two-Machine Flowshop Scheduling with Availability Constraints. European J. Operational Research 114(2), 420–429 (1999)
3. Topcuoglu, H., Hariri, S., Wu, M.-Y.: Performance-Effective and Low-Complexity Task Scheduling for Heterogeneous Computing. IEEE Transactions on Parallel and Distributed Systems 13(3), 260–274 (2002)
4. Hagras, T., Janecek, J.: A high performance, low complexity algorithm for compile-time task scheduling in heterogeneous systems. Parallel Computing 31(7), 653–670 (2005)
5. Song, S.-S., Hwang, K., Kwok, Y.-K.: Risk-Resilient Heuristics and Genetic Algorithms for Security-Assured Grid Job Scheduling. IEEE Transactions on Computers 55(6), 703–719 (2006)
6. Qin, X., Xie, T.: An Availability-Aware Task Scheduling Strategy for Heterogeneous Systems. IEEE Transactions on Computer 57(2), 188–199 (2008)
7. Sun, J., Xu, W.B.: A Global Search Strategy of Quantum-behaved Particle Swarm Optimization. In: Proceedings of IEEE Conference on Cybemetics and Intelligent Systems, pp. 111–116 (2004)
8. Sun, J., Feng, B., Xu, W.B.: Particle Swarm Optimization with Particles Having Quantum Behavior. In: Proceedings of Congress on Evolutionary Computation, pp. 325–331 (2004)
9. Ding, F., Li, K.: An Improved Task Scheduling Algorithm for Heterogeneous Systems. In: Proceedings of the 2009 International Joint Conference on Computational Sciences and Optimization, pp. 90–94. IEEE Computer Society, Los Alamitos (2009)
10. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: The 6th Int. Symposium on Micro Machine and Human Science, Nagoya, Japan, pp. 39–43 (1995)
11. Liu, H., Xu, S., Liang, X.: A Modified Quantum-Behaved Particle Swarm Optimization for Constrained Optimization. In: International Symposium on Intelligent Information Technology Application Workshops, pp. 531–534 (2008)
12. Kacem, I., Sadfi, C., El-Kamel, A.: Branch and Bound and Dynamic Programming to Minimize the Total Completion Times on a Single Machine with Availability Constraints. In: Proc IEEE Int'l Conf. Systems, Man and Cybernetics, pp. 1657–1662 (2005)
13. Deepa, R., Srinivasan, P.T., Doreen Hephzibah Miriam, D.: An Efficient Task Scheduling Technique in Heterogeneous Systems Using Self-Adaptive Selection-Based Genetic Algorithm. In: Proceedings of the international symposium on Parallel Computing in Electrical Engineering, pp. 343–348. IEEE Computer Society, Los Alamitos (2006)

A Novel Encoding Scheme of PSO for Two-Machine Group Scheduling

Cheng-Dar Liou and Chun-Hung Liu

Department of Business Administration, National Formosa University,
64,Wunhua Rd., Huwei, Yunlin County
63201, Taiwan, ROC
cdliou@nfu.edu.tw

Abstract. This paper investigates the two-machine flow shop group scheduling problem with the transportation times and sequence-dependent setup times considerations. The objective is to minimize the total completion time. In this paper, a novel encoding scheme of PSO for flow shop group scheduling is proposed to effectively solve various instances with group numbers up to 15. Note that the proposed encoding scheme simultaneously determines the sequence of jobs in each group and the sequence of groups. Three different lower bounds are developed to evaluate the performance of the proposed PSO algorithm. Limited numerical results show that the proposed PSO algorithm performs well for all test problems.

Keywords: Group scheduling, Transportation times, Sequence-dependent, Setup times, PSO.

1 Introduction

Production Scheduling based on the group technology concept is called group scheduling (GS) in which those parts requiring a similar production process are grouped in the same group. The objective of GS problem (GSP) is to identify the sequence of groups as well as the sequence of jobs in each group such that the total completion time is minimized. Unlike the other classic scheduling problems, GSP must simultaneously determine the sequence of jobs in each group and the sequence of groups. As jobs that belong to the same group are similar, changing from one job to another in the same group requires negligible setup time or it can be included along with the processing time. However, a change from one group to another would require spending a significant amount of time that cannot be ignored. Moreover, the required setup time for a machine to process a group depends on the group previously processed on that machine. Such a GSP is known as a flow shop sequence- dependent group scheduling problem (FSDGSP). Garey et al. [1] proved that the multi-stage flow shop job scheduling problem with minimized total flow time is an NP-hard problem. A two-machine GSP belongs to a multi-stage flow shop scheduling problem. Therefore, the two-machine GSP is an NP-hard problem. Because two-machine FSDGSP generalizes two-machine GSP, it is also an NP-hard problem.

In practice, there are several FSDGSP applications, e.g. the automobile manufacturing paint shop. Recent studies also show another application for electronic manufacturing. Schaller et al. [2] discussed a printed circuit board industrial problem in which a major setup is required to switch from one group of printed circuit boards to another. References in Logendran et al. [3] are relative studies to this field.

Particle swarm optimization (PSO) algorithm was developed by Kennedy and Eberhart in 1995 [4]. It was used to simulate the social behavior of organisms in a flock of birds or a school of fish and was introduced as a population-based searching technique. As an evolutionary algorithm, PSO conducts a search through updating a population (called a swarm) of individuals (called particles). The relationship between the swarm and particles in PSO is like the relationship between a population and chromosomes in GA (Genetic Algorithm). In PSO, the problem solution space is described as a search space and each position in the search space is a possible solution for the problem. When PSO is applied to an optimization problem with m variables, the solution space can be formulated as an m-dimensional search space and the value of the jth variable is formulated as the position on jth dimension. All particles cooperate to discover the best position (solution) in the search space.

The main advantages of PSO are that it contains a simple structure and is immediately accessible for practical applications, ease of implementation, robustness and speed in acquiring solutions. Therefore, the PSO algorithm has high potential for solving this two-machine FSDGSP.

The purposes of this paper are as follows:

(1) To first consider the two-machine FSDGSP with job transportation times between machines, and sequence-dependent setup times. Clearly, the studied two-machine FSDGSP generalizes the typical two-machine flow shop GSPs.
(2) To propose an encoding scheme of PSO for solving the two-machine FSDGSP. Numerical results for various instances, with group numbers up to 15, are reported and discussed.
(3) To develop three lower bounds for the two-machine FSDGSP.

2 Problem Description

2.1 Notations

The following notations are introduced to characterize the problem.

M_k	the kth machine, $k \in \{1, 2\}$
$\underline{n}$	the number of groups
G_i	the set of jobs in group i, $i = 1, 2,\ldots, \underline{n}$
n_i	the number of jobs in group G_i, $i = 1, 2,\ldots, \underline{n}$
J_{ij}	the jth job in group G_i, $j = 1, 2,\ldots, n_i$
a_{ij}	the processing time of job J_{ij} in group G_i on M_1 1, $j = 1, 2,\ldots, n_i$
b_{ij}	the processing time of job J_{ij} in group G_i on M_2, $j = 1, 2,\ldots, n_i$
t_{ij}	the transportation time of moving job J_{ij} in group G_i from M_1 to M_2, $j = 1, 2,\ldots, n_i$
s_{pjm}	the setup time of group G_j on M_m, if group G_j is processed immediately after group G_p, m=1, 2

T_i the minimum time length for processing jobs in group G_i determined by Maggu and Das's algorithm [5]

The assumptions for the studied two-machine FSDGSP are the same with those in Logendran et al. [3].

2.2 Objective Function

Yang et al. [6] proposed a polynomial time algorithm to solve two-machine flow shop GSP with sequence-independent setup and removal times and a transportation time between machines was also considered. We extend their algorithm to solve the two-machine FSDGSP with sequence-dependent setup times and the transportation time between machines.

In the following, we define each group as a composition job, which is similar to the one presented in Yang et al. [6]. The composition job is defined such that there is no intermediate idle time among these operations on each machine. For a group G_i , which is processed after group G_k, the associated composition job is defined as a processing vector (α_i, β_i, δ_i), where

$$\alpha_i = T_i - \sum_{j=1}^{n_i} b_{ij} + s_{ki1} - s_{ki2} \tag{1}$$

$$\beta_i = T_i - \sum_{j=1}^{n_i} a_{ij} \tag{2}$$

$$\delta_i = T_i + s_{ki1} - \max\{\alpha_i, 0\} - \max\{\beta_i, 0\} \tag{3}$$

Note that α_i and β_i may be negative. $|\alpha_i|$ and $|\beta_i|$ are the lengths of time periods when exactly one of the machines is busy and δ_i is the length of the time period when both machines are busy. For a group sequence S, let C_{ij} be the completion time of group G_i on machine j, j =1, 2, with group G_k being processed immediately before G_i. Then the completion time of group G_i can be obtained as follows.

$$C_{i1} = C_{k1} + \max\{\alpha_i, 0\} + \delta_i - \min\{0, \beta_i\} \tag{4}$$

$$C_{i2} = \max\{C_{k1} + \max\{\alpha_i, 0\}, C_{k2} - \min\{0, \alpha_i\}\} + \delta_i + \max\{\beta_i, 0\} \tag{5}$$

Therefore, we can evaluate the total completion time for a group sequence S using Eqs. (4) and (5).

3 Novel PSO Encoding Scheme

It is generally believed that the major difficulty in applying a PSO algorithm to combinatorial optimization problems is its continuous nature. To remedy this disadvantage, the smallest-position-value (SPV) rule borrowed from the random key representation [7] and ranked-order-value (ROV) [8] are usually utilized in the PSO algorithm to convert the continuous position values into a discrete job permutation. According to Liu et al, the ROV encoding scheme has better performance than the SPV encoding scheme for flow shop scheduling problems [8]. However, their coding scheme cannot directly apply to the group scheduling problems.

Our coding scheme is based upon the permutation of {1, 2,…, N}, where $N = n_1+n_2+\ldots+n_{\underline{n}}$. The following steps will transform any permutation of {1, 2,…, N} into a feasible group job permutation. The main steps are shown as follows.

Step 1. Generate N continuous random values from (0, 1) in a vector, say R, and rank R by ROV encoding scheme, shown in Liu et al. [8], in a vector, say *R_rank*.

Step 2. Generate a group vector *GV* as $[\underbrace{1,1,\ldots,1}_{n_1}, \underbrace{2,2,\ldots,2}_{n_2}, \underbrace{3,3,\ldots,3}_{n_3}\ldots \underbrace{\underline{n},\underline{n},\ldots,\underline{n}}_{n_{\underline{n}}}]$.

Step 3. Compute group value, *G_value*, by *G_value*(i) = *GV*(*R_rank*(i)), i = 1, 2,…, N.

Step 4. *G_order* = ∅. For i = 1, 2,…, N, if *G_value*(i) = k, then
(1) append *R_rank*(i) to G_k, k = 1, 2, …, $\underline{n}$.
(2) append k to *G_order* if $k \notin$ *G_order*.

Step 5. The order of groups is *G_order* and the order of jobs in group k is G_k, k = 1, 2,…, $\underline{n}$.

4 Three Lower Bounds for the Two-Machine FSDGSP

In this paper, three different lower bounds for two-machine FSDGSP are developed. Our lower bounds, borrowed the concept from Logendran et al. [3], are based upon the FSDGSP with a single group.

4.1 FSDGSP with a Single Group

For a single group i, the right-shifted optimal schedule without setup times of groups can be divided into three different parts, namely head (H_i), body (B_i), and tail (TL_i) for this group, as shown in Fig. 1. We may compute the lengths of H_i, B_i and TL_i for group G_i using the following equations.

$$H_i = \min\left(T_i - \sum_{j=1}^{n_i} b_{ij}, \sum_{j=1}^{n_i} a_{ij}\right) \tag{6}$$

$$B_i = \max\left(\sum_{j=1}^{n_i} a_{ij} - H_i, 0\right) \tag{7}$$

$$TL_i = \min\left(T_i - \sum_{j=1}^{n_i} a_{ij}, \sum_{j=1}^{n_i} b_{ij}\right) \tag{8}$$

In Eqs. (6)-(8), the last term are utilized to deal with the situation when T_i is greater than the sums of $\sum_{j=1}^{n_i} a_{ij}$ and $\sum_{j=1}^{n_i} b_{ij}$. Adding the setup times to the right-shifted schedule, then the lower bounds can be evaluated.

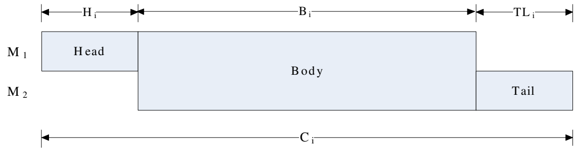


Fig. 1. H_i, B$_i$ and TL_i for group G_i.

4.2 Three Lower Bounds

Following a process similar to that of Logendran et al. [3], we develop the following M_1-based lower bound for the studied two-machine FSDGSPs.

$$\text{LB1} = C'_{SM1} + \sum_{i=1}^{\underline{n}} s_{i1}^{\min} + \sum_{i=1}^{\underline{n}} (H_i + B_i) \tag{9}$$

where

C'_{SM1} can be computed using the similar procedure in Logendran et al. [3],

$s_{i1}^{\min} = \min_{k \neq i} s_{ki1}$, $i=1, 2,\dots,\underline{n}$, H_i and B_i are computed using Eqs. (6) and (7).

The M_2-based lower bound is evaluated by using the similar concept of M_1-based, except that the head (H_i) is replaced by tail (TL_i) for each group G_i. The M_2-based lower bound is:

$$\text{LB2} = C'_{SM2} + \sum_{i=1}^{\underline{n}} s_{i2}^{\min} + \sum_{i=1}^{\underline{n}} (B_i + TL_i) \tag{10}$$

If the algorithm of Yang et al. [6] is replaced S_{i1} and S_{i2} with $s_{i1}^{\min}$and $s_{i2}^{\min}$, then its optimal makespan is a valid lower bound, LB3, for the FSDGSPs. Therefore, the solution quality of PSO solution can be evaluated by

$$\text{The PSO algorithm solution / max \{LB1, LB2, LB3\}} \tag{11}$$

5 Computational Results and Conclusions

To test the performance of the proposed PSO algorithm for more two-machine FSDGSPs, we tested various instances based upon a combination of $\underline{n}$, n_i, the ratio of setup times, and the range of transportation times as below:

(1) Number of groups ($\underline{n}$). The number of groups is generated from a uniform discrete distribution *DU* [2, 15]. Note that Schaller et al. [2] tested their experiments with largest $\underline{n}$ =10.
(2) Number of jobs in a group (n_i). The number of jobs in a group is generated based on *DU* [2, 10] Note that Schaller et al. [2] and Nasser et al. [9] set $2 \leq n_i \leq 10$
(3) The setup times. Two classes (intervals) of uniform setup times, namely, *DU* [1, 10] and *DU* [11, 20], are tested. Note that, if setup times for M_1 and M_2 are generated based on *DU* [1, 10] and *DU* [11, 20] respectively, then the average ratio of setup times, i.e. s_1/s_2 , is less than 1 ([(1+10)2]/[(11+20)/2] = 0.3548).
(4) The transportation times (t_{ij}). Two classes (intervals) of uniform transportation times, namely, *DU* [1, 15] and *DU* [16, 30], are tested.
(5) The processing times. We adopt random integers from *DU* [1, 20], which is similar to the one in Nasser et al. [9].

For each of the instances, we set the swarm size = 30×N, maximal iterations = 15×N, w = 1.0, c_1 = 3.14, c_2 = 3.14, $V_{\max}$ = 4.0, where N is the sum of jobs in all groups. The PSO algorithm was coded in MATLAB 7.0 and all results were computed using Intel-Pentium 2.4GHz PC with 1.97GB RAM. Tables 1 shows the numerical results for the instances using the PSO algorithm. Note that, for the convenience of comparison, we let $\bar{z}$ be the PSO solution and $\underline{z}$ = max (LB1, LB2, LB3). In addition, we let:

avg $\bar{z}/\underline{z}$ = the average ratio of $\bar{z}/\underline{z}$ for all three types of tests,
max $\bar{z}/\underline{z}$ = the maximum ratio of $\bar{z}/\underline{z}$ among 20 experiments,
min $\bar{z}/\underline{z}$ = the minimum ratio of $\bar{z}/\underline{z}$ among 20 experiments.
From Table 1, we observe and draw conclusions as follows:

(1) the avg $\bar{z}/\underline{z}$ values vary from 1.0067-1.0330 for all instances. This implies that the proposed PSO is robust for all test instances.
(2) the ratios of (max $\bar{z}/\underline{z}$)/(min $\bar{z}/\underline{z}$) vary from 1.000-1.011, 1.000-1.012, and 1.000-1.009, for instances of $S_1/S_2 < 1.0$, $S_1/S_2 = 1.0$ and $S_1/S_2 > 1.0$, respectively. This implies that the range of solutions of PSO is pretty small.
(3) the CPU times for solutions of experiments in Table 1 are 62.83s, and 17650s for 16 and 120 jobs, respectively. It implies that the proposed PSO is efficient for the test instances.

Table 1. The numerical results for various instances

NoG	NoJ		TJ	T	L1	L2	L3	avg $\bar{z}/\underline{z}$	CPUtimes
	From	To							
6	2	9	30	T1	1.000	1.014	1.005	1.0130	388.3
				T2	1.000	1.000	1.007	1.0173	388.3
8	3	10	46	T1	1.002	1.010	1.005	1.0187	1260.5
				T2	1.000	1.012	1.000	1.0217	1268.0
11	2	9	56	T1	1.011	1.006	1.006	1.0090	2374.8
				T2	1.006	1.008	1.005	1.0203	2360.1
7	4	10	48	T1	1.000	1.001	1.004	1.0153	1295.0
				T2	1.006	1.000	1.005	1.0190	1307.9
3	2	9	16	T1	1.000	1.000	1.000	1.0330	62.95
				T2	1.000	1.000	1.000	1.0067	62.83
12	2	10	82	T1	1.003	1.005	1.001	1.0077	6278.1
				T2	1.007	1.006	1.006	1.0113	6327.8
14	5	10	120	T1	1.003	1.004	1.005	1.0132	17426.2
				T2	1.003	1.003	1.004	1.0179	17650.3
10	2	10	55	T1	1.005	1.003	1.009	1.0267	2152.2
				T2	1.006	1.002	1.004	1.0143	2182.9
15	2	10	93	T1	1.006	1.003	1.003	1.0157	9578.9
				T2	1.004	1.009	1.004	1.0203	9586.3
5	2	9	27	T1	1.000	1.009	1.000	1.0233	276.1
				T2	1.000	1.000	1.000	1.0130	287.15

Note: 1. The ratio is (max $\bar{z}/\underline{z}$)/ (min $\bar{z}/\underline{z}$). T1=*DU* [1, 15], T2=*DU* [16, 30]
2. NoG: Number of Groups, NoJ: Number of Jobs, TJ: Total Jobs. T: Type
3. L1: $S_1/S_2 < 1.0$, L2: $S_1/S_2 = 1.0$, L3: $S_1/S_2 > 1.0$

References

1. Garey, M.D., Johnson, D.S., Sethi, R.: The complexity of flowshop and jobshop scheduling. Math. Oper. Res. 1(2), 117–129 (1976)
2. Schaller, J.E., Gupta, J.N.D., Vakharia, A.J.: Scheduling a flowline manufacturing cell with sequence dependent family setup times. Eur. J. Oper. Res. 125, 324–339 (2000)

3. Logendran, R., Salmasi, N., Sriskandarajah, C.: Two-machine group scheduling problems in discrete parts manufacturing with sequence-dependent setups. Comp. Oper. Res. 33, 158–180 (2006)
4. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of IEEE international conference on neural networks, Piscataway, NJ, pp. 1942–1948 (1995)
5. Maggu, P.L., Das, G.: On sequencing problem with transportation times of jobs. Pure A. Math. 12, 1–6 (1980)
6. Yang, D.L., Chern, M.S.: Two-machine flowshop group scheduling problem. Comp. Oper. Res. 27, 975–985 (1999)
7. Bean, J.C.: Genetic algorithms and random keys for sequencing and optimization. ORSA J. Comp. 6(2), 154–160 (1994)
8. Liu, B., Wang, L., Jin, Y.H.: An effective hybrid PSO-based algorithm for flow shop scheduling with limited buffers. Comp. Oper. Res. 35(9), 2791–2806 (2008)
9. Nasser, S., Rasaratnam, L., Mohammad, R.S.: Total flow time minimization in a flowshop sequence-dependent group scheduling problem. Comp. Oper. Res. 37(1), 199–212 (2010)

Improved Quantum Particle Swarm Optimization by Bloch Sphere

Yu Du[1], Haibin Duan[1,2], Renjie Liao[1], and Xihua Li[1]

[1] National Key Laboratory of Science and Technology on Holistic Control, School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, China
[2] Provincial Key Laboratory for Information Processing Technology, Suzhou University, Suzhou 215006, China
duyu1962@sina.com, hbduan@buaa.edu.cn, lrjconan@gmail.com, maillixihua@qq.com

Abstract. Quantum Particle Swarm Optimization (QPSO) is a global convergence guaranteed search method which introduces the Quantum theory into the basic Particle Swarm Optimization (PSO). QPSO performs better than normal PSO on several benchmark problems. However, QPSO's quantum bit(Qubit) is still in Hilbert space's unit circle with only one variable, so the quantum properties have been undermined to a large extent. In this paper, the Bloch Sphere encoding mechanism is adopted into QPSO, which can vividly describe the dynamic behavior of the quantum. In this way, the diversity of the swarm can be increased, and the local minima can be effectively avoided. The proposed algorithm, named Bloch QPSO (BQPSO), is tested with PID controller parameters optimization problem. Experimental results demonstrate that BQPSO has both stronger global search capability and faster convergence speed, and it is feasible and effective in solving some complex optimization problems.

Keywords: Quantum Particle Swarm Optimization (QPSO), Bloch Sphere, Bloch QPSO(BQPSO), global search.

1 Introduction

Particle Swarm Optimization (PSO)[1], is a population based stochastic optimization technique proposed by Kennedy and Eberhart. As an emerging intelligent technology, PSO proves to be comparable in performance with other evolutionary algorithms such as Simulated Annealing (SA) and Genetic Algorithm (GA)[2]-[4]. However, as demonstrated by F. Van Den Bergh[5], the particle in PSO is restricted to a finite sampling space for each iteration of the swarm. This restriction weakens the global search capability and its optimization efficiency and may lead to premature convergence.

To overcome the above shortcomings of the PSO, a Quantum Particle Swarm Optimization (QPSO)[6], which takes into account the global optimization capability and accelerated calculation characteristic of quantum computing, was proposed recently. However, QPSO uses qubits that are in the Hilbert space's unit circle with only one variable[7], therefore the quantum properties have been undermined to a large extent.

In order to improve the global search capability, a novel Bloch Sphere encoding mechanism was proposed in this paper.

2 Basic QPSO

In order to improve search ability and optimization efficiency and to avoid premature convergence for particle swarm optimization, a novel Quantum Particle Swarm Optimization for continuous space optimization is proposed. The positions of particles are encoded by the probability amplitudes of qubits, and the movements of particles are performed by quantum rotation gates, which achieve particles searching. The mutations of particles are performed by quantum non-gate to increase particles diversity. As each qubit contains two probability amplitudes, and each particle occupies two positions in space, therefore it accelerates the searching process.

Similar to PSO, QPSO is also a probabilistic search algorithm. A qubit position vector as a string of n qubits can be defined as follows:

$$P_i = \left[\begin{array}{c|c|c|c} \cos(\theta_{i1}) & \cos(\theta_{i2}) & \cdots & \cos(\theta_{in}) \\ \sin(\theta_{i1}) & \sin(\theta_{i2}) & \cdots & \sin(\theta_{in}) \end{array}\right]$$

where $\theta_{ij} = 2\pi \times rand$; $rand$ is the random number between 0 and 1, $i = 1,2,\cdots,m;\ j = 1,2,3,\cdots,n$, m is the population size and n is the space dimension.

This shows that each particle of the populations occupies the following two positions, corresponding to the probability amplitudes of state '0' and '1'.

$$P_{ic} = (\cos(\theta_{i1}),\cos(\theta_{i2}),\cdots\cos(\theta_{in})),\ P_{is} = (\sin(\theta_{i1}),\sin(\theta_{i2}),\cdots\sin(\theta_{in}))$$

In QPSO, the particle position movement is realized by quantum rotation gates. Therefore, particle velocity update in standard PSO is converted to qubit rotation angles update in quantum rotation gates, while particle movement update is converted to qubit probability amplitudes update.

Qubit rotation angle is updated as: $\Delta\theta_{ij}(t+1) = w\cdot\Delta\theta_{ij}(t) + c_1 r_1(\Delta\theta_l) + c_2 r_2(\Delta\theta_g)$

where $\Delta\theta_l$ is determined by individual optimal position and $\Delta\theta_g$ is determined by global optimal position. Reference[8]-[9] gives a query table to find out the right $\Delta\theta_l$ and $\Delta\theta_g$.

Directed by the current individual and global optimal position, the movements of particles are performed by quantum rotation gates $U(\theta)$, then $[\cos(\theta_{ij}(t)),\quad \sin(\theta_{ij}(t))]^T$ is updated as:

$$U(\theta)\bullet\begin{bmatrix}\cos(\theta_{ij}(t))\\ \sin(\theta_{ij}(t))\end{bmatrix} = \begin{bmatrix}\cos(\Delta\theta_{ij}(t+1)) & -\sin(\Delta\theta_{ij}(t+1))\\ \sin(\Delta\theta_{ij}(t+1)) & \cos(\Delta\theta_{ij}(t+1))\end{bmatrix}\bullet\begin{bmatrix}\cos(\theta_{ij}(t))\\ \sin(\theta_{ij}(t))\end{bmatrix} = \begin{bmatrix}\cos(\theta_{ij}(t)+\Delta\theta_{ij}(t+1))\\ \sin(\theta_{ij}(t)+\Delta\theta_{ij}(t+1))\end{bmatrix} \quad (1)$$

where $i = 1,2,\cdots,m;\ j = 1,2,3,\cdots,n;$ Then updated two new locations of particle P_i are:

$$\overline{P}_{ic} = (\cos(\theta_{i1}(t) + \Delta\theta_{i1}(t+1)),\cdots,\cos(\theta_{in}(t) + \Delta\theta_{in}(t+1)))$$

$$\overline{P}_{is} = (\sin(\theta_{i1}(t) + \Delta\theta_{i1}(t+1)),\cdots,\sin(\theta_{in}(t) + \Delta\theta_{in}(t+1)))$$

As each qubit contains two probability amplitudes, each particle occupies two positions in space, therefore it accelerates the searching process.

Mutation operator was proposed in the QPSO to help increase the particles diversity and global search capability. Quantum non-gate V is used here as a mutation operator. To randomly select a number of qubits based on pre-determined mutation probability and impose the quantum non-gate to interchange two probability amplitudes on the same bit. Such a mutation is in fact a kind of qubit rotation update: suppose a qubit has an angle t, after the mutation, the angle turns to be $\pi/2 - t$, which means it rotates forward at an angle of $\pi/2 - 2t$.

The specific mutation operation is as follows:

$$V \bullet \begin{bmatrix} \cos(\theta_{ij}(t)) \\ \sin(\theta_{ij}(t)) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \bullet \begin{bmatrix} \cos(\theta_{ij}(t)) \\ \sin(\theta_{ij}(t)) \end{bmatrix} = \begin{bmatrix} \sin(\theta_{ij}(t)) \\ \cos(\theta_{ij}(t)) \end{bmatrix} = \begin{bmatrix} \cos(\frac{\pi}{2} - \theta_{ij}(t)) \\ \sin(\frac{\pi}{2} - \theta_{ij}(t)) \end{bmatrix} \tag{2}$$

This kind of uniform forward rotation can increase the diversity of particles and reduce the premature convergence probability.

3 The Proposed Bloch QPSO with Bloch Sphere

As the evolvement method in QPSO is a probability operation, individuals will inevitably produce degradation phenomenon during population evolution. And the determination of rotation angle orientation by far is almost based on the look-up table[8]-[9], which involves multi-conditional judgments, thus reduces the algorithm efficiency. Therefore, we propose a novel Bloch Sphere encoding mechanism and take a simple angle orientation determination method to solve the above problems.

3.1 The Bloch Sphere Encoding Mechanism

In quantum computing, the smallest information units are quantum bits, or qubits. In three-dimensional Bloch Sphere, a qubit can be written in the form as: $|\varphi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle$,where $\left|\cos\frac{\theta}{2}\right|^2$ and $\left|e^{i\varphi}\sin\frac{\theta}{2}\right|^2$ respectively represent the probability of becoming $|0\rangle$ or $|1\rangle$, also they satisfy the normalization condition: $\left|\cos\frac{\theta}{2}\right|^2 + \left|e^{i\varphi}\sin\frac{\theta}{2}\right|^2 = 1$ .Therefore qubits can be expressed by using the quantum probability amplitudes as: $\left[\cos\frac{\theta}{2},\ e^{i\varphi}\sin\frac{\theta}{2}\right]^T$. In the Bloch sphere, a point P can be determined by two angles θ and φ, as it is shown in Figure 1 below.

Figure 1 tells us that every qubit corresponds to a point in the Bloch Sphere, thus we can directly use the Bloch Sphere coordinates to encode the particles positions. Suppose that P_i is the i th particle in the population. Then Bloch Sphere encoding process is described as follows:

$$P_i = \begin{vmatrix} \cos\varphi_{i1}\sin\theta_{i1} \\ \sin\varphi_{i1}\sin\theta_{i1} \\ \cos\theta_{i1} \end{vmatrix} \begin{matrix} \cdots \\ \cdots \\ \cdots \end{matrix} \begin{vmatrix} \cos\varphi_{in}\sin\theta_{in} \\ \sin\varphi_{in}\sin\theta_{in} \\ \cos\theta_{in} \end{vmatrix}$$

where $\varphi_{ij} = 2\pi \times rand$, $\theta_{ij} = \pi \times rand$, *rand* is the random number in $[0, 1]$, $i = 1,2,\cdots,m;$ $j = 1,2,3,\cdots,n;$ m is the population size and n is the space dimension. Then each qubit is encoded as follows: $P_{ix} = (\cos(\varphi_{i1})\sin(\theta_{i1}),\cdots,\cos(\varphi_{in})\sin(\theta_{in}))$, $P_{iy} = (\sin(\varphi_{i1})\sin(\theta_{i1}),\cdots,\sin(\varphi_{in})\sin(\theta_{in}))$, $P_{iz} = (\cos(\theta_{i1}),\cdots,\cos(\theta_{in}))$.

To facilitate the presentation, we defined P_{ix}, P_{iy}, P_{iz} as position X , Y, Z .

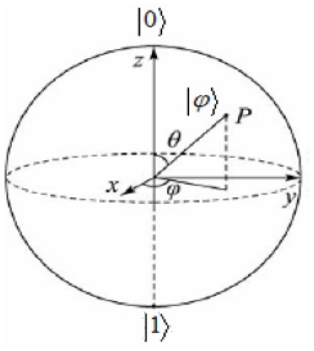


Fig. 1. Qubit Bloch Sphere

In the encoding mechanism above, each Bloch Sphere coordinate is treated as a particle position, thus each qubit contains three probability amplitudes and each particle occupies three positions in space. As a result, the Bloch QPSO has expanded solution numbers and enhanced the probability of obtaining global optimal solution.

3.2 Bloch Qubit Update and Mutation

Similar to QPSO, the particle position movement in BQPSO is realized by quantum rotation gates. Suppose that particle P_i gets current individual optimal position X , namely $P_{il} = (\cos(\varphi_{il1})\sin(\theta_{il1}),\cdots,\cos(\varphi_{il n})\sin(\theta_{il n}))$. While current global optimal position is $P_g = (\cos(\varphi_{g1})\sin(\theta_{g1}),\cdots,\cos(\varphi_{gn})\sin(\theta_{gn}))$.

Based on the above assumption, the Bloch Qubit probability amplitude is updated as follow. Derived from the matrix equation (3)[10], we get the rotation gate U:

$$U \bullet \begin{bmatrix} \cos\varphi_{ij}(t)\sin\theta_{ij}(t) \\ \sin\varphi_{ij}(t)\sin\theta_{ij}(t) \\ \cos\theta_{ij}(t) \end{bmatrix} = \begin{bmatrix} \cos(\varphi_{ij}(t)+\Delta\varphi_{ij}(t+1))\sin(\theta_{ij}(t)+\Delta\theta_{ij}(t+1)) \\ \sin(\varphi_{ij}(t)+\Delta\varphi_{ij}(t+1))\sin(\theta_{ij}(t)+\Delta\theta_{ij}(t+1)) \\ \cos(\theta_{ij}(t)+\Delta\theta_{ij}(t+1)) \end{bmatrix} \tag{3}$$

$$U = \begin{bmatrix} \cos\Delta\varphi_{ij}(t+1)\cos\Delta\theta_{ij}(t+1) & -\sin\Delta\varphi_{ij}(t+1)\cos\Delta\theta_{ij}(t+1) & \sin\Delta\theta_{ij}(t+1)\cos(\varphi_{ij}(t+1)+\Delta\varphi_{ij}(t+1)) \\ \sin\Delta\varphi_{ij}(t+1)\cos\Delta\theta_{ij}(t+1) & \cos\Delta\varphi_{ij}(t+1)\cos\Delta\theta_{ij}(t+1) & \sin\Delta\theta_{ij}(t+1)\sin(\varphi_{ij}(t+1)+\Delta\varphi_{ij}(t+1)) \\ -\sin\Delta\theta_{ij}(t+1) & -\tan(\varphi_{ij}(t+1)/2)\sin\Delta\theta_{ij}(t+1) & \cos\Delta\theta_{ij}(t+1) \end{bmatrix}$$

It is obvious that the qubit phase rotate $\Delta\varphi_{ij}(t+1)$ and $\Delta\theta_{ij}(t+1)$ respectively under the rotation gate action. $\Delta\varphi_{ij}(t+1)$ and $\Delta\theta_{ij}(t+1)$ are updated as follows:

$$\Delta\varphi_{ij}(t+1) = w\cdot\Delta\varphi_{ij}(t) + c_1 r_1(\Delta\varphi_l) + c_2 r_2(\Delta\varphi_g)$$
$$\Delta\theta_{ij}(t+1) = w\cdot\Delta\theta_{ij}(t) + c_1 r_1(\Delta\theta_l) + c_2 r_2(\Delta\theta_g)$$

Both $\Delta\varphi_{ij}(t+1)$ and $\Delta\theta_{ij}(t+1)$ are crucial to the convergence quality, in that the angle sign decides the convergence orientation while the angle size decides the convergence speed. From the update rule we know that $\Delta\varphi_{ij}(t+1)$ and $\Delta\theta_{ij}(t+1)$ are determined on $\Delta\varphi_l$, $\Delta\varphi_g$, $\Delta\theta_l$ and $\Delta\theta_g$. By far $\Delta\varphi_l$, $\Delta\varphi_g$, $\Delta\theta_l$ and $\Delta\theta_g$ are determined by a look-up table[8]-[9] containing all various possible conditions. However, as it involves multi-conditional judgments, this method reduces the efficiency of this algorithm to a large extent. To solve this problem, we use the following simple method to determine the related increment angle[10]:

$$\Delta\theta_l = \begin{cases} 2\pi + \theta_{ilj} - \theta_{ij} & (\theta_{ilj} - \theta_{ij} < -\pi) \\ \theta_{ilj} - \theta_{ij} & (-\pi \le \theta_{ilj} - \theta_{ij} < \pi) \\ \theta_{ilj} - \theta_{ij} - 2\pi & (\theta_{ilj} - \theta_{ij} > \pi) \end{cases} \qquad \Delta\theta_g = \begin{cases} 2\pi + \theta_{gj} - \theta_{ij} & (\theta_{gj} - \theta_{ij} < -\pi) \\ \theta_{gj} - \theta_{ij} & (-\pi \le \theta_{gj} - \theta_{ij} < \pi) \\ \theta_{gj} - \theta_{ij} - 2\pi & (\theta_{gj} - \theta_{ij} > \pi) \end{cases}$$

The calculation method for $\Delta\varphi_l$, $\Delta\varphi_g$ is the same as that for $\Delta\theta_l$, $\Delta\theta_g$.

To generalize the effect of quantum non-gate from the Hilbert space's unit circle to three-dimensional Bloch Sphere, we give a three-dimensional mutation operator, which satisfy the following matrix equation[10]:

$$V\cdot\begin{bmatrix} \cos\varphi_{ij}(t)\cos\theta_{ij}(t) \\ \sin\varphi_{ij}(t)\sin\theta_{ij}(t) \\ \cos\theta_{ij}(t) \end{bmatrix} = \begin{bmatrix} \cos(\pi/2 - \varphi_{ij}(t))\sin(\pi/2 - \theta_{ij}(t)) \\ \sin(\pi/2 - \varphi_{ij}(t))\sin(\pi/2 - \theta_{ij}(t)) \\ \cos(\pi/2 - \theta_{ij}(t)) \end{bmatrix} \tag{4}$$

Derived from (4), we get three-dimensional mutation operator V as follows:

$$V = \begin{bmatrix} 0 & \cot\theta_{ij}(t) & 0 \\ \cot\theta_{ij}(t) & 0 & 0 \\ 0 & 0 & \tan\theta_{ij}(t) \end{bmatrix}$$

Specific mutation process of BQPSO is similar to that of QPSO.

The process of our proposed BQPSO for solving complex optimization problems can be described as follows:

Step 1: Initialization of particle populations. Bring a random angle φ in $[0, 2\pi]$, a random angle θ in $[0, \pi]$, then qubits are produced by Bloch Sphere encoding. Initialize the size of angle increment $|\Delta\varphi| = \varphi_0$ and $|\Delta\theta| = \theta_0$. Set other parameters: max circulation generation- ger max ,mutation probability- Pm, population size- m, space dimension- n, and optimization problem solution scope: (a_j, b_j) $(j = 1,2,\cdots,n)$.

Step 2: Solution Space Transformation. Map three approximate positions for each particle form unit space $I^n = [-1,1]^n$ to optimization problem solution space Ω, then we get approximate solution $X(t)$.

Step 3: Evaluate fitness of each particle. If the particle's current position is superior to its own-memory optimal position, then replace the latter with current position. If the current global optimal position is superior to the optimal global position ever searched, then replace the latter with the current global optimal position.

Step 4: Use rotation gates to update the population according to formula (3).

Step 5:Use the three-dimensional mutation operator V under the mutation probability to mutate the quantum population according to formula (4).Then a new particle population was produced.

Step 6: Evaluate all the new fitness of each particle after the update operation and mutation operation. Update individual optimal position and global optimal position by the same methods that are used in Step 3.

Step 7: If the stopping criterion is satisfied, the proposed BQPSO algorithm stops, then output the best solution, else return to Step 3.

The above-mentioned procedures of the proposed BQPSO process can also be described in the Figure 2.

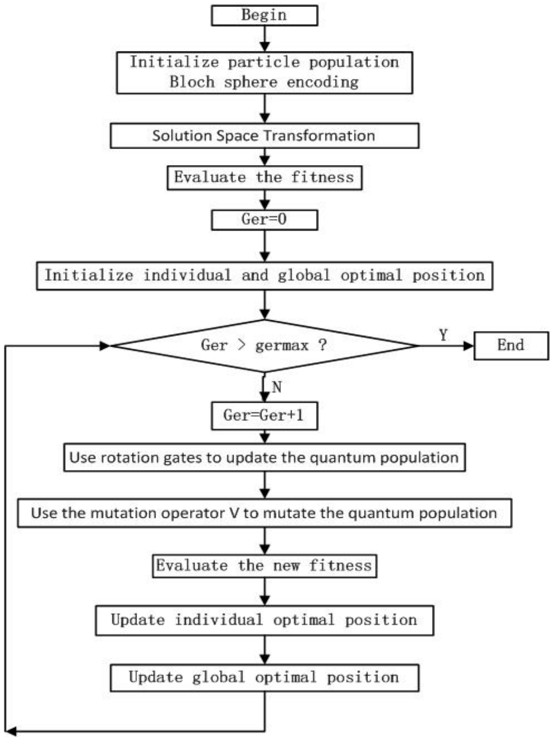

Begin
Initialize particle population
Bloch sphere encoding
Solution Space Transformation
Evaluate the fitness
Ger=0
Initialize individual and global optimal position
Ger > germax ?
Y
End
N
Ger=Ger+1
Use rotation gates to update the quantum population
Use the mutation operator V to mutate the quantum population
Evaluate the new fitness
Update individual optimal position
Update global optimal position

Fig. 2. The flowchart of the proposed BQPSO

4 Experimental Result

In order to investigate the feasibility and effectiveness of the proposed BQPSO, a series of experiments are conducted on a PID controller parameters optimization problem: to find the optimal parameters configuration (proportional coefficient, integral coefficient and derivative coefficient) for a second-order control system, of which the closed-loop transfer function is $G(s) = \frac{400}{s^2 + 50s}$.

In order to obtain satisfactory dynamic characteristics, we use the time integration of absolute error as the minimum objective function J, which has taken overshoot, adjustment time and static error altogether into account. The minimum objective function J is defined as follows: $J = \int_0^\infty (w_1|e(t)| + w_2 u^2(t) + w_4|\Delta y|)dt + w_3 \cdot t_u$ where $w_4 >> w_1$, $\Delta y(t) = y(t) - y(t-1)$, $e(t)$ is static error, $u(t)$ is PID controller output, t_u is adjustment time, $y(t)$ is the output of the controlled system, w_1, w_2, w_3, w_4 are weights. We take the fitness function as: $f = 1/J$. In the three conducted experiments, the first experiment uses standard PSO, the second experiment uses QPSO, while the third experiment uses the proposed BQPSO.

The three algorithms have been encoded and run the simulation in Matlab. Parameters were set to the following values: Sampling time is $1ms$, simulation time is $0.15s$, $ger\max$ =100, Pm =0.05, m=30, n=3, $w_1 = 0.999$, $w_2 = 0.001$, $w_3 = 2.0$, $w_4 = 100$, weighting factor $w = 0.5$, self-factor $c_1 = 2.0$, global-factor $c_2 = 2.0$. Draw the minimum objective function convergence curve and optimized system step response curve respectively on PSO, QPSO, and BQPSO. The results are shown in Figure 3 and Figure 4.

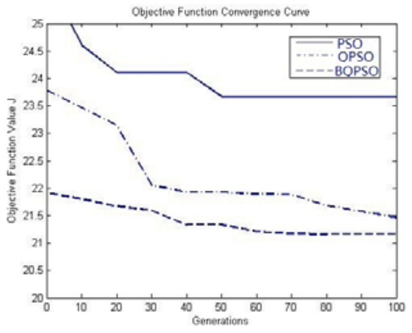


Fig. 3. Objective Function Convergence Curves

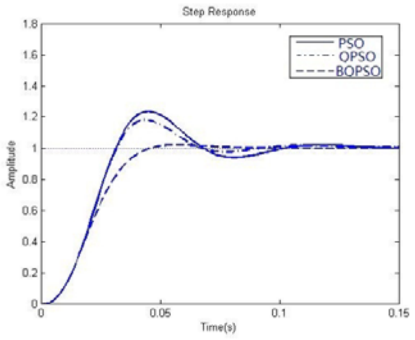


Fig. 4. Step Response

The Objective Function Convergence Curve in Figure 3 shows that, in the first experiment, it's easy to get into premature convergence with standard PSO. QPSO has more superior global search capability, while BQPSO performs best in the global search process. Also BQPSO has a faster convergence speed, compared with QPSO.

The Step Response in Figure 4 reflects that the control system optimized by BQPSO has smallest overshoot and transition time, while its stability, accuracy and rapidity of the system have been greatly improved, which indicates that BQPSO has strong robustness when applied to solve complex optimization problems.

It is obvious that our proposed BQPSO can find better solutions than standard PSO and QPSO in solving continuous optimization problems, for the reason that it has a more excellent performance with strong ability to find optimal solution and quick convergence speed.

5 Conclusions

This paper has presented an improved QPSO with Bloch Sphere for solving the continuous optimization problems. The serial experimental results verify that our proposed BQPSO is a practical and effective algorithm in solving some complex optimization problems, and also a feasible method for other complex real-world optimization problems.

Our future work will focus on applying the newly proposed BQPSO approach in this paper to other combinatorial optimization problems and combine it with other optimization methods. Furthermore, we will make a greater effort to give a complete theoretical analysis on the proposed BQPSO model.

Acknowledgments. This work was partially supported by the National Natural Science Foundation of China (Grant No. 60975072), the Program for New Century Excellent Talents in University of China (Grant No. NCET-10-0021) and the Beijing NOVA Program Foundation of China (Grant No. 2007A0017), the Aeronautical Science Foundation of China (Grant No. 2008ZC01006), and Open Fund of the Provincial Key Laboratory for Information Processing Technology, Suzhou University.

References

1. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: Proc. IEEE Conf. On Neural Network, pp. 1942–1948 (1995)
2. Angeline, P.J.: Evolutionary Optimization Versus Particle Swarm Optimization. In: Philosophy and Performance Differences. Evolutionary Programming VIII. LNCS, vol. 1477, pp. 601–610. Springer, Heidelberg (1998)
3. Eberhart, R.C., Shi, Y.H.: Comparison between Genetic Algorithm and Particle Swarm Optimization. In: Porto, V.W., Waagen, D. (eds.) EP 1998. LNCS, vol. 1447, pp. 611–616. Springer, Heidelberg (1998)
4. Krink, T., Vesterstorm, J., Riget, J.: Particle Swarm Optimization with Spatial Particle Extension. In: Proceedings of the IEEE Congress on Evolutionary Computation, vol. 2, pp. 1474–1479 (2002)
5. Van den Bergh, F.: An Analysis of Particle Swarm Optimizers. PhD Thesis. University of Pretoria, South Africa (2001)
6. Li, P.C., Li, S.Y.: Quantum particle swarms algorithm for continuous space optimization. Journal of Quantum Electronics 24(4), 463–468 (2007)

7. Al-Rabadi, A.N.: New dimensions in non-classical neural computing, part II: quantum, nano, and optical. International Journal of Intelligent Computing and Cybernetics 2(3), 513–573
8. Xing, Z., Duan, H.: An Improved Quantum Evolutionary Algorithm with 2 crossovers. In: Yu, W., He, H., Zhang, N. (eds.) ISNN 2009. LNCS, vol. 5551, pp. 735–744. Springer, Heidelberg (2009)
9. Eberhart, R.C., Shi, Y.H.: Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of the International Congress on Evolutionary Computation, Piscataway, pp. 84–88. IEEE Press, Los Alamitos (2000)
10. Li, P.C., Li, S.Y.: Quantum Computation and Quantum Optimization Algorithm, pp. 113–117. Harbin Institute of Technology Press (2009)

An Improved Particle Swarm Optimization for Permutation Flowshop Scheduling Problem with Total Flowtime Criterion

Xianpeng Wang and Lixin Tang

Liaoning Key Laboratory of Manufacturing System and Logistics, The Logistics Institute, Northeastern University, Shenyang, 110004, China
wangxianpeng@ise.neu.edu.cn, qhjytlx@mail.neu.edu.cn

Abstract. This paper deals with the m-machine permutation flowshop scheduling problem to minimize the total flowtime, an NP-complete problem, and proposes an improved particle swarm optimization (PSO) algorithm. To enhance the exploitation ability of PSO, a stochastic iterated local search is incorporated. To improve the exploration ability of PSO, a population update method is applied to replace non-promising particles. In addition, a solution pool that stores elite solutions found in the search history is adopted, and in the evolution process each particle learns from this solution pool besides its personal best solution and the global best solution so as to improve the learning capability of the particles. Experimental results on benchmark instances show that the proposed PSO algorithm is competitive with other metaheuristics.

Keywords: Permutation flowshop, particle swarm optimization.

1 Introduction

As one of the best known production scheduling problems, the permutation flowshop scheduling problem (PFSP) has always attracted considerable attentions of researchers due to its strong industrial background. In the PFSP, there is a set of jobs $J=\{1, 2, \ldots, n\}$ and a set of machines $M=\{1, 2, \ldots, m\}$. Each job $i\in J$ must be processed on these m ($m\geq 2$) machines in the same machine order of 1, 2, …, m. That is, the processing of each job should start from machine 1, then machine 2, and finish on machine m. The processing time of job $i\in J$ on machine $j\in M$ is denoted as p_{ij}, which is fixed and nonnegative. The jobs are available at time zero, and the processing of each job cannot be interrupted. At any time, each job can be processed on at most one machine, and each machine can process at most one job. A job cannot start to be processed on machine j until this job has been completed on machine j–1 and machine j is available. The objective of PFSP considered in this paper is to sequence these n jobs on m machines so that the total flowtime (TFT) can be minimized. Let $\pi=(\pi(1), \pi(2), \ldots, \pi(n))$ denote a job permutation (i.e. a job processing order), in which $\pi(k)$ represents the index of the job arranged at the k-th position of π, then the completion time of job $\pi(k)$ on each machine j can be calculated as follows: $C_{\pi(1),1} = p_{\pi(1),1}$; $C_{\pi(1),j} = C_{\pi(1),j-1} + p_{\pi(1),j}$, $j=2, 3, \ldots, m$; $C_{\pi(k),1} = C_{\pi(k-1),1} + p_{\pi(k),1}$, $k=2, 3, \ldots, n$; $C_{\pi(k),j} =$

max{$C_{\pi(k),j-1}$, $C_{\pi(k-1),j}$}+ $p_{\pi(k),j}$, k=2, 3, …, n; j=2, 3, …, m. Then the total flowtime can be defined as the sum of completion times of all jobs $TFT(\pi) = \sum_{k=1}^{n} C_{\pi(k),m}$.

The PFSP was first introduced by [1] and proven to be NP-complete in the strong sense even when m=2 ([2]), and since then it has obtained considerable attentions and many methods have been proposed. As reviewed by [3], these solution methods can be classified into three categories: exact methods ([4-5]), constructive methods ([6-11]), and metaheuristics ([12-15]). These methods generally use the benchmark problems of [16] to evaluate their performance.

In this paper, we propose an improved particle swarm optimization (PSO) for the PFSP. To enhance the exploration ability, a stochastic iterated local search is adopted. To improve the search diversification, a population update method is used to replace non-promising particles. In addition, a solution pool that stores elite solutions found in the search history is adopted to guide each particle's flight, besides its personal best solution and the global best solution. The rest of this paper is organized as follows. Section 2 describes the proposed PSO. Computational results on benchmarks are presented in Section 3. Finally, Section 4 concludes the paper.

2 Proposed PSO Algorithm

In the PSO algorithm introduced by [17-18], a swarm consists of m individuals (called *particles*) flying around in an n-dimensional search space. The position of the ith particle at the tth iteration is used to evaluate the particle and represent the candidate solution for the optimization problem. It can be represented as $X_i^t = [x_{i1}^t, x_{i2}^t, ..., x_{in}^t]$, where x_{ij}^t is position value of the ith particle with respect to the jth dimension (j=1, 2, …, n). During the search process, the position of a particle i is influenced by two factors: the best position visited by itself (p_{best}) denoted as $P_i^t = [p_{i1}^t, p_{i2}^t, ..., p_{in}^t]$, and the position of the best particle found so far in the swarm (g_{best}) denoted as $G^t = [g_1^t, g_2^t, ..., g_n^t]$. The new velocity ($V_i^t = [v_{i1}^t, v_{i2}^t, ..., v_{in}^t]$) and position of particle i at the next iteration are calculated according to:

$$v_{ij}^{t+1} = w \cdot v_{ij}^t + c_1 r_1 \cdot (p_{ij}^t - x_{ij}^t) + c_2 r_2 \cdot (g_j^t - x_{ij}^t) \tag{1}$$

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

where w is the *inertia* parameter, c_1 and c_2 are respectively *cognitive* and *social* learning parameter, and r_1, r_2 are random numbers in (0,1).

2.1 Solution Representation

In the PSO, we use each dimension to represent a job and consequently a particle $X_i^t = [x_{i1}^t, x_{i2}^t, ..., x_{in}^t]$ corresponds to the continuous position values for n jobs in the PFSP. Then the smallest position value (SPV) rule proposed by [14] is adopted to transform a particle with continuous position values into a job permutation. A simple example is provided in Table 1 to illustrate the SPV rule.

Table 1. Solution representation and the corresponding job permutation using SPV rule

Dimension j	1	2	3	4	5	6	7	8
x_{ij}^t	0.54	–0.75	–1.02	–0.41	0.92	–1.20	0.23	0.12
Job, π_i^t	6	3	2	4	8	7	1	5

2.2 Particle Learning Strategy

In the evolution process of standard PSO, each particle only learns the personal best it has found so far, and the global best the whole swarm has found so far. However, this stipulation may rigidly limit the learning capacity of particles. In the multi-objective PSO (MOPSO) proposed by [19], a so-called *external repository* was used to store the nondominated solutions found so far, and in the evolution process each particle also learns from a randomly selected solution from the *external repository*. Such a learning strategy results in a good improvement on the performance of the MOPSO. Therefore, based on the idea of the *external repository*, in our PSO algorithm a solution pool (denoted as R) is introduced to store the elite solutions found in the search history, and each particle also learn from an elite solution randomly selected from R. Using this strategy, the velocity update equation becomes:

$$v_{ij}^{t+1} = w \cdot v_{ij}^t + c_1 r_1 \cdot (p_{ij}^t - x_{ij}^t) + c_2 r_2 \cdot (g_j^t - x_{ij}^t) + c_3 r_3 \cdot (e_j^t - x_{ij}^t) \quad (3)$$

where c_3 is the learning parameter for the guiding solution e^t combined from a number of randomly selected elite solutions from R and r_3 is a random number in (0,1).

2.3 Update of the Elite Solution Pool

In our PSO, the solutions in the elite solution pool R is sequenced in the non-decreasing order of their objective value, that is, the first solution in R is the best one while the last solution in R is the worst one. At each iteration of our PSO, we use the newly updated personal best solutions to update the *elite solution pool R*. That is, if a newly updated personal best solution of a certain particle (i.e., x_i) is better than the worst one stored in R (i.e., $y_{|R|}$) and at the same time the minimum distance of x_i to the first $|R|$-1 solutions in R is not smaller than that of $y_{|R|}$, then this worst solution $y_{|R|}$ is replaced by the newly updated personal best solution x_i.

To define the distance between two solutions x_1 and x_2, we first use the SPV rule to transform them into two discrete job permutations $\pi_1=\{\pi_1(1), \pi_1(2), \ldots, \pi_1(n)\}$, and $\pi_2=\{\pi_2(1), \pi_2(2), \ldots, \pi_2(n)\}$, and then the distance between x_1 and x_2 can be calculated as $d(x_1, x_2) = \sum_{i=1}^{n} sign(|\pi_1(i) - \pi_2(i)|)$, in which $sign(s)$=1 if $s \neq 0$ (i.e, these two solutions have different jobs arranged in their i-th position); otherwise $sign(s)$=0. The minimum distance of x_i to the first $|R|$-1 solutions in R is the minimum one among $d(x_i, y_k)$ for k=1, 2, …, $|R|$-1.

We adopt this strategy because it can maintain the diversity of the elite solution pool R, which in turn will help to improve the search diversity of our PSO algorithm.

2.4 Population Initialization

The population with n_{pop} solutions is initialized by two kinds of heuristic procedures so that both the solution quality and the solution diversity can be considered. The first solution is obtained by a deterministic rule that first sequences jobs in the non-decreasing order of the total processing time of each job and then based on this sequence determines the final job permutation using the NEH method proposed by [20]. Since this solution is a discrete job permutation, we should convert it into a particle with continuous position values. For this job permutation, the position value of the job arranged in the jth position is calculated by $x_{\min} + j\times(x_{\max} - x_{\min}) / n$, where $x_{\min} = -1.0$ and $x_{\max} = 1.0$. The other solutions are generated are randomly generated according to: $x_{ij}^0 = x_{\min} + (x_{\max} - x_{\min})\times rand$, where *rand* denotes a random number uniformly distributed in [0, 1]. The initial velocities for the PSO particles are also generated by a randomly scheme that is similar to the position value formula: $v_{ij}^0 = v_{\min} + (v_{\max} - v_{\min})\times rand$, where $v_{\min} = -1.0$ and $v_{\max} = 1.0$.

2.5 Stochastic Iterated Local Search

The iterated local search (ILS) algorithm is a well known metaheuristics for NP-hard combinatorial optimization problems such as the TSP for its effectiveness and simplicity in practice ([21]). Each iteration of the ILS algorithm mainly consists of two steps: local search and *kick*. Given a starting solution s, the local search procedure first finds a local optimum solution s^* from s. If s^* can go through the acceptance criterion, then the *kick* procedure will generate a new intermediate solution s' from s^*. Subsequently s' will be taken as the new starting solution for the next iteration. The iteration of ILS repeats until the stop criterion is reached.

To accelerate the local search speed of the proposed ILS algorithm, we prefer to use a stochastic local search, which is based on the *insertion* neighborhood. For a given discrete job permutation π^t, let w and z denote two different random integer numbers generated in [1, n], and then the *insertion* neighborhood move used in the stochastic local search to generate a neighbor solution $\pi^{t\prime}$ is denoted as $\pi^{t\prime}$ =*insert*(π^t, w, z) that removes the job at the w-th position and then inserts it in the z-th position. The *kick* used in the proposed ILS algorithm consists of two steps and can be described as follows. First, randomly delete three jobs from the job permutation. Second, insert the three jobs into the job permutation at their best positions according to their deletion sequence. The detailed procedure of the proposed stochastic ILS algorithm is illustrated in Figure 1. Different from other metaheuristics such as the PSO of [14] that applied the local search on only the global best particle or solution at each iteration, we prefer to apply the stochastic ILS on both the global best particle and the best particle in the current population.

Different from the adjustment method used in [14], we do not adjust the position values whenever an *insertion* move is applied, but adjust the position values for one time. That is, the position values are adjusted only after the local optimum is obtained. For example, let π^t = (6, 3, 2, 4, 8, 7, 1, 5) and the obtained local optimum $\pi^{t\prime}$ = (6, 7, 2, 5, 4, 8, 3, 1). Then we just reassign the position values in the non-decreasing order to each job according to $\pi^{t\prime}$. Table 2 illustrates the position value adjustment method for this example.

```
Begin:
  Initialization:
    Let  0 be the input initial solution. Set $\pi = \pi_0$, the acceptance threshold $T = 0.05$, and
    the number of consecutive iterations that the best solution $\pi_0$ has not been improved
    to be $n_{counter} = 0$.
  while ($n_{counter} \le 2$) do
    1. Apply the kick procedure to $\pi$, and denote the obtained job permutation as
       $\pi' = kick(\pi)$.
    2. Set loop_counter=0, and then apply stochastic local search to $\pi'$.
         while ($loop\_counter < n \times (n-1)/2$)
           2.1 Generate a random number $r$ in [0, 1], and two random integer numbers
               $w$ and $z$.
           2.2 Generate $\pi'' = insert(\pi', w, z)$. If $f(\pi'') < f(\pi')$, then set $\pi' = \pi''$.
           2.3 Set $loop\_counter = loop\_counter + 1$.
      end while
    3. If $f(\pi') < f(\pi_0)$, set $\pi_0 = \pi'$, $\pi = \pi$, and $n_{counter} = 0$; otherwise set $n_{counter} = n_{counter} + 1$.
    4. If $f(\pi') \ge f(\pi_0)$ but $(f(\pi') - f(\pi_0)) / f(\pi_0) \le T$, set $\pi = \pi'$.
    5. Set $T = T \times 0.95$.
  end while
  Report the improved solution $\pi_0$.
End
```

Fig. 1. The main procedure of the proposed stochastic ILS algorithm

Table 2. Position value adjustment according to the obtained local optimum

Dimension j		1	2	3	4	5	6	7	8
Before ILS	x_{ij}^t	0.54	−0.75	−1.02	−0.41	0.92	−1.20	0.23	0.12
	Job, π_i^t	6	3	2	4	8	7	1	5
After ILS	x_{ij}^t	0.92	−0.75	0.54	0.12	−0.41	−1.20	−1.02	0.23
	Job, π_i^t	6	7	2	5	4	8	3	1

2.6 Population Update Method

As the evolution process of the PSO continues, some particles may always fly around low quality regions, or may be trapped in local optimum. To break through these obstacles for finding better solutions, a population update method is developed to improve the search diversification of the PSO. To determine the particles with low probability of finding better solutions, we track the consecutive iterations without improvement on the personal best p_{best} for each particle X_i (denoted as $NOPB_i$) and the global best g_{best} (denoted as *NOGB*). If $NOPB_i$ reaches a limit (*e.g.*, 10 consecutive iterations), then particle X_i will be reinitialized by combining three elite solutions that are randomly selected from the elite solution pool R. If *NOGB* reaches a limit (*e.g.*, 20 consecutive iterations), then the whole population will be reinitialized, but the solution pool R keeps unchanged.

The combination method for the three randomly selected solutions from R, i.e., $y_1=\{y_{11}, y_{12}, ..., y_{1n}\}$, $y_2=\{y_{21}, y_{22}, ..., y_{2n}\}$, $y_3=\{y_{31}, y_{32}, ..., y_{3n}\}$, can be described as follows: each position value of the new combined solution $X_i=\{x_{i1}, x_{i2}, \ldots, x_{in}\}$ is calculated as $x_{ij}=(y_{1j}+y_{2j}+y_{3j})/3$.

3 Computational Experiments

To test the performance of our PSO algorithm, computational experiments were carried out on the well-known standard benchmark set of [16] that is composed of 90 instances ranging from 20 jobs and 5 machines to 100 jobs and 20 machines. In this benchmark set there are 10 instances for each problem size. Our PSO algorithm was implemented using C++, and tested on a personal PC with Pentium IV 3.0 GHz CPU and 512 MB memory. To make a fairly comparison with the PSO$_{VNS}$, we use the same parameter setting proposed by [14]. That is, the population size is taken as $n_{pop}=2n$; the initial inertia weight is set to $w=0.9$ and never less than 0.4; the decrement factor for w is taken as 0.975; the acceleration coefficients are set to $c_1=c_2=2$; the maximum iteration number T_{max} is taken as 500. Furthermore, we set a maximum runtime limit for our PSO to be $m\times n\times 0.09$ seconds.

To evaluate the performance, our PSO algorithm (denoted as PSO*) was compared with other powerful methods, e.g. the constructive heuristics of [10], the ant colony algorithm of [13], and the PSO$_{VNS}$ of [14]. The solution quality was measured by the relative percent deviation (denoted as RPD) of the best solution found among R (R=5) replicated runs for each instance with respect to the best known results. That is, RPD is calculated as $RPD=\min\left\{\frac{(H_i-U_i)\times 100}{U_i}, i\in R\right\}$, in which H_i is the total flowtime value obtained by a certain algorithm whereas U_i is the best result obtained among the algorithms of [10] and [13] (this best result is denoted as LR&RZ).

Table 3. Performance comparison of PSO$_{VNS}$ and PSO* for total flowtime criterion

Problem	PSO$_{VNS}$		PSO*	
	RPD	CPU	RPD	CPU
20*5	**-0.175**	3.18	-0.17	9.03
20*10	-0.037	7.21	-0.04	18.05
20*20	2.758	11.93	**-0.07**	36.05
50*5	**-0.603**	41.71	-0.29	23.04
50*10	-0.819	74.49	**-0.90**	45.57
50*20	0.857	143.32	**-0.94**	90.70
100*5	**-0.570**	222.28	-0.05	60.35
100*10	**-0.692**	407.88	-0.38	96.71
100*20	-0.104	824.41	**-0.87**	187.15
Average	**0.068**	**192.93**	**-0.412**	**62.96**

Since the PSO_{VNS} algorithm proposed by [14] improved 57 out of 90 best known solutions reported by [10] and [13] for the total flowtime criterion, we compare our PSO^* algorithm with it. The comparison results are given in Table 3, in which the values are the average performance of the 10 instances for each problem size and the CPU is measured in seconds. From this table, it can be seen that our PSO^* algorithm performs much better than the PSO_{VNS} algorithm since the average RPD of our PSO^* algorithm is -0.412% while the average RPD of the PSO_{VNS} algorithm is 0.068%. Furthermore, our PSO^* algorithm can obtain better results for instances of 20×20, 50×20 and 100×20, which are more difficult to solve than other instances. Based on the assumption that our CPU with 3.0 GHz is 1.15 times faster than the CPU with 2.6 GHz in [14], the computation time of PSO algorithm can be treated as 62.96×1.15=72.44 seconds, which is much shorter than that of [14].

4 Conclusion

This paper proposes an improved PSO for the permutation flowshop problem with the total flowtime minimization. The search intensification of PSO is enhanced by incorporating a stochastic iterated local search, while the search diversification of PSO is improved by a population update method. Different from traditional PSO, a solution pool that stores elite solutions found in the search history is adopted to improve the learning ability of particles. Computational experiments on benchmark instances show that our algorithm outperforms the previous PSO algorithm.

Acknowledgements. This research is supported by National Natural Science Foundation for Distinguished Young Scholars of China (Grant No. 70425003), National Natural Science Foundation of China (Grant Nos. 60674084 and 70902065), and the Fundamental Research Funds for the Central Universities (Grant No. N090404018).

References

1. Johnson, S.M.: Optimal two- and three-stage production schedules with setup times included. Naval Research Logistics Quarterly 1, 61–68 (1954)
2. Framinan, J.M., Leisten, R., Ruiz-Usano, R.: Comparison of heuristic for flowtime minimisation in permutation flowshops. Computers & Operations Research 32, 1237–1254 (2005)
3. Garey, M.R., Johnson, D.S., Sethi, R.: The complexity of flowshop and jobshop scheduling. Mathematics of Operations Research 1, 117–129 (1976)
4. Bansal, S.P.: Minimizing the sum of completion times of n jobs over m machines in a flowshop – a branch and bound approach. AIIE Transactions 9, 306–311 (1977)
5. Chung, C.S., Flynn, J., Kirca, O.: A branch and bound algorithm to minimize the total flow time for m-machine permutation flowshop problems. International Journal of Production Economics 79, 185–196 (2002)
6. Gupta, J.N.D.: Heuristic algorithms for multistage flowshop scheduling problem. AIIE Transactions 4, 11–18 (1972)

7. Rajendran, C., Ziegler, H.: An efficient heuristic for scheduling in a flowshop to minimize total weighted flowtime of jobs. European Journal of Operational Research 103, 129–138 (1997)
8. Rajendran, C.: Heuristic algorithm for scheduling in a flowshop to minimize total lowtime. International Journal of Production Economics 29, 65–73 (1993)
9. Wang, C., Chu, C., Proth, J.M.: Heuristic approaches for $n/m/F/\sum C_i$ scheduling problems. European Journal of Operational Research 96, 636–644 (1997)
10. Liu, J.Y., Reeves, C.R.: Constructive and composite heuristic solutions to the $P//\sum C_i$ scheduling problem. European Journal of Operational Research 132, 439–542 (2001)
11. Framinan, J.M., Leisten, R.: An efficient constructive heuristic for flowtime minimisation in permutation flow shops. OMEGA 31, 311–317 (2003)
12. Yamada, T., Reeves, C.R.: Solving the C_{sum} permutation flowshop scheduling problem by genetic local search. In: Proceedings of IEEE international conference on evolutionary computation, pp. 230–234 (1998)
13. Rajendran, C., Ziegler, H.: Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs. European Journal of Operational Research 2004 155, 426–438 (2005)
14. Tasgetiren, M.F., Liang, Y.C., Sevkli, M., Gencyilmaz, G.: A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem. European Journal of Operational Research 177(3), 1930–1947 (2007)
15. Jarboui, B., Ibrahim, S., Siarry, P., Rebai, A.: A combinatorial particle swarm optimisation for solving permutation flowshop problems. Computers and Industrial Engineering 54, 526–538 (2008)
16. Taillard, E.: Benchmarks for basic scheduling problems. European Journal of Operational Research 64, 278–285 (1993)
17. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Network, pp. 1942–1948 (1995)
18. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the sixth international symposium on micro machine and human science, pp. 39–43 (1995)
19. Coello, C.A., Pulido, G.T., Lechuga, M.S.: Handling multiple objectives with particle swarm optimization. EEE Transactions on Evolutionary Computation 8(3), 256–279 (2004)
20. Nawaz, M., Enscore, E.E., Ham, I.: A heuristic algorithm for the m-machine, n-job sequencing problem. Omega 11, 91–95 (1983)
21. Congram, R.K.: Polynomially searchable exponential neighbourhoods for sequencing problems in combinatorial optimisation. PhD Thesis, University of Southampton (2000)

Broadband MVDR Beamformer Applying PSO

Liang Wang and Zhijie Song

Department of Ocean Technology, Ocean University of China,
238 Songling Road, Qingdao, 266100, P.R. China
hdwangliang@126.com

Abstract. In this paper, a broadband MVDR(minimum variance distortionless response) beamforming method based on time-domain (TMVDR) is presented. Using TMVDR beamformer, stable sample matrix estimation could be obtained in short time period. To obtain the stable optimum solution of TMVDR, a numerical searching method optimized by PSO algorithm with constrain condition is introduced. Out-sea experiment show the performance of TMVDR beamformer applying PSO algorithm proposed in this paper.

Keywords: DOA estimation, adaptive beamforming, PSO algorithm.

1 Introduction

The MVDR beamformer has superior performance on DOA(direction of arrival) estimation in low SNR (Signal to Noise Ratio) condition, and has been widely used in areas like sonar, radar and communication etc. In applying MVDR adaptive beamforming to passive detection of broadband signals two principal concerns arise. The first is snapshot deficient in the present of fast-moving targets. Broadband MVDR beamforming was usually implemented in frequency domain [1],[2], which firstly decomposes the time-domain data into certain sub-bands by Fourier Transform and then processes on each sub band. This method requires multiple snapshots in order to get stable correlation matrix estimation. Snapshot deficient causes distortion of beam pattern, loss of SINR, and related effects [3]. The second concern is unstable inverse when the eigenvalue dispersion of sample matrix is large since MVDR beamforming method currently adapt SMI (Sample Matrix Invert) algorithm [4]. That means tiny disturbance of sample matrix estimation will cause tremendous compute error when existing strong interference. Although diagonal loading [5] method could improve the robustness of the correlation matrix and obtain stable solution with fewer snapshots, but correspondingly the array gain and azimuth resolution ability decreases obviously because extra noise is introduced.

In this paper, we discussed a TMVDR (Time-domain MVDR) beamformer applying PSO (Particle Swarm Optimization) algorithm [6]. The array weights of TMVDR beamformer are designed to be a complex vector by treating two-channel orthogonal signals as one complex analytical signal, which minimize the power at its output while providing, at the same time, a fixed response toward the direction of arrival of the signal of interest. Different with frequency domain MVDR, TMVDR doesn't need block processing since it needn't FFT transform, thus we can obtain relative stable sample matrix estimation in short time period. To obtain the optimum solution of TMVDR, numerical search method optimized by PSO algorithm is proposed to replace the SMI algorithm. Numerical search algorithm can solve the problem of unstable inverse by searching a group of optimum solution in the weight vector space without diagonal loading. That is, we can always find a set of optimum weight vector to minimize the power of interference and noise by numerical searching method.

The paper is organized as follows. In Sec II, a TMVDR beamformer based on time-domain analytical signal is formatted. Sec III proposes a numerical searching method optimized by PSO algorithm to obtain the stable solution of TMVDR. Sec IV investigates the performance of TMVDR beamformer applying PSO algorithm via out-sea experiment. Sec V contains a brief synopsis of this paper and results of our work.

2 TMVDR Beamformer

Supposing the broadband passive signal is received by uniform line array composed of M elements, after delayed time of $\Delta i (1 \le i \le M)$, the signal vector received by array is

$$\boldsymbol{S} = [s_1, s_2, \cdots s_M]^T \tag{1}$$

The array output is:

$$\boldsymbol{X} = \boldsymbol{S} + \boldsymbol{N} \tag{2}$$

where $\boldsymbol{N} = [n_1, n_2, \cdots n_M]^T$ is the additive noise vector, T denotes transpose.

Eq. (2) is Hilbert transformed to

$$\overline{\boldsymbol{X}} = \overline{\boldsymbol{S}} + \overline{\boldsymbol{N}} \tag{3}$$

where $\overline{\boldsymbol{X}} = H(\boldsymbol{X})$, $\overline{\boldsymbol{S}} = H(\boldsymbol{S})$, $\overline{\boldsymbol{N}} = H(\boldsymbol{N})$.

Time domain analytical signal can be constructed as

$$\boldsymbol{Y} = \boldsymbol{Y}_s + \boldsymbol{Y}_n = \boldsymbol{X} + j \cdot \overline{\boldsymbol{X}} \tag{4}$$

where $\boldsymbol{Y}_s$ denotes the signal part of time-domain analytical signal , $\boldsymbol{Y}_n$ denotes the noise part .Thus the adaptive array output is:

$$C = \boldsymbol{W}^H \boldsymbol{Y} = \boldsymbol{W}^H \boldsymbol{Y}_s + \boldsymbol{W}^H \boldsymbol{Y}_n \tag{5}$$

where superscript H means transpose and conjugate , $\boldsymbol{W} = [w_1 \ w_2 \ \cdots \ w_M]^T$.

When the scanning direction is aimed to the incoming signal , there is

$$s_1 + j \cdot \overline{s}_1 = s_2 + j \cdot \overline{s}_2 \cdots = s_M + j \cdot \overline{s}_M = s + j \cdot \overline{s} \tag{6}$$

where s_i is the i_{th} interested signal received on array .Under the constrain condition

$$\sum_{i=1}^{M} w_i = 1 \tag{7}$$

Eq. (5) can be written as:

$$C = s + j \cdot \overline{s} + \boldsymbol{W}^H \boldsymbol{Y}_n \tag{8}$$

When the interested signal, interference and noise are cross uncorrelated, because the real part and the imaginary part of analytical signal are orthogonal, then the array output power comes to:

$$P = E\{CC^*\} = \boldsymbol{W}^H \boldsymbol{R} \boldsymbol{W} = s^2 + (\overline{s})^2 + \boldsymbol{W}^H \boldsymbol{R}_n \boldsymbol{W} \tag{9}$$

where

$$\boldsymbol{R} = E\{\boldsymbol{Y}\boldsymbol{Y}^H\} \tag{10}$$

$$\boldsymbol{R}_n = E\{\boldsymbol{Y}_n \boldsymbol{Y}_n^H\} \tag{11}$$

Hilbert transformation can't change the power of the signal of interest, making $(\overline{s})^2 = s^2$ a constant. From Eq.(9), a minimum $\boldsymbol{W}^H \boldsymbol{R}_n \boldsymbol{W}$ means a minimum total output power. Therefore, TMVDR can be expressed as a constrained optimum problem :

$$\begin{aligned} &\underbrace{\min}_{W} \{P\} \\ &s.t. \quad \sum_{i=1}^{M} w_i = 1 \end{aligned} \tag{12}$$

Using Eq. (10), exact sample matrix can be estimated in few snapshots period if the noise and interference is short time stable since TMVDR doesn't need split the data to blocks for Fourier transform.

3 Numerical Search Method

Using TMVDR beamformer, the total output power of array tends to be stable in short time period. The disturbance of sample matrix has little effect on output power. Thus we can obtain the optimum solution of Eq.(12) by numerical search algorithm. However, the computational complexity is huge by Numerical search algorithm than SMI algorithm. In this paper we adapt PSO algorithm to accelerate the convergence speed.

3.1 PSO Algorithm

Supposing a particle swarm composed of N particles, the position vector of the *i*th particle in the M dimension weights space is

$$\boldsymbol{Z}_i = [z_1 \ z_2 \ \cdots \ z_M]^T \tag{13}$$

and the moving velocity vector is

$$\boldsymbol{V}_i = [v_1\ v_2\ \cdots\ v_M]^T \tag{14}$$

Then the current position of each particle in the swarm is updated as follows:

$$\begin{cases} \boldsymbol{V}_i^{l+1} = a \cdot \boldsymbol{V}_i^l + c_1 \cdot rand(0,1) \cdot (\boldsymbol{p}_i - \boldsymbol{Z}_i^l) \\ \qquad\quad + c_2 \cdot rand(0,1) \cdot (\boldsymbol{p}_g - \boldsymbol{Z}_i^l) \\ \boldsymbol{Z}_i^{l+1} = \boldsymbol{Z}_i^l + \boldsymbol{V}_i^{l+1} \end{cases} \tag{15}$$

where a is the inertia weight, c_1 is the own experience weight and c_2 is the global experience weight, $rand(0,1)$ is uniform random number distributing in the range [0,1], $\boldsymbol{p}_i$ is the best previous position, $\boldsymbol{p}_g$ is the global best position. The iteration terminates when $\boldsymbol{p}_g$ remains invariant and then $\boldsymbol{p}_g$ is the global optimum solution.

PSO algorithm introduces own experience weight and companion's experience weight, making the particle at the local extreme value easily out of the local optimum position. Hence PSO algorithm is efficient for searching problem and has rapid convergence velocity.

3.2 Range of Search Space

Supposing the optimum solution of (12) is

$$\boldsymbol{W}_{opt} = [w_1, w_2 \cdots, w_M]^T = \boldsymbol{C} \circ \boldsymbol{D} \tag{16}$$

where

$$\boldsymbol{C} = [C_1, C_2, \cdots, C_M]^T \tag{17}$$

$$\boldsymbol{D} = [e^{j\varphi_1}, e^{j\varphi_2}, \cdots, e^{j\varphi_M}]^T \tag{18}$$

$\boldsymbol{C}$ stands for the mode vector of optimum weight , $\varphi_1, \varphi_2, \cdots, \varphi_M$ the phase of the optimum weight vector.

Defining $C_{\max} = \underbrace{\max}_{i=1,2,\cdots,M} [C_i]$, from Eq. (11), the minimum output power of array is

$$P_{\min} = \sum_{l=1}^{K} (\sum_{i=1}^{M} w_i x_{i_l})^2 \tag{19}$$

where K is the number of sampling time points. According to constrain condition $\sum_{i=1}^{M} w_i = 1$, make identity transformation to Eq.(19)

$$P_{\min} = \sum_{l=1}^{K} \left(\sum_{i=1}^{M} \frac{w_i x_{i_l}}{\sum_{i=1}^{M} w_i} \right)^2 = \sum_{l=1}^{K} \left[\sum_{i=1}^{M} \frac{w_i x_i / C_{\max}}{\sum_{l=1}^{M} w_i / C_{\max}} \right]^2 = \sum_{l=1}^{K} \left[\sum_{i=1}^{M} \frac{\overline{w}_i x_i}{\sum_{l=1}^{M} \overline{w}_i} \right]^2 \tag{20}$$

where

$$\overline{w}_i = w_i / C_{\max} \tag{21}$$

the solution of MVDR is unique under ideal conditions, Hence,

$$\overline{\boldsymbol{W}} = \boldsymbol{W}_{opt} \tag{22}$$

where, $\overline{\boldsymbol{W}} = [\overline{w}_1, \overline{w}_2, \cdots \overline{w}_M]^T$.

Form Eq.(20) and Eq.(21), the maximum mode of $\boldsymbol{W}_{opt}$ is 1. Therefore we can obtain the range of searching weight vector is

$C_i \in [0,1]$ 和 $D_i \in [0, 2\pi]$, $i = 1, 2, \cdots, M$.

3.3 Introduce of Constrain Condition

Supposing the complex weight vector of *i*th particle is $\boldsymbol{W} = [w_1, w_2, \cdots w_M]^T$. Constrain condition can be represented as

$$\begin{cases} \sum_{i=1}^{M} \operatorname{Re}(w_i) = 1 \\ \sum_{i=1}^{M} \operatorname{Im}(w_i) = 0 \end{cases} \tag{23}$$

making

$$a_i = \frac{\operatorname{Re}(w_i) + \frac{1}{M}}{\sum_{i=1}^{M} [\operatorname{Re}(w_i) + \frac{1}{M}]} \tag{24}$$

$$b_i = \operatorname{Im}(w_i) - \frac{1}{M} \sum_{i=1}^{M} \operatorname{Im}(w_i) \tag{25}$$

Where $\frac{1}{M}$ is introduced to ensure $a_i \le 1$. Constructing the vector as follows,

$$\boldsymbol{W}_c = [a_1 + jb_1, a_2 + jb_2, \cdots a_M + jb_M]^T \tag{26}$$

Eq. (26) is just the complex weight vector with constrain condition. Hence the steps of PSO algorithm can be described as follows:

1). Set the initial position vector of particle as Eq.(16),where $C_i = rand(0,1)$, $D_i = e^{j2\pi rand(0,1)}$.
2). Using Eq.(24)and (25), respectively process the real part and imaginary part of the initial position vector, and then obtain the complex weight vector as Eq. (26).
3). Calculate the objective function P in Eq.(9). Find the minimum objective function $P_{\min}$ and record the best previous position $\boldsymbol{p}_i$ and the best global position $\boldsymbol{p}_g$.
4). Update the position of particle according to Eq. (13).
5). The iteration terminates when $P_{\min}(n)$ (n is the iteration times) remains invariant($|P_{\min}(n+k) - P_{\min}(n)| < \varepsilon$) .The best global position $\boldsymbol{p}_g$ at this time is the optimum solution of TMVDR.

4 Result of Sea Experiment

4.1 Sea Experiment Description

In order to investigate the performance of TMVDR beamformer applying PSO algorithm, the data of out-sea experiment was processed. The array in experiment was uniform line array composed of 32 sensors; the interval between each unit is 1m; the band width is 300-700Hz, the total time period for data processing is 100 seconds, within which the target were near 40º, 70º, 90º, 140º ,150º and a strong interference source was around 0º~20º. Data sampling frequency was 6 KHz.

4.2 Signal Processing

The data was respectively processed by TMVDR method applying PSO algorithm and FMVDR (frequency domain MVDR) using diagonal loading algorithm.

FMVDR's covariance matrix estimation follow the next steps: Split the data into many blocks, each block has 512 sampling time points. There are 256 points overlap with two adjacent block, which is also the data length of each snapshot. After DFT, the cross-spectral density matrix was estimated on each frequency. The number of snapshots for each estimation is 64.That is ,the number of sampling time points for each processing is (64+1)x256=16640. Diagonal loading adapted the optimum factor[5].

TMVDR chooses 300-700Hz band width through band-pass filter, and only uses 1000 sampling time points to estimate the output power for each processing.

4.3 Comparison

Figure 1 displays energy level for 0°~180°azimuth angle v.s. time respectively processed by TMVDR and FMVDR. In Figure 1, as we have expected, TMVDR method has far better signal detection ability than FMVDR method. Tracks of sources are bright in the TMVDR result. Especially for the weak targets near 140°and 150°, FMVDR almost

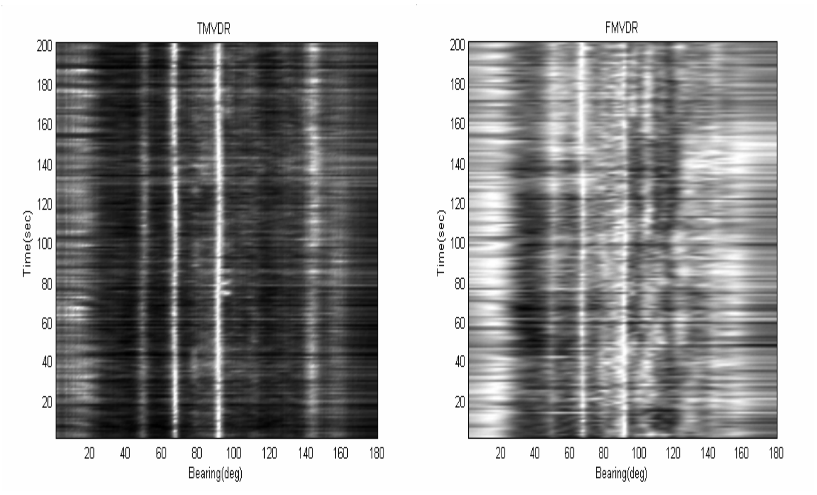


Fig. 1. Output of TMVDR method and FMVDR method

cannot detect the signal because the diagonal loading cause the degradation of array gain. In addition, the period of TMVDR for each processing is 1000/6000≈0.16s, far shorter than FMVDR , which period for each processing is 16640/6000≈2.77s.That makes TMVDR has better performance than FMVDR in signal detection for the fast moving targets.

5 Conclusion

A time domain broadband MVDR beamforming method is presented in this paper. On the basis of constructing time domain analytical signal, TMVDR introduces complex weights and could obtain stable sample matrix estimation in short time period. To obtain the stable optimum solution of TMVDR, a numerical searching method optimized by PSO algorithm with constrain conditions is proposed to replace the SMI algorithm. Using PSO algorithm, stable optimum weights vector can be searched with rapid convergence velocity.

The performance of TMVDR method applying PSO algorithm is put into test via out-sea experiment. Compared with FMVDR method applying diagonal loading algorithm, TMVDR method has better performance than FMVDR method for the problems of passive detection and azimuth estimation on broadband sound sources.

References

1. Kim, B.-C., Lu, I.-T.: High Resolution Broadband Beamforming Based on the MVDR Method. In: Proc. MTS/IEEE Oceans, vol. 3, pp. 1673–1676 (2000)
2. Yang, Y., Wan, C., Sun, C.: Broadband Beamspace DOA Estimation Algorithms. In: Proceeding of Oceans 2003, vol. 3, pp. 1654–1660 (2003)

3. Van Trees, H.L.: Optimum Array Processing. John Wiley & Sons, Inc., New York (2002)
4. Manolakis, D.G., Ingle, V.k., Kogon, S.M.: Statistical and Adaptive Signal Processing. Tsinghua University Press (2003)
5. Mestre, X., Lagunas, M.A.: Finite sample size effect on minimum variance beamformers: Optimum diagonal loading factor for large arrays. IEEE Transactions on Signal Processing 54, 69–82 (2006)
6. Li, B., Xiao, Y.: New Evolution Algorithm: Particle Swarm Optimization. Computer Science 30, 19–22 (2003)

Medical Image Registration Algorithm with Generalized Mutual Information and PSO-Powell Hybrid Algorithm

Jingzhou Zhang, Pengfei Huo, Jionghua Teng, Xue Wang, and Suhuan Wang

College of Automation, Northwestern Polytechnical University, Xi'an 710129, China

Abstract. The medical image registration algorithm uses the mutual information measure function that has many local extremes. Therefore, we propose our medical image registration algorithm that combines generalized mutual information with PSO-Powell hybrid algorithm and uses the objective measure function based on Renyi entropy. The Renyi entropy can remove the local extremes. We use the particle swarm optimization (PSO) algorithm to locate the measure function near the local extremes. Then we take the local extremes as initial point and use the Powell optimization algorithm to search for the global optimal solution. Section 2.2 of the paper presents the six-step procedure of our registration algorithm. We simulate medical image data with the registration algorithm; the simulation results, given in Table. 2 and 3, show preliminarily that the registration algorithm can eliminate the local extremes of objective measure function and accelerate the convergence rate, thus obtaining accurate and better registration results.

Keywords: Medical image registration, Generalized mutual information, Measure function, Optimization algorithm.

1 Introduction

Image registration is the process of overlaying two or more images of the same scene taken at different times, from different viewpoints, and/or by different sensors. It geometrically aligns two images—the reference and sensed images [1]. Ever since the concept of mutual information was introduced into the region of image registration has the validity of this new approach been widely accepted. Recently the registration based on mutual information has been widely used in occasions of image registration[2-4]. The registration algorithm based on maximization of mutual information(MMI) only uses statistical performance of gray values whereas neglects the image anatomical characteristics, so it is more robust and accuracy than traditional based on feature [5]. However, a few shortcomings still exist in the MMI registration algorithm, the objective measure function based on mutual information will lead to produce some local optimum in searching process, it would lead that the optimization process end in local optimum rather than global optimum [6].

Through the analysis of Renyi entropy [7], we found that mutual information based on Renyi entropy can not only remove unwanted local optimum but also has the depth of the basin of attraction. A new generalized mutual information measure function based on Renyi entropy has been given in this paper on the basis of the two characteristics of

Renyi entropy. At first, the PSO algorithm was used to find the local extreme of this measure function, which is use of the feature removing unwanted local optimum and smoothing out optimal curve. Then, the Powell optimization algorithm was used to locate the global optimal solution, which is use of the characteristic that has the depth of the basin of attraction, make the registration function easier to be optimized. The objective function can be located fast and accuracy through the mixed optimization algorithm and measure function.

The registration tests proven that this algorithm and measure function can overcome the local extreme of the mutual information measure function, make registration results up to a sub-pixel level, and also have better robust and accuracy.

2 Methods

2.1 Generalized Mutual Information Based on Renyi Entropy

According to Renyi entropy's definition, the Renyi entropy of an image is defined as:

$$R_q(X) = (1-q)^{-1}\ln(\sum_{i=1}^{N} P_i^q), q \in R, q \neq 1 \tag{1}$$

It is important that the Renyi entropy tends to the Shannon entropy as q→1. In analogy to Shannon entropy normalized mutual information, the generalized normalized mutual information of the two images based on Renyi entropy is given by

$$I(A,B) = \frac{R(A)+R(B)}{R(A,B)} \tag{2}$$

An image is selected as reference image. The floating image can be get by translating the reference image along x axis. The normalized mutual information of the two images can be computed, which is based on Renyi entropy and Shannon entropy, when Parameter q of the Renyi entropy is 2, 1.5, 1.25, 1.1, 0.9, 0.5, 0.25 respectively. The results are shown in Fig.1[7].

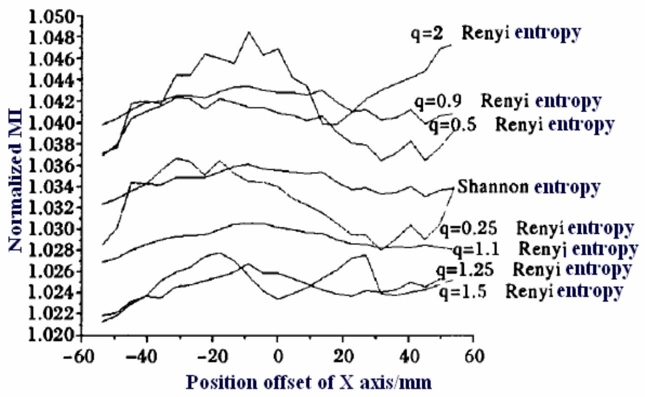


Fig. 1. They are the translation curves of normalized MI based on different entropy

According to Fig.1, Renyi entropy is more approximate to the Shannon entropy when q is more near one. The normalized mutual information curve based on Renyi entropy is most approximate to normalized mutual information curve based on Shannon entropy as q is 1.1. But at the same time the normalized mutual information curve based on Renyi entropy remove unwanted local optimum, smooth out optimal curve than curve based on Shannon entropy.

2.2 Hybrid Optimization Algorithm Based on PSO and Powell

PSO [8] is a stochastic population based optimization algorithm, firstly introduced by Kennedy and Eberhart in 1995. It is a global optimization algorithm. But in the end it is difficult to decide whether the solution we got is global optimum value in solution space, furthermore, whether it is located next to global optimization value. However, the Powell algorithm has a good performance in finding out local extreme. Thus the Powell algorithm could combine with the PSO Algorithm, and different optimization algorithms are used in different searching process. At first, the PSO global optimization algorithm was used to find the local extreme of generalized mutual information measure based on Renyi entropy as q is equal to 1.1, which is use of the feature that remove unwanted local optimum, smooth out optimal curve. Then, the Powell local optimization algorithm was used to locate the global optimal solution by searching the current local optimal extreme, which is use of the characteristic that has the depth of the basin of attraction as q is 0.99, make the registration function easier to be optimized. The PSO-Powell hybrid optimization algorithm in this paper solved searching the registration parameters process described as follows.

(1) Select a point T in three-dimensional solution space randomly, and Initialize a particle swarm in the solution space with the center of point. As a three-dimensional vector, the displacement of x, y and the rotational angle z.
(2) Do a certain number of iteration using PSO algorithm, then we can get the current local optimal solution T^1, when parameter q in objective measure function is equal to 1.1 in this process.
(3) Calculate the objective measure functional value MI and MI^1 in point T and T^1 respectively. If MI is less than MI^1, go to (4), otherwise go to (6).
(4) Taking T^1 as initial point and use Powell optimization algorithm to get the optimal point in the neighbor region, when parameter q in objective measure function is equal to 0.99 in this process.
(5) Initialized the Particle Swarm with the initial point $T=T^2$, then go to (2).
(6) Output the optimal solution and corresponding objective measure functional value as the optimal transformation parameters.

The initial strategy of PSO algorithm are improved in the searching process after the first search, in short the last searching result of the Powell optimization algorithm is as the current initial optimal point of PSO algorithm. Thus the algorithm performance would be improved because it not only makes full used of the previous calculated results but also decreases possibility falling into the local extreme.

3 Experimental Results

The generalized mutual information measure function and mixed optimization algorithm are used in single-modality and multi-modality image registration. Single-modality image registration is using three different medical images and three floating images which are got form the former after a certain degree of rotation and translation transformation. Multi-modality image registration is using two different modality medical images, at the same time the performances of PSO algorithm, Powell algorithm and mixed algorithm are compared in the same computer.

3.1 Single-Modality Medical Image Registration

Selecting two medical images as reference images, the first image is translated 9 pixels to the right along x axis, 4 pixels downward along y axis and rotated 7.5 degrees along counter-clockwise. The second image is translated 6 pixels to the right along x axis, 8 pixels downward along y axis and rotated 4.7 degrees along counter-clockwise. Then we can get two floating images. Using the two reference images and two floating images, single-modality image registration experiments are made with the algorithm in this paper. The results are shown in Table.1.

Table 1. They are the results of the single-modality medical image registration experiment. The *RMS* is Root-mean-square of the 50 registration results. The *mean* is average of the 50 registration results.

		x	y	z	△x	△y	△z	GMI	T
Image A	15th	9.0012	4.0051	7.4982	0.0012	0.0051	-0.0018	1.4450	74.810
	30th	9.0061	4.0031	7.4988	0.0061	0.0031	-0.0012	1.4450	100.386
	45th	9. 0109	3. 9972	7. 5012	0. 0109	-0. 0028	0. 0012	1. 4450	69. 277
	mean	9. 0040	4. 0009	7. 5008	0. 0040	0. 0009	0. 0008	1. 4450	103. 692
	RMS	9.0040	4.0009	7.5008	0.0000	0.0000	0.0000	1.4450	109.386
Image B	15th	6.0169	8.0006	4.6976	0.0169	0.0006	-0.0024	1.4212	146.793
	30th	6.0086	7.9986	4.7014	0.0086	-0.0014	0.0014	1.4213	139.386
	45th	6.0147	7.9892	4.7009	0.0147	-0.0108	0.0009	1.4212	146.776
	mean	6. 0170	8. 0039	4. 7026	0. 0170	0. 0039	0. 0026	1. 4212	110. 338
	RMS	6.0170	8.0039	4.7026	0.0001	0.0001	0.0000	1.4212	117.120

According to the table, in the registration results of image A the horizontal offset error of the 45th results between theoretical value and registration result is 0.0109, it is more than 0.01. And the other parameters selected are all less than 0.01. While in the registration results of image B it is only that the parameter errors of the 30th registration results are all less than 0.01, and the others are not all less than 0.01. So the registration results of image B are slightly inferior to image A. However, the single-modality image registration results are all a high accuracy.

At last, using the 15th registration results as registration parameters, and with bilinear transformation, image A is used for registration simulation. The Fig.2 could get form the simulation. The effect of the methods adopted in this paper is proved to be effective and better precision.

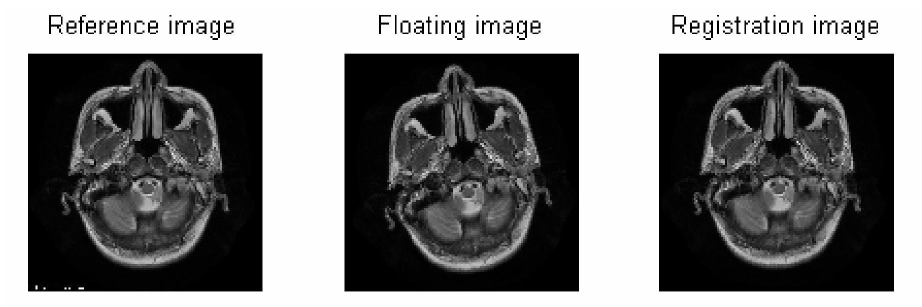


Fig. 2. It is the simulation experiment of the single-modality medical image registration

3.2 Multi-modality Medical Image Registration

We choose two groups of CT/MRI images. The two groups of CT/MRI images are used for multi-modality medical image registration, the registration optimization algorithm are Powell algorithm, PSO algorithm and PSO-Powell mixed algorithm

Respectively, and the objective measure function is adopted generalized mutual information measure function based on Renyi entropy. The results of multi-modality medical image registration are shown in Table.2.

According to the table, in the three algorithms the experimental time of the Powell algorithm and PSO algorithm are less than the hybrid optimization algorithm. While not only the average measure function values got by hybrid algorithm are the largest, but also that measure function values and registration parameters in every time got by hybrid algorithm are more concentrated in distribution than the other algorithm. General speaking, the registration results are up to a good precision by this hybrid algorithm and generalized measure function measure function in this paper.

Table 2. They are the results of the Multi-modality medical image registration experiment. The *mean* is average of the 50 registration results.

		First group images					Second group images				
		x	y	z	GMI	T	x	y	z	GMI	T
Powell method	15th	11. 302	-5. 695	4. 807	1. 1202	56. 124	31. 026	-11. 380	-7. 704	1. 0941	72. 812
	30th	10.332	-6.548	7.066	1.1205	44.017	34.250	-12.906	-2.980	1.0947	16.618
	45th	11.531	-5.888	3.575	1.1207	37.875	32.395	-13.228	-6.330	1.0944	35.333
	mean	10. 797	-6. 123	5. 701	1. 1206	34. 896	32. 306	-12. 646	-5. 715	1. 0943	37. 629
PSO method	15th	10. 614	-5. 320	6. 738	1. 1253	23. 189	33. 804	-11. 779	-4. 736	1. 0956	36. 313
	30th	8.904	-5.091	6.682	1.1251	49.801	33.791	-11.769	-4.750	1.0957	42.225
	45th	9.142	-5.471	4.472	1.1247	23.768	33.548	-11.482	-4.801	1.0954	31.145
	mean	9. 977	-5. 607	5. 632	1. 1248	25. 213	33. 041	-11. 874	-5. 863	1. 0953	32. 807
Powell +PSO method	15th	11. 253	-5. 618	4. 757	1. 1256	42. 168	33. 873	-11. 432	-4. 672	1. 0959	97. 020
	30th	11.136	-5.533	4.711	1.1257	131.71	33.815	-11.332	-4.741	1.0959	84.559
	45th	11.134	-5.672	4.709	1.1255	116.30	32.636	-11.145	-4.233	1.0958	68.079
	mean	11. 169	-5. 586	4. 708	1. 1256	101. 46	33. 574	-11. 394	-4. 652	1. 0958	65. 669

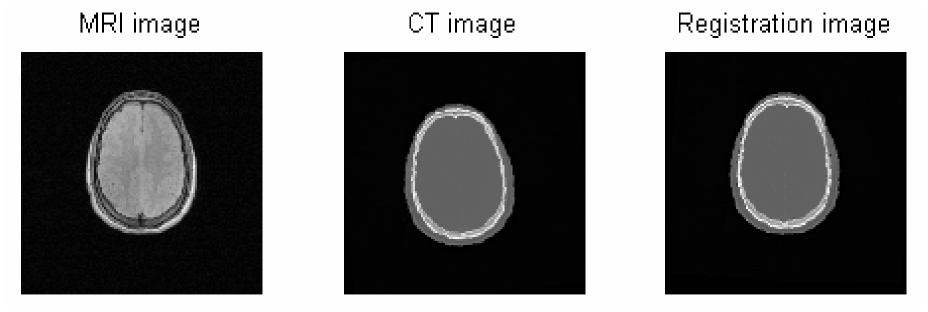


Fig. 3. It is the simulation experiment of the multi-modality medical image registration

At last, we choose one group of CT/MRI images to make registration experiment. Method used for registration is the method adopted in this paper. Then we can get the Fig. 3. The effect of the methods adopted in this paper is proved to be better.

4 Conclusions

Though registration results of MMI have the advantages of high accuracy and independence from any image pretreatments, there are still a few shortcomings. The registration images would exist better local matching, the interpolation algorithm would bring in errors. Thus objective measure function will produce a lot of local extreme, which has a large influence on optimization. The generalized mutual information measure function based on Renyi entropy and PSO-Powell mixed optimization algorithm have been adopted in this paper. At first, the PSO optimization algorithm was used to find the local extreme of generalized mutual information measure based on Renyi entropy. Then, the Powell optimization algorithm was used to locate the global optimal solution by searching the current local optimal extreme. The registration results have proven that this algorithm and measure function can make them up to a sub-pixel level, and also have better robust and accuracy.

References

1. Zitova, B., Flusser, J.: Image registration methods: A survey. J. Image and Vision Computing 21, 977–1000 (2003)
2. Ardizzone, E., Gambino, O., La Cascia, M., Lo Presti, L., Pirrone, R.: Multimodal non-rigid registration of medical images based on mutual information maximization. In: 14th IEEE International Conference on Image Analysis and Processing (2007)
3. Liu, Y., Fedorov, A., Kikinis, R., Chrisochoides, N.: Real-time Non-rigid Registration of Medical Images on a Cooperative Parallel Architecture. In: IEEE International Conference on Bioinformatics and Biomedicine (2009)
4. Andronache, A., von Siebenthal, M., Szekely, G., Cattin, P.: Non-rigid registration of multimodal images using both mutual information and cross-correlation. J. Medical Image Analysis 12(1), 3–15 (2008)

5. Yang, F., Zhang, H.: Multiresolution 3D Image Registration Using Hybrid Ant Colony Algorithm and Powell's Method. J. Journal of Electronics & Information Technology 3 29(3), 622–625 (2007)
6. Feng, L., Yan, L., Huang, D., He, M., Teng, H.: A Study of PSO and Powell Hybrid Algorithm Applied in Medical Image Registration. J. Beijing Biomedical Engineering 4 4(1), 8–12 (2005)
7. Zhang, H., Zhang, J., Sun, J.: Medical Image Registration Method Based on Mixed Mutual Information. J. Computer Applications 10 26(10), 2351–2353 (2006)
8. Chen, Y., Lin, C., Mimori, A.: Multimodal Medical Image Registration Using Particle Swarm Optimization. In: Eighth International Conference on Intelligent Systems Design and Applications, pp. 127–131 (2008)

Particle Swarm Optimization for Automatic Selection of Relevance Feedback Heuristics

Peng-Yeng Yin

Department of Information Management, National Chi-Nan University
Nantou 54561, Taiwan
pyyin@ncnu.edu.tw

Abstract. Relevance feedback (RF) is an iterative process which refines the retrievals by utilizing user's feedback marked on retrieved results. Recent research has focused on the optimization for RF heuristic selection. In this paper, we propose an automatic RF heuristic selection framework which automatically chooses the best RF heuristic for the given query. The proposed method performs two learning tasks: query optimization and heuristic-selection optimization. The particle swarm optimization (PSO) paradigm is applied to assist the learning tasks. Experimental results tested on a content-based retrieval system with a real-world image database reveal that the proposed method outperforms several existing RF approaches using different techniques. The convergence behavior of the proposed method is empirically analyzed.

Keywords: information retrieval, relevance feedback, long-term learning, heuristic selection, particle swarm optimization.

1 Introduction

Information retrieval techniques [1] have been intensively applied to many areas such as library dialog systems, database retrieval, web-based multimedia archiving, knowledge management, to name a few. The most popular abstraction for realizing the information retrieval tasks is the vector space model proposed by Salton in 1970's [2]. The vector space model represents each document by a set of numerical attributes (categorical attributes can be also used by defining an appropriate metric). Given a query document, the most relevant documents can be retrieved by selecting the closest documents to the query by reference to a distance norm such as the *Euclidean Distance* or *Cosine Distance*.

However, the retrieval precision obtained by using the vector space model may be unsatisfactory due to imperfect selection of attributes, noisy values, or null data. This problem has been regarded as the existence of semantic gap between the human perception model and the vector space model. A remedy to this problem is to treat the retrieval session as repetitive query reformulations. Through successive human-computer interactions, the query descriptive information (attributes, matching models, metrics or any meta-knowledge) is re-modified as a response to the user's relevance feedback (RF) on previously retrieved results. Therefore, a near-optimal query model is eventually produced and is able to attain high retrieval precision.

The major RF heuristics can be classified into three categories, namely, *query vector modification* (QVM) [3-4], *feature relevance estimation* (FRE) [5-6], and *classification-based* (CB) *methods* [7-9]. The QVM method reformulates the query vector through user's feedback so as to move the query toward relevant documents and away from irrelevant ones. The FRE approach learns the relevance weight for each attribute and incorporates the weights into the metric norm. As such, more relevant attributes have larger impacts on the retrieval results. While the CB method trains a classifier (by using Bayes theory, a support vector machine, or boosting technique) from the prior history of user's feedbacks for classifying the test data.

Although the RF heuristics can improve the precision of retrievals, each RF heuristic has its own bias and weakness. To maximize the precision improvement, recent approaches strive to find an optimal strategy that selects the best RF heuristic for a particular query. Minka and Picard [10] proposes the four-eye system which selects and combines groupings of the data, where groupings can be reduced by highly specialized attributes. Automatic selection are facilitated by analyzing relevant and irrelevant examples from the user. Yin *et al.* [11] presents a reinforcement learning method for integrating existing RF heuristics in one system. Various integration schemes are proposed to maximize the synergism and a long-term memory is used to exploit the meta-knowledge across multiple sessions. This research trend shares a synergy with another field named *hyperheuristic* [12]. Hyperheuristic can be thought of a heuristic for choosing heuristics. It works in the heuristics space instead of the solution space. In light of this, we propose in this paper a particle swarm optimization (PSO) method [13] for facilitating the automatic selection of multiple RF heuristics. The PSO paradigm is applied to perform two learning tasks, namely, the query reformulation and heuristic selection.

The remainder of this paper is organized as follows. Section 2 reviews the related RF heuristics and the particle swarm optimization. Section 3 describes the proposed method. Section 4 presents the experimental results and provides comparative performance evaluation. Finally, conclusions are given in Section 5.

2 Related Work

2.1 Vector Space RF Heuristics

Assume that there are *n* documents stored in a repository. Each document *D* is described by *r* attributes, $\vec{D} = (d_1, d_2, ..., d_r)$. Let *Q* be the query vector, denoted by $\vec{Q} = (q_1, q_2, ..., q_r)$. The *Euclidean distance*, $dist_{Euclidean}$, can be used for estimating the dissimilarity between *Q* and *D*. However, the closest documents according to Euclidean distance may not be considered as relevant by the user due to the semantic gap previously mentioned. The following RF heuristics can be applied to improve the user's satisfaction.

- *Query vector modification* (QVM). Let *R* and *N* denote the subsets of the retrieved result that are marked relevant and irrelevant, respectively, by the user in the incumbent feedback iteration. QVM reproduces a new query vector by the following formula.

$$Q \leftarrow \alpha Q + \beta \sum_{D_j \in R} \frac{D_j}{|R|} - \gamma \sum_{D_j \in N} \frac{D_j}{|N|} \tag{1}$$

where D_j is a document in R or N; α, β and γ are the relative weights contributed by the current query, relevance experience and irrelevance experience, respectively. The newly produced query vector is then used for searching the next retrievals. QVM has the effect of guiding the reformulation of the query towards relevant documents and away from irrelevant ones, and the moving velocity is accelerated by an inertia term considering previous query.

- *Feature relevance estimation* (FRE). The approach assumes that each attribute can have a different weight when judging the relevance between Q and D. A simple notion for estimating the relevance weight w_i of individual attribute is the attribute projection scheme that assesses the retrieval ability using each attribute alone. Finally, the relevance weights are incorporated into the dissimilarity metric to express the degree of emphasis on the corresponding attribute, *viz.*,

$$dist_{FRE} = \left\| \vec{Q} - \vec{D} \right\|_W = \sqrt{\left(\vec{Q} - \vec{D}\right) W \left(\vec{Q} - \vec{D}\right)^T} \tag{2}$$

where W is a diagonal attribute weight matrix whose diagonal entries are equal to w_i and off-diagonal entries are zero. Therefore, $dist_{Euclidean}$ can be viewed as a special case of $dist_{FRE}$ where W is equal to the identity matrix. Note that the relevance weights are query-dependent and are re-evaluated in each feedback iteration. These weights are used as constants within a given iteration when comparing the similarity between the query and each of the stored documents.

2.2 Particle Swarm Optimization

Particle swarm optimization (PSO) mimicking the swarming behavior of flocking birds and schooling fish was first introduced by Kennedy and Eberhart [13]. Ethologists observe that swarming birds/fish flock synchronously, change direction suddenly, scatter and regroup iteratively, and finally stagnate at a target. This form of social interactions increases the likelihood for finding food. The PSO algorithm emerges as a good option among others for solving nonlinear optimization.

One of the best forms of PSO is the Type 1 constriction factor model [14] which guarantees the convergence of the particles trajectories. To illustrate, the procedure is summarized in Fig. 1. A swarm of S particles are initialized at random where each particle is a vector of r parameter values for targeting the optimum solution. A velocity vector is also initialized at random for each particle to conduct the search trajectories. The swarm enriches its intelligence by storing the best solutions seen by every particle. In particular, particle i remembers the best position it visited so far, referred to as $pbest_i$, and the best position ($gbest$) observed by this particle's neighbors. The algorithm operates in an iterative manner. At each iteration, particle i adjusts its velocity v_{ij} and position p_{ij} through each dimension j by referring to $pbest_{ij}$ and $gbest_j$, while still explores new regions by random multipliers to escape from local optima.

```
1 Initialize.
    1.1 Generate S particle positions, $\vec{P}_i = (p_{i1}, p_{i2}, \ldots, p_{ir})$, $1 \le i \le S$, at random.
    1.2 Generate S velocity vectors, $\vec{V}_i = (v_{i1}, v_{i2}, \ldots, v_{ir})$, $1 \le i \le S$, at random.
2 Repeat until a stopping criterion is met.
    2.1 Evaluate the fitness of each particle $\vec{P}_i$.
    2.2 Determine the best vector $pbest_i$ visited so far by each particle $\vec{P}_i$.
    2.3 Determine the best vector $gbest$ observed so far by neighbors of particle $\vec{P}_i$.
    2.4 For each particle $\vec{P}_i$, update velocity vector $\vec{V}_i$ by
            $v_{ij} \leftarrow K(v_{ij} + c_1 rand_1(pbest_{ij} - p_{ij}) + c_2 rand_2(gbest_j - p_{ij}))$, $\forall j = 1, \ldots, r$
            $K = 2 \Big/ \left| 2 - (c_1 + c_2) - \sqrt{(c_1 + c_2)^2 - 4(c_1 + c_2)} \right|$ subject to $c_1 + c_2 > 4$
    2.5 For each particle $\vec{P}_i$, update particle's position by
            $p_{ij} \leftarrow p_{ij} + v_{ij}$, $\forall j = 1, \ldots, r$
```

Fig. 1. Summary of the PSO algorithm

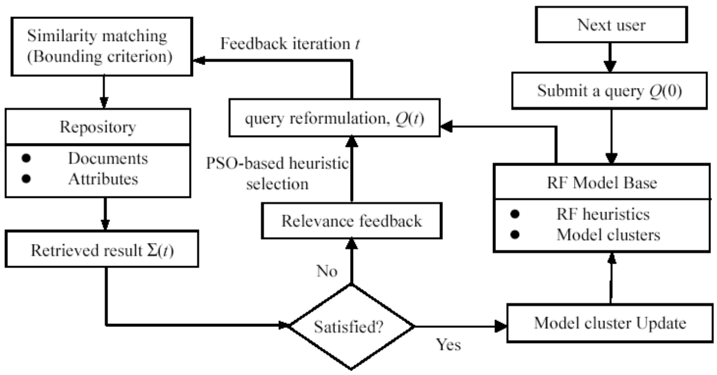


Fig. 2. Conception of the system diagram

3 Proposed Method

Fig. 2 depicts the diagram of the proposed information retrieval system. The user starts a new query session by submitting a query attribute vector $Q(0)$, the number in the parenthesis indicates the number of experienced iterations. $Q(0)$ is used to retrieve a model from the RF model base. The model cluster whose centroid is the closest to $Q(0)$ is selected and the corresponding RF heuristic is returned for conducting query reformulation. Then the reformulated query is matched to each of the stored documents using the selected RF heuristic and the top ranked documents $\Sigma(t)$ are retrieved. If the user is not satisfied with the result, he/she can start another round of RF by labeling at least one retrieved document as either relevant or irrelevant. Otherwise, the best RF heuristic is employed to update the model base. In contrast to the traditional RF process which executes a single RF heuristic, our system selects adaptively the best RF heuristic for the given query to achieve the maximum retrieval precision.

PSO is thus employed to optimize the heuristic selection by reference to the user's relevance feedback as the performance index. The principal components of our system are elaborated in the following subsections.

3.1 RF Model Base

The RF model base comprises multiple RF heuristics learned by PSO. For illustration, our model base is implemented with various combinations of QVM and FRE; however, the system framework given in Fig. 2 does not invite this limitation. In this context, each RF heuristic is determined by QVM weights (α, β, γ) and FRE weights (w_i). The RF model base is initialized with m RF heuristics by taking random values for (α, β, γ; w_i, $i = 1, 2, \ldots, r$).

One extreme implementation of this method is to create a separate heuristic for each query and then optimize this heuristic according to the subsequent feedbacks (obviously, the traditional fashion using a single RF heuristic for all queries lies on the other extremity). However, this would require the establishment of a huge model base that accounts for any given query. As an alternative, the queries can be partitioned into clusters in the attribute space, and those queries belonging to the same cluster would be processed using the same RF heuristic. The determination of the number of clusters is a tradeoff between retrieval precision and computational efficiency. We have found empirically that the precision improvement is negligible when the number of clusters (m) exceeds 10% of the number of documents (M).

3.2 Query Reformulation

Now we describe the optimization of RF heuristic selection and the query reformulation (see the left loop in Fig. 2). Let us denote the query and the RF heuristic derived at feedback iteration t by $Q(t)$ and ($\alpha(t)$, $\beta(t)$, $\gamma(t)$; $w_i(t)$, $i = 1, 2, \ldots, r$). We describe the processes for $t = 0$ and $t > 0$, separately, as follows. For the iteration with $t = 0$, the initial query $Q(0)$ is reformulated according to the retrieved RF heuristic and is then used for similarity matching. The similarity matching is conducted according to Eq. (2) and the list $\Sigma(0)$ of the top v most similar documents is returned for the user's inspection. If the user is satisfied with $\Sigma(0)$, the session is terminated without performing the model cluster update. Otherwise, the user activates a new RF iteration ($t = t + 1$) by marking relevant and irrelevant images in $\Sigma(0)$.

Now we describe the operations performed in the feedback iterations with $t > 0$. The PSO optimization mechanism is initialized when $t = 1$, and a swarm of S particles are created. Instead of creating the particles at random, the particles are initialized by perturbing the RF heuristic ($\alpha(0)$, $\beta(0)$, $\gamma(0)$, $w_1(0)$, $w_2(0)$, …, $w_r(0)$) to expedite the optimizing speed. Let us denote the initial swarm by ($\alpha^i(0)$, $\beta^i(0)$, $\gamma^i(0)$, $w_1^i(0)$, $w_2^i(0)$, …, $w_r^i(0)$), $i = 1, 2, \ldots, S$. These particles are then modified using the particle movement formulas, and the new positions of the S particles are obtained. At the query reformulation stage, S queries are produced (i.e., the query $Q(0)$ is expanded to S queries for $t > 0$). The ith query $Q^i(t)$ is reformulated using the following equation,

$$Q^i(t) \leftarrow \alpha^i(t)Q^i(t-1) + \beta^i(t) \sum_{D_j \in R^i(t-1)} \frac{D_j}{\left|R^i(t-1)\right|} - \gamma^i(t) \sum_{D_j \in N^i(t-1)} \frac{D_j}{\left|N^i(t-1)\right|}, \quad i = 1, 2, \ldots, S, \tag{3}$$

where $R^i(t\text{-}1)$ and $N^i(t\text{-}1)$ denote the sets of relevant and irrelevant documents, respectively, from the retrieved result $\Sigma^i(t\text{-}1)$ in the (t-1)th feedback iteration.

The swarm of particles $Q^i(t)$, $i = 1, 2, \ldots, S$, compete to generate $\Sigma(t)$ of the top v most similar documents. The list $\Sigma(t)$ is returned for display and requests the user for another round of relevance feedback. To monitor the retrieval performance using the RF heuristic carried by a given particle, for each member in $\Sigma(t)$, the source particle(s) that generates this member should be tallied. Hence, we have

$$\Sigma(t) = \bigcup_{i=1,\ldots,S} \Sigma^i(t), \tag{4}$$

where $\Sigma^i(t)$ consists of those members in $\Sigma(t)$ that are retrieved by $Q^i(t)$.

By examining the feedback information in $\Sigma^i(t)$, we are able to perform two important learning tasks. (1) *RF heuristic selection optimization*: The *fitness* of the ith particle can be evaluated by counting the number of members in $\Sigma^i(t)$ that are marked as relevant by the user. In particular, the fitness of particle i is defined as the *precision rate* (PR) of $\Sigma^i(t)$ which is calculated by PR = $\left|R^i(t)\right| / \left|\Sigma^i(t)\right|$. As such, the historical best particles, *pbest* and *gbest*, can be determined by reference to fitness values and they are then employed to guide the optimization for the RF heuristic selection. (2) *Query optimization*: With $R^i(t)$ and $N^i(t)$, the relevant and irrelevant documents from $\Sigma^i(t)$, we are able to reformulate (optimize) the next query $Q^i(t+1)$.

If the user is satisfied with $\Sigma^i(t)$ and decides to terminate the RF process, the *gbest* particle is responsible for model base update. Let Q^* be the query vector reformulated by the RF heuristic contained in *gbest*. The Euclidean distance between Q^* and each of the cluster centroids in the model base is computed, and the model base entry with the minimum Euclidean distance is replaced by a linear combination of the old entry values and the optimal query Q^* and the RF heuristic (α^*, β^*, γ^*, w_1^*, w_2^*, …, w_r^*).

4 Experimental Results

To illustrate the feasibility of the proposed method, we chose the application for content-based image retrieval which advocates *active queries* that retrieve relevant images by visual information attributes (color, shape, texture, etc.) The existing RF heuristics that we have implemented for comparison include QVM [4], FRE [6], SVM-active [8], and Log-based [15]. A large real-world image repository (the UCR database) is used for simulations.

UCR database is a real-world image repository containing 10,038 images. The collection of these images is classified into 56 topics including mountains, oceans, forests, buildings, cars, humans, animals, flowers, and others. In our experiments on performance evaluation, the images from the same topic are considered relevant while the images from different topics are deemed irrelevant. Each image in the database is described by 22 visual attributes, namely, 16 Gabor attributes (mean and standard deviation of filtered images at 4 orientation and 2 scales) and 6 color attributes (mean and standard deviation from the HSV color domain).

We compare the performance of the proposed PSO-based automatic heuristic selection method with that of the existing RF heuristics. Fig. 3(a) shows the average precision rate performance, by presenting each database image as a query, with the number of feedback iterations (*t*) obtained using the competing methods. Among all the methods, QVM and FRE seem to perform worst by improving the precision rate with 10% after consuming nine iterations of feedback. The SVM-active method enhances the retrieval performance for about 20% for the same number of feedback iterations, but the major improvement is observed after the third iteration of feedback. It is noteworthy that none of the three methods exploits long-term information through multiple query sessions, so the retrieval performance is significantly surpassed by the Log-based and the PSO-based methods.

The performance of the PSO-based and Log-based methods is comparable after the third iteration of feedback, but the PSO-based method clearly outperforms the Log-based technique in the first display and those in the first two feedback iterations. In fact, the Log-based method accumulates incrementally the users' feedback information in a log and utilizes them as the training samples, so the retrieval performance starts improving from the first feedback iteration, but not in the first display. By contrast, the proposed PSO-based method accesses the RF model base and applies the optimal RF heuristic for the given query to determine the first display, thus yielding significant improvement in the precision rate at the beginning.

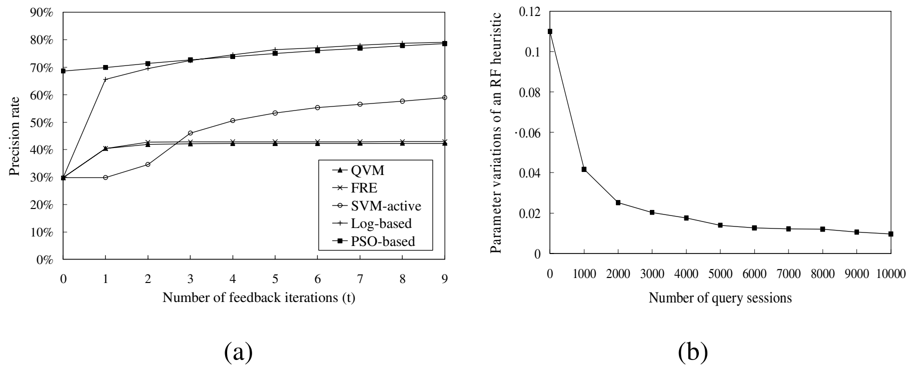


Fig. 3. Performance evaluation

It is desirable to know whether the proposed method exhibits a convergence behavior as the system experiences more query sessions. The convergence can be analyzed by monitoring the particles' velocities. As previously noted, the particle encodes RF heuristics by (α, β, γ; w_i, i = 1, 2, …, r) in the context of our learning task. We measure the difference of an RF heuristic when it is learned at successive query sessions to note the momentum of particles' velocities. Fig. 3(b) shows the average parameter variations of an RF heuristic during the evolution. The proposed method gradually learns the target RF heuristic for each of the clusters stored in the model base because the parameter variations do not change a lot after 6000 query sessions. From the PSO's point of view, the move velocities of particles are decreased as the number of query sessions increases, manifesting a trajectory convergence behavior.

5 Conclusions

Our survey on previous relevance feedback (RF) research discloses an emerging trend that the approaches based on heuristic selection optimization have prevailing advantages compared to other works. In this paper, we have proposed an automatic RF heuristic selection framework which conducts two optimization tasks, the query optimization and heuristic-selection optimization. The optimization task is accomplished using the PSO technique. Experimental results on the UCR database manifest that the proposed method outperforms several existing RF methods.

References

1. Singhal, A.: Modern Information Retrieval: A Brief Overview. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering 24(4), 35–43 (2001)
2. Salton, G.: The SMART System. Prentice Hall, Englewood Cliffs (1971)
3. Rocchio, J.J.: Relevance feedback in information retrieval. In: Salton, G. (ed.) The SMART System, pp. 313–323. Prentice-Hall, New Jersey (1971)
4. Ciocca, G., Schettini, R.: A relevance feedback mechanism for content-based image retrieval. Information Processing and Management 35(6), 605–632 (1999)
5. Rui, Y., et al.: Relevance feedback: a power tool for interactive content-based image retrieval. IEEE Trans. Circuits and Systems for Video Technology 8(5), 644–655 (1998)
6. Peng, J., Bhanu, B., Qing, S.: Probabilistic feature relevance learning for content-based image retrieval. Computer Vision and Image Understanding 75(1-2), 150–164 (1999)
7. Cox, I., et al.: The Bayesian image retrieval system, PicHunter: theory, implementation, and psychophysical experiments. IEEE Trans. Image Processing 9(1), 20–37 (2000)
8. Tong, S., Chang, E.Y.: Support vector machine active learning for image retrieval. In: Proceeding of the ACM International Conference on Multimedia, pp. 107–118 (2001)
9. Tieu, K., Viola, P.: Boosting image retrieval. In: Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 228–235 (2000)
10. Minka, T.P., Picard, R.W.: Interactive learning with a society of models. Pattern Recognition 30, 565–581 (1997)
11. Yin, P.Y., Bhanu, B., Chang, K.C., Dong, A.: Integrating relevance feedback techniques for image retrieval using reinforcement learning. IEEE Trans. Pattern Analysis & Machine Intelligence 27, 1536–1551 (2005)
12. Cowling, P., Soubeiga, E.: Neighbor structures for personnel scheduling: a summit meeting scheduling problem. In: Burke, E., Erben, W. (eds.) PATAT 2000. LNCS, vol. 2079. Springer, Heidelberg (2001)
13. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings IEEE Int'l. Conf. on Neural Networks, vol. IV, pp. 1942–1948 (1995)
14. Clerc, M., Kennedy, J.: The particle swarm explosion, stability, and convergence in a multidimensional complex space. IEEE Transaction on Evolutionary Computation 6, 58–73 (2002)
15. Hoi, C.H., Lyu, M.R.: A novel log-based relevance feedback technique in content-based image retrieval. In: Proceedings of the ACM International Conference on Multimedia, pp. 24–31 (2004)

Performance of Optimized Fuzzy Edge Detectors using Particle Swarm Algorithm

Noor Elaiza Abdul Khalid and Mazani Manaf

Faculty of Computer and Mathematical Sciences
Universiti Teknologi MARA
Shah Alam, Selangor, Malaysia
elaiza@tmsk.uitm.edu.my

Abstract. The purpose of the paper is to compare the performance of various fuzzy edge detectors which have been optimized by Particle Swarm Optimization (PSO). Three different type edge detectors Classical fuzzy Heuristic (CFH), Gaussian rule based (GRBF) and Robust Fuzzy Complement (RFC) are used. These edge detectors are effective in detecting edges, however the edges are thick. This paper proposes the used of particle swarm optimization algorithm as a method of producing thin and measurable edges. The fuzzy edge detectors are used in the initial swarm population and the objective function. The performance is based on the consistency of the visual appearance, fuzzy membership threshold and the number of complete edges detected. All three optimized edge detector performs reasonably well but CFHPSO outperform the rest.

Keywords: Fuzzy edge detection, Particle swarm optimization, Stochastic, Hybrid.

1 Introduction

Medical images are difficult to segment due to grainy image regions and complexity of the anatomical features [1]. This is due to gray level inconsistency and the lack of strong edges at its border. Image segmentation techniques can be categorized such edge-based methods, region-based methods and mixed contour-region-based methods [2]. Most edge and boundary detection method encounter problem such as broken contour due to noise or blurring [3], [4].

Fuzzy techniques in edge detection have becoming more imperative in digital image processing. Tizhoosh [5] introduced heuristic memberships function, simple fuzzy rules and fuzzy complements in his edge detector system.

PSO is a stochastic optimization population-based evolutionary computation technique, inspired by the social behavior of birds [6]. This paper compares the performance of the fusion of Particle swarm optimization with three types of fuzzy edge detectors. The fuzzy edge detection method are incorporated into the particle swarm algorithm in two stages; the initial population and as the objective function. This algorithm is tested on hand radiographs images [7].

2 Approach and Methods

The main challenge is to detect the outer cortical (OC) the strong edges and the inner cortical (IC) weak edges. Radiographs used in this research are retrospective images of non-dominant hand radiographs from the Hospital University Sains Malaysia (HUSM) [8]. The methodology consists of three phases which are the preprocessing phase where smoothing filters are used to reduce noise, followed by the fuzzy edge detection phase and the optimization stage. Radiograph images are often fuzzy and noisy [9]. The preprocessing stage uses mean 3x3 as smoothing filters. The second stage is the implementation of the fuzzy edge detection algorithms that results in the detection of thick edges [7]. These results are used as search space in the particle swarm optimization algorithm. This stage significantly reduced the search spaces.

2.1 Fuzzy Edge Detection

The initial stage of implementing the fuzzy edge detectors is the representation of images that are in spatial domain into the fuzzy domain. The image of M x N dimension and L levels are taken as an array of fuzzy singleton sets. The edges are detected as in Equation (1). m= 1,2…M; n=1,2…N; $0 \le \mu \le 1$.

$$X = \bigcup_{m=1}^{M} \bigcup_{n=1}^{N} \frac{\mu_{mn}}{g_{mn}} \tag{1}$$

The membership function μ_{mn} is obtained through (2);

$$\mu_{mn} = \frac{g_{mn}}{\max\limits_{i\in\{1,M\},j\in\{1,N\}} g_{i,j}} \tag{2}$$

The image X′ containing all edges is calculated as (3);

$$X = \bigcup_{m=1}^{M} \bigcup_{n=1}^{N} \frac{\hat{\mu}_{mn}}{g_{mn}} \tag{3}$$

Where μ_{mn} = degree of edginess and g_{mn} = center of pixel

Three types of fuzzy edge detection techniques, Gaussian rule based fuzzy (GRBF)[10], robust fuzzy complement (RFC) and classical fuzzy heuristics (CFH). RFC and CFH method are fast fuzzy edge detection algorithms by [5].

1. Gaussian Rule Based Fuzzy Edge Detection (GRBF)

GRBF is a modification of the RBF algorithm. Edge is characterized as relatively high difference between maximum and minimum gray levels. The RBF algorithm rules used in the algorithm are stated below:

If *W is a ω x ω neighborhood pixels,*
and *difference between the minimum gray level in W(g_{min}) and the maximum gray level in W(g_{max}) is normalized by global image intensity range is high*
and *the center pixel(g_{mn}) is equal to mean value,*
then *the edginess is high.*

The edginess membership functions for the above rules are defined as Equation (4) and Equation (5):-

$$\hat{\mu}^1 = \min \frac{\left(1, \max\limits_{i,j\in[1,\omega]} W(i,j) - \min\limits_{i,j\in[1,\omega]} W(i,j)\right)}{\Delta_1} \tag{4}$$

$$\hat{\mu}^2 = 1 - \min \frac{\left(1, g_{mn} - \underset{i,j\in[1,\omega]}{Mean} W(i,j)\right)}{\Delta_2} \tag{5}$$

Where Δ_1= (max$_G$-Min$_G$)/2 and Δ_2= (max$_G$-Min$_G$)/4

The edge map is then calculated using the T-norm in equation (6):

$$X' = \bigcup_{m=1}^{M} \bigcup_{n=1}^{N} \frac{\min(\hat{\mu}^1_{mn}, \hat{\mu}^2_{mn})}{g_{mn}} \tag{6}$$

The Gaussian membership functions are modification from Equation (4) and Equation (5). The equations are defined as in Equation (7) and Equation (8):-

$$\mu_1 = 1.0 - e^{-((\max L - \min L)/\Delta_1)^2 / 2\sigma^2} \tag{7}$$

Where Δ_1 = maxG - minG /1.5 and Δ_2 = maxG – minG.

$$\mu_1 = 1.0 - e^{-((G_{xy} - mean)/\Delta_1)^2 / 2\sigma^2} \tag{8}$$

Where mean is the local mean gray level, G_{xy} is the pixel gray level, maxG is the maximum graylevel value in Global area, minG is the minimum gray level value in Global area, maxL is the maximum gray level value in local area and minL is the minimum graylevel value in local area.

Both of the global and local maximum and minimum values are considered.

2. Robust Fuzzy Complement (RFC)

Robust Fuzzy complement based on fuzzy complement algorithm based on Equation (3). The degree of fuzziness is the fuzzy complement algorithm takes the complement values of X’ as shown in Equation (9). Let $\hat{x}$ be the complement of image X':

$$\hat{x} = 1 - x' = \bigcup_{m=1}^{M} \bigcup_{n=1}^{N} \frac{1 - \overset{\wedge}{\mu}_{mn}}{g_{mn}} \tag{9}$$

The degree of fuzziness γ of an edgy image X' can be calculated with Equation (10).

$$\gamma\left(\hat{x}\right) = \frac{2}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} T\left(\overset{\wedge}{\mu}_{mn}, 1 - \overset{\wedge}{\mu}_{nm}\right) \tag{10}$$

In an optimal edge the fuzziness of complement membership matrix is only high for the row/column containing the gray level transition. An edginess measure is defined as a w x w neighborhood. Equation (11) and Equation (12) is the proposed T-norm.

$$\hat{\mu}_{mn} = \min\left(1, \frac{2}{w}\sum_i \sum_j \min\left(\mu_{ij}, 1 - \mu_{ij}\right)\right) \tag{11}$$

where **T** is t-norm such as minimum operator

$$\hat{\mu} = \min\left(1, \frac{2}{w}\sum_i \sum_j \mu_{ij} x \left(1 - \mu_{ij}\right)\right) \tag{12}$$

To make the (11) more robust and less sensitive to noise the equation is extended as in (12) and (13).

$$\hat{\mu} = \min\left(1, \frac{2}{w}\sum_i \sum_j \mu_{ij} x \left(1 - \mu_{ij}\right)\right) x \left(\frac{\max\limits_{spatial}\left(g_{ij}\right) - \min\limits_{spatial}\left(g_{ij}\right)}{\max\limits_{global}\left(g_{mn}\right)}\right) \tag{13}$$

The extension is a difference of the local maximum gray level value with the local minimum gray level divided with the global maximum. The global maximum implies the contrast of the image.

3. Classical Fuzzy Heuristics (CFH)
The membership is based on the gray level difference of center pixel value g_{mn} and the surrounding neighboring pixels within a window of size WxW as show in Equation (14) [9].

$$\hat{\mu} = \frac{\sum_i \sum_j \left|g_{ij} - g_{mn}\right|}{\Delta + \sum_i^{w} \sum_j^{w} \left|g_{ij}\right| - g_{mn}} \tag{14}$$

The lower Δ, the more edges are detected.

2.2 Particle Swarm Optimization

In 1995, Kennedy and Eberhart introduced an algorithm inspired by social interactions of individuals within a swarm and named it particle swarm Optimization (PSO) [11]. Each particle in the swarm represents a candidate solution to the optimization problem. PSO consists of two main components that is the cognitive and social component. The *cognitive component* quantifies the performance of particle *i* relative to past performances [12],[6]. The *social component* on the other hand quantifies the performance of particle relative to a group of particles, or neighbors.

The PSO can be design as full model, the Social-only model and the Cognitive-only model. Research has shown that the social-only model is faster and more efficient than the full and cognitive-only models [13], [14]. This research focuses more on the social components only.

2.3 The Fuzzy Particle Swarm Optimization Algorithm (FPSO)

Fuzzy edge detection are used to find the IC and OC edges. These edges are thick and are unsuitable for measuring the cortical thickness CT, the internal diameter (ID) and the outer diameter (OD) of the metacarpal [7].

The parameters play an important role in determining the success rate of any evolutionary algorithm. In this research we only consider two parameters, the swarm size

[15] and neighbourhood size [16]. The proposed technique uses a swarm size, n obtained from the fuzzy edge detector. FPSO consists of four steps that is the dynamic population selection, Edge test or the velocity to determine the lbest pixels, the connectivity test or the objective function to determine the final IC or OC edges which is the G_{best} pixels and the stopping function.

Step 1. Dynamic Population Selection: The thick edges detected by fuzzy edge detector algorithm are used as the initial particle selection for FPSO. The population is dynamic because neighborhood pixels that are not selected as optimum will automatically be killed thus the size of the swarm dwindles down very fast.

Step2. Social Network Structure-Edge test or the velocity to determine the lbest pixels: Particles in the neighbourhood communicates among one another through information exchange and moving towards optimum position. The neighbourhood structure that is used in this project is a 3X3 neighbourhood as shown in Figure 2. The image is scanned from the leftmost selected edge pixels population P_{xy}. Then the membership value of pixel $P_{x'y'}$ are tested against pixels P_{x1y1}, P_{x2y2} and P_{x3y3} shown in Figure 1. $P_{x'y'}$ is considered as lbest if its fuzzy membership value of is greater than the other the tested pixels. This pixel replaces P_{xy} as center of the neighborhood and the process is iterated.

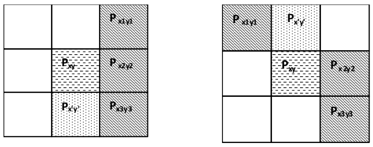
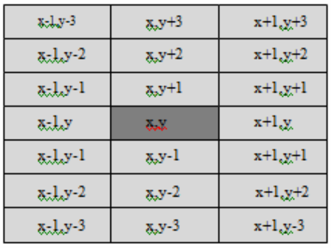


Fig. 1. Edge test pixels in 3x3 neighborhoods

Step3. The connectivity test or the objective function to determine the final IC or OC edges which is the Gbest pixels: This test is to eliminate noise and ensure that the pixel is truly the object edge. The elimination process uses connectivity rule is based on a minimum of five neighboring lbest pixels are connected within a neighborhood of size 3x8 shown in Figure 2. The lbest pixels will be considered as the gbest if more than 5 of the pixels in are selected as lbest. The process is iterated ten times using the neighborhood structure.



x-1,y-3	x,y+3	x+1,y+3
x-1,y-2	x,y+2	x+1,y+2
x-1,y-1	x,y+1	x+1,y+1
x-1,y	x,y	x+1,y
x-1,y-1	x,y-1	x+1,y+1
x-1,y-2	x,y-2	x+1,y+2
x-1,y-3	x,y-3	x+1,y-3

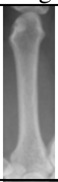
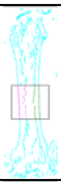
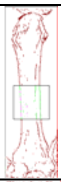
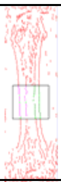
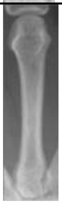
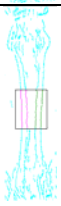
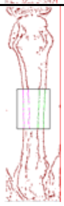
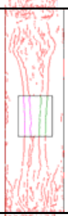
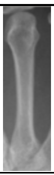
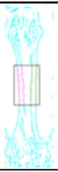
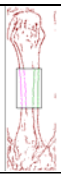
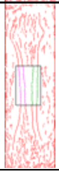
Fig. 2. The neighborhood connectivity test

Step 4. Stopping function: The iteration will only end when the entire particles in the swarm are processed.

3 Results and Discussion

Visually it can be seen that the FPSO is successful in finding a thin edge of the IC and OC. The result of implementing this algorithm can be observed in Table 1.

Table 1. The original image and optimized edge detected image using RFCPSO, GRBFPSO and CFHPSO

	Original	RFCPSO	GRBFPSO	CFHPSO
C2				
C3				
C4				

Samples of images processed with the various FPSO techniques are presented in Table1. GRBFPSO produces the worst results in most cases. The edges detected with the RFCPSO and CFHPSO shows significantly high optimum edge detected where all four complete edges are detected.

3.1 Performance Measure from the Percentage of Complete Edges

The most suitable threshold is selected based on the performance of the various fuzzy edge detectors. This is determined by the most number of complete edges is the detection of all four edges of the left OC, left IC, right IC and the right OC based on a line scan across the bones that are detected by the FPSO algorithm. Summary of the numbers of complete edges are presented in Table 2.

Table 2. Summary of the number of complete LP, LE, RE and RP edges

		MEAN	SD
C2	FPSORFC	70	13.266
	FPSOGRBF	28	19.55677
	FPSOCFH	76	9.959622
C3	FPSORFC	85	10.98201
	FPSOGRBF	66	20.7724
	FPSOCFH	78	16.62275
C4	FPSORFC	62	28.83825
	FPSOGRBF	50	28.03634
	FPSOCFH	67	24.22007

FPSOGRBF detects the least number of edges. Even though PSORFC detects the most outlines, it is more susceptible to noise. This leaves the most suitable cortical edge detection algorithm to be FPSOCFH.

4 Conclusion

In this paper we have presented three edge detection methods that is a hybrid of fuzzy edge detection and particle swarm optimization. The hybrid method proves to be quite efficient in finding the boundary of both the weak and strong edges. It is observed that the boundary for all the three hybrid edge detectors; GRBFPSO; RFCPSO and CFHPSO are quite efficient for the strong edges. RFCPSO and CFHPSO also performs well in detecting the weak edges however boundary obtained for GRBFPSO are very inconsistent and highly disconnected. This is easily quantified by the average number of complete edges detected where GRBFPSO shows a very low average for number of complete edges as compared to RFCPSO and CFHPSO. CFHPSO shows the highest number of complete edges thus indicating it as the most promising algorithm in edge detection for this kind of images.

Acknowledgments. I would like to thank Dr Hanafi Ali for the knowledge radiograph images and Ms Shafaf Ibrahim for her assistance. Also thanking Mr. Mohamad Norman Mohd Nordin for the digitized image.

References

1. O'Donnell, L.: Semi-Automatic Medical Image Segmentation. Department of Electrical Engineering and Computer Science, Thesis, Massachusetts Institute of Technology (2001)
2. Gonzalez, R.C., Woods, R.E.: Digital Image Processing, 2nd edn. Addison Wesley, Reading (2002)
3. Lasaygues, P., et al.: Progress Towards in Vitro Quantitative Imaging of Human Femur Using Compound Quantitative Ultrasonic Tomography. Physics in Medicine and Biology 50, 263–2649 (2005)

4. Thodberg, H.H., Rosholm, A.: Application of the Active Shape Model in a Comercial Medical Device for Bone Densitometry. Image and Vision Computing 21, 1155–1161 (2003)
5. Tizhoosh, H.R.: Fast fuzzy Edge Detection. In: Proceedings of Fuzzy Information Processing Society, NAFIPS, 2002 Annual Meeting of the North American, pp. 239–242 (2002)
6. Eberhart, R.C., Shi, Y.: Particle Swarm Optimization: Developments, Applications and Resources. In: Proceedings of Congress on Evolutionary Computation. IEEE Service Center, NJ (2001)
7. Khalid, N.E.A., Manaf, M., Aziz, M.E.: Modified Fuzzy Heuristics Edge Detection – Tubular Bone Boundary Detection. NSFTA (2008)
8. Kareem, A., Shuaib, L.I., Lee, C.W., Aziz, M.E.: The Determination of Local Malay Female Bone Mineral Density and Its Correlation With Geometric Properties in The Evaluation of Skeletal Status. In: Lights of Medicine 9th National Conference on Medical Sciences (2004)
9. Khalid, N.E.A., et al.: CR Images of Metacarpel Cortical Edge Detection-Bone Profile Histogram Approximation Method. In: Intelligent and Advanced Systems, ICIAS 2007, November 25-28, pp. 702–708 (2007)
10. Khalid, N.E.A., et al.: Gaussian Rule Based Fuzzy (GRBF) Membership Edge Detection on Hand Phantom Radiograph Images. In: Fifth International Conference on Computer Graphics, Imaging and Visualisation, CGIV 2008, August 26-28, pp. 265–270 (2008)
11. Eberhart, R.C., Kennedy, J.: A New Optimizer Using Particle Swarm Theory. In: Proceedings of the Sixth International Symposium on Micromachine and Human Science, Nagoya, Japan, pp. 39–43 (1995)
12. Omran, M.G., Engelbrecht, P.A., Salman, A.: A Color Image Quantization Algorithm Based on Particle Swarm Optimization. Informatica 29, 261–269 (2005)
13. Kennedy, J.: The Particle Swarm: Social Adaptation of Knowledge. In: IEEE International Conference on Evolutionary Computation, pp. 303–308 (1997)
14. Carlisle, A., Dozier, G.: Adapting Particle Swam Optimization to Dynamic Environments. In: The Proceedings of the 2000 International Conference on Artificial Intelligence, pp. 429–433 (2000)
15. Bergh, V.D., Engelbrecht, A.: Effects of Swarm Size on Cooperative Particle Swarm Optimisers. In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2001, pp. 892–899 (2001)
16. Suganthan, P.: Particle Swarm Optimiser with Neighborhood Operator. In: Proceedings of the 1999 Congress on Evolutionary Computation, pp. 1958–1962 (1999)

PSO Heuristics Algorithm for Portfolio Optimization[*]

Yun Chen and Hanhong Zhu

Department of Public Economics & Administration
Shanghai University of Finance and Economics (SUFE)
200433 Shanghai, China
zhh@mail.shufe.edu.cn

Abstract. One of the most studied problems in the financial is the intractability of the portfolios. Some practical formulations of the problem include various kinds of nonlinear constraints and objectives and can be efficiently solved by approximate algorithms. In this paper, we present a meta-heuristic algorithm named Particle Swarm Optimization (PSO) to the construction of optimal risky portfolios for financial investments. The PSO algorithm is tested on two portfolio optimization models and a comparative study with Genetic Algorithm has been implemented. The PSO model demonstrates high computational efficiency in constructing optimal risky portfolios. Preliminary results show that the approach is very promising and achieves results comparable or superior with the state of the art solvers.

Keywords: Swarm Intelligence (SI), Particle Swarm Optimization (PSO), Portfolio Management (PM), Sharp Ratio (SR), Efficient Frontier (EF).

1 Introduction

Portfolio management is one of the most studied topics in finance. The problem is concerned with managing the portfolio of assets that minimizes the risk objectives subjected to the constraint for guaranteeing a given level of returns. In this paper, we deal with the so-called Mean-Variance portfolio selection, which is formulated in the similar way done by Markowitz [1]. His theory has revolutionized the way people think about portfolio of assets, and has gained widespread acceptance as a practical tool for portfolio optimization. But in some cases, the characteristics of the problem, such as its size, real-world requirements [2], very limited computation time, and limited precision in estimating instance parameters, may make analytical methods not particularly suitable for tackling large instances of the constrained Mean-Variance Model. Therefore researchers and practitioners have to resort to heuristic techniques, in which we can found state-of-the-art solvers for the problem.

There are some literatures of solving PO problem using heuristic methods, these methods consist of Genetic Algorithms (GA), tabu search and Simulated Annealing (SA) [3], local search and quadratic programming procedure [4], Ant Colony Optimization (ACO) [5], neural network model [6] and others.

[*] This research is supported by the Shanghai key scientific and technological project (Grant No. 08DZ1120500).

This study presents the PSO algorithm to solve the portfolio optimization problem. PSO is a population based stochastic optimization technique developed in 1995 [7]. Asset allocation in the selected assets is optimized using a PSO based on Markowitz's theory. The rest of the paper is organized as follows. Section 2 describes models for portfolio optimization. In section 3, Back ground of PSO and previous work are summarized. PSO model for optimal portfolio is also discussed. In order to test the efficiency of the proposed PSO solver, a simulation and comparative study with GA heuristics is performed in Section 4. Final conclusions and future research are drawn in Section 5.

2 Models for Portfolio Optimization (PO)

One of the fundamental principles of financial investment is diversification where investors diversify their investments into different types of assets. Portfolio diversification minimizes investors' exposure to risks, and maximizes returns on portfolios. It can be referred to as a multi-objective optimization problem.

There are many methods to solve the multi-objectives optimization problems. One basic method is to transfer the multi-objective optimization problems into a single-objective optimization problem. We can divide these methods into two different types: (1). We can select one important objective function as the objective function to optimize. The rest of objective function can be defined as constraints conditions. (2). We construction an evaluation function which is used as the only objective function. This paper used the second method.

Markowitz Mean-Variance Model

The Markowitz Mean-Variance model [1] for security selection of risky portfolio construction is described as:

$$\text{Min}\sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_{ij} \tag{1}$$

$$\text{Subject to} \sum_{i=1}^{N} w_i r_i = \mathrm{R}^*, \tag{2}$$

$$\sum_{i=1}^{N} w_i = 1, \tag{3}$$

$$0 \le w_i \le 1 \;\; i = 1, \cdots, N. \tag{4}$$

where N is the number of different assets, σ_{ij} is the covariance between returns of assets i and j, w_i is the weight of each stock in the portfolio, r_i is the mean return of stock i and R^* is the desired mean return of the portfolio. We also use the second method to model portfolio optimization problem as the following two models:

2.1 Efficient Frontier Model

We can find the different objective function values by varying desired mean return R^*, so a new named risk aversion parameter $\lambda \in [0,1]$ has been introduced, the sensitivity of the investor to the risk increase as λ increasing from zero to unity. With the λ, the model can be described as:

$$\mathrm{Min}\lambda\left[\sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_{ij}\right]-(1-\lambda)\left[\sum_{i=1}^{N} w_i r_i\right]. \tag{5}$$

$$\text{Subject to}\sum_{i=1}^{N} w_i = 1, \tag{6}$$

$$0 \le w_i \le 1 \;\; i = 1,\cdots,N. \tag{7}$$

In the model included parameter λ, we can draw a continuous curve that is called an efficient frontier according the Markowitz theory[1], the curve composed of mean return and variance according different λ, and every point on an efficient frontier curve indicates an optimum, and this indicates the portfolio optimization problem is a multi-objective optimization.

2.2 Sharpe Ratio Model

Instead of focusing on the mean variance efficient frontier, we seek to optimize the portfolio Sharpe Ratio (SR) [8]. The Sharpe ratio is quite simple and it is a risk-adjusted measure of return that is often used to evaluate the performance of a portfolio. It is described as the following equation:

$$SR = \frac{R_p - R_f}{StdDev(p)} \tag{8}$$

where p is the portfolio, R_p is the mean return of the portfolio p, R_f is the test available rate of return of a risk-free security. StdDev(p) is the standard deviation of R_p. Adjusting the portfolio weights w_i, we can maximizing the portfolio Sharpe Ratio in effect balancing the trade-off between maximizing the expected return and at the same time minimizing the risk. In this study, the PSO optimization can find the most valuable portfolio with good stock combinations.

3 PSO Algorithm for Portfolio Optimization

3.1 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a population based stochastic optimization technique developed by Kennedy and Eberhart in 1995 [7], inspired by social behavior of bird flocking. It belongs to Swarm Intelligence (SI), which originates from the study of natural creatures living in a group. Each individual possess little or no wisdom, but by interacting with each other or the surrounding environment, they can perform very complex tasks as a group.

PSO could be explained well in an imagined scenario: a group of birds are flying in an area to look for food, and there's only one piece of food in this area. the easiest way to find the food is to follow the one who is closest to the food.

The basic concept of PSO lies in accelerating each particle toward its *pbest* which was achieved so far by that particle, and the *gbest* which is the best value obtained so far by any particle in the neighborhood of the particle, with a random weighted acceleration at each time step.

Each particle tries to modify its position using the following information:

- The current positions ($\vec{X}(t)$),
- The current velocities ($\vec{V}(t)$),
- The distance between the *pbest* and the current position ($\vec{P}_i - \vec{X}(t)$),
- The distance between the *gbest* and the current position ($\vec{P}_G - \vec{X}(t)$).

In this paper, we will apply PSO algorithm to solve the portfolio optimization problem.

3.2 Fitness function

Fitness function is a critical factor in the PSO method. Every particle in the PSO's population has a fitness value, and it moves in solution space with respect to its previous position where it has met the best fitness value. In this paper, the sharpe ratio (according equation 8) and the effect frontier model (according equation 5) will be used as objective functions and which are defined as:

$$f_P = \frac{\sum_{i=1}^{N} w_i r_i - R_f}{\sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_{ij}} \tag{9}$$

$$f_P = -\left(\lambda\left[\sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_{ij}\right] - (1-\lambda)\left[\sum_{i=1}^{N} w_i r_i\right]\right) \tag{10}$$

Where f_p is the fitness value of particle *p*. At every step, a particle's personal best position and the best neighbor in the swarm are updated if an improvement in any of the best fitness values is observed.

3.3 Particles Movement

In the algorithm of PSO, each solution is called a "particle", and every particle has its position, velocity, and fitness value. At each iteration, every particle moves towards its personal best position and towards the best particle of the swarm found so far. The velocity changes according to formulation (11):

$$\vec{v}_i(t+1) = w\vec{v}_i(t) + c_1 r_1[\vec{p}_i(t) - \vec{x}_i(t)] + c_2 r_2[\vec{p}_g(t) - \vec{x}_i(t)] \tag{11}$$

where t is the iteration sequence of the particle i, c_1 and c_2 are positive constant parameters called acceleration coefficients which are responsible for controlling the maximum step size, r_1 and r_2 are random numbers between (0, 1), w is a constant. and $\vec{v}_i(t+1)$ is particle *i*'s velocity at iteration *t* + 1. $\vec{v}_i(t)$ is particle *i*'s velocity at iteration t. $\vec{x}_i(t)$ is particle *i*'s position at iteration t. $\vec{p}_i(t)$ is the historical individual best position of the swarm. Finally, the new position of particle *i*, $\vec{x}_i(t+1)$, is calculated as shown in (12)

$$\vec{x}_i(t+1) = \vec{x}_i(t) + \vec{v}_i(t+1) \tag{12}$$

The details about PSO algorithm can be referred to the paper[9]. To improve the performance of PSO, the parameter can be adjusted. For example, the constant *w* can be replaced by formulation (13), and also the constant c_1 and c_2 by function (14) and (15). But in the experiment, the result of large scale portfolio (i.e. 40 stocks) didn't significantly improved.

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \times iter \tag{13}$$

$$c_1 = c_{1\max} - \frac{c_{1\max} - c_{1\min}}{iter_{\max}} \times iter \tag{14}$$

$$c_2 = c_{2\max} - \frac{c_{2\max} - c_{2\min}}{iter_{\max}} \times iter \tag{15}$$

4 Experiments and Discussion

The PSO experiments for the portfolio optimization has been performed on two restricted risky portfolio of 8 stocks and 15 stocks based on two portfolio optimization models. Table 1 and table 2 have shown the results from PSO algorithm and genetic algorithm (GA) about the two portfolios. All stocks are selected from the Shanghai Stock Exchange 50 Index (the SSE 50 Index). Individual stock's historical daily returns are selected from 1 January 2009 to 3 April 2009.

In order to evaluate the performance of PSO algorithm, we compare PSO with another heuristic Algorithm, named GA. In the experiments, PSO algorithm has been developed using Matlab as software development tool. GA has been developed using the software named GeneHunter [11].

Base on the efficient frontier model, the results of the optimal risky portfolios developed by PSO and GA algorithms for the two portfolios are shown in the Table 1. In the portfolio of 8 stocks and 15 stocks, The fitness values obtained by PSO are all better than those of the GA algorithms.

Base on the Sharpe Ratio model, the results of the optimal risky portfolios developed by PSO and GA algorithms for the two portfolios are shown in the Table 2. In the portfolio of 8 and 15 stocks, when the risk free is 0 and 0.3%, the Sharpe Ratio value obtained by PSO are all better than those of the GA algorithms.

Taking the sets of optimal portfolios obtained with PSO and GA algorithms, we trace out their efficient frontiers in Figure 1 (a)-(d). In situation of the different portfolio optimization models and different portfolios, the PSO efficient frontier is always upon those of the GA algorithm. It means that we can get higher mean return under the same risk, or lower risk under the same mean return from the PSO portfolio than GA portfolios. So PSO portfolio is the best solution.

At the same time, we take the sets of optimal portfolios obtained with PSO on Sharpe Ratio (SR) model and Efficient Frontier (EF) model, we trace out their efficient frontiers in figure 2.(a) and (b). In the portfolio of 8 stocks, when the standard deviation is lower (about 0.03), the efficient frontier based on the SR model is upon those of the EF model. But when the standard deviation increasing, the efficient frontier based on the EF model will be upon those of the SR model. But in the portfolio of

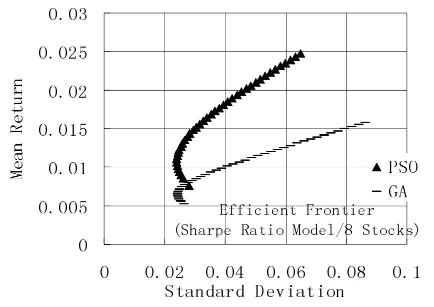
Table 1. Two portfolios' results of PSO and GA algorithms based on the efficient frontier model

λ	PSO/GA	StdDev		ER		Fitness Value	
		8	15	8	15	8	15
1	PSO	0.05%	0.09%	0.57%	0.98%	-0.05%	0.98%
	GA	0.06%	0.07%	0.74%	0.81%	-0.06%	0.81%
0.8	PSO	0.08%	0.09%	0.90%	0.97%	0.12%	0.76%
	GA	0.05%	0.07%	0.74%	0.84%	0.11%	0.66%
0.6	PSO	0.11%	0.10%	0.97%	0.97%	0.32%	0.54%
	GA	0.07%	0.07%	0.74%	0.84%	0.25%	0.48%
0.4	PSO	0.10%	0.09%	0.96%	0.97%	0.53%	0.34%
	GA	0.05%	0.07%	0.74%	0.83%	0.42%	0.29%
0.2	PSO	0.10%	0.08%	0.94%	0.95%	0.74%	0.13%
	GA	0.06%	0.06%	0.74%	0.82%	0.58%	0.11%
0	PSO	0.11%	0.04%	0.96%	0.50%	0.96%	-0.04%
	GA	0.06%	0.05%	0.74%	0.64%	0.74%	-0.05%

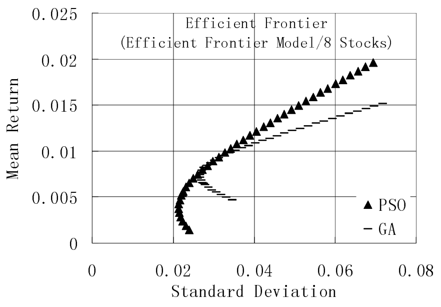
Table 2. Two portfolios' results of PSO and GA algorithms based on the sharpe ratio model

Stocks	RiskFree	PSO/GA	ER	StdDev	Sharpe Ratio
8	0	GA	0.66%	2.42%	27.27%
		PSO	0.86%	2.61%	32.75%
	0.3%	GA	0.53%	2.63%	12.42%
		PSO	0.72%	2.90%	17.83%
15	0	GA	0.69%	2.69%	25.63%
		PSO	0.89%	2.58%	34.50%
	0.3%	GA	0.57%	2.47%	18.96%
		PSO	0.84%	2.76%	26.73%

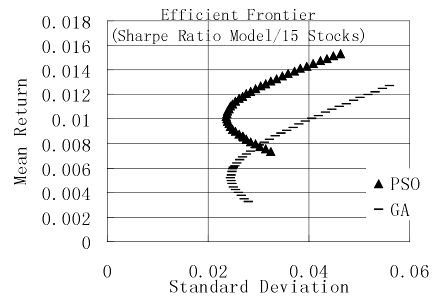
Data has been download from the web: http://finance.yahoo.com. The Risk Free has been selected subjectively. ER is 'expected return', StdDev is 'standard deviation', λ is the risk aversion parameter in the efficient frontier model.



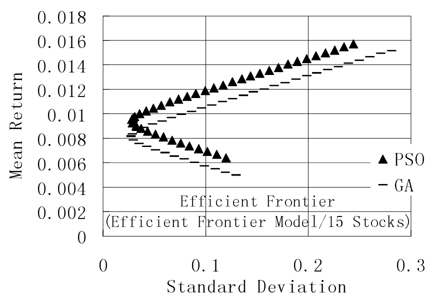
(a) 8 stock, Sharpe Ratio Model



(b) 8 Stock, Efficient Frontier Model

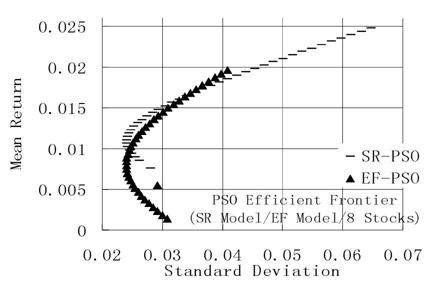


(c) 15Stock, Sharpe Ratio Model

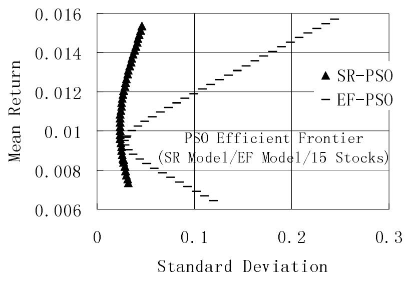


(d) 15Stock, Sharpe Ratio Model

Fig. 1. The Efficient frontier of the portfolio gotten from PSO and GA heuristics



(a) the portfolio of 8 stocks



(b) the portfolio of 15 stocks

Fig. 2. The efficient frontier of the portfolio gotten from PSO on SR model and EF model

15 stocks, the efficient frontier obtained based on SR model is always upon those of EF model. So we can come to conclude that portfolio optimization model is very important for the portfolio optimization.

To sum up, from the experiments on different portfolios and different portfolio optimization models, we can conclude that the PSO approach is better than the GA algorithm in this case. PSO algorithm clearly shows the efficiency and effectiveness of solving high-dimensional constrained optimization problems.

5 Conclusion

The paper focuses on solving the portfolio optimization problem in finance investment management. A meta-heuristic Particle Swarm Optimization method has been developed to optimize investment portfolios, in which the objective functions and constraints are based on the Markowitz model, the Sharp Ratio (SR) model and the Efficient Frontier (EF) model. In order to make a valid comparison with other methods, different test problems were solved and the results obtained when compared with the results of Genetic Algorithms (GA) demonstrated the superiority of the PSO algorithm. At the same time, the portfolio optimization model is a key factor for the portfolio management. Future research may be conducted to further investigate the application of some derived models or hybrid models of PSO to other investment strategy problems.

Acknowledgments. The authors thank Prof. Kesheng Wang (*Norway University of Science and Technology, NTNU. Norway)* for his valualbe ideas and comments.

References

1. Markowitz, H.: Portfolio Selection. The Journal of Finance 7, 77–91 (1952)
2. Zhou, X.Y., Li, D.: Continuous-Time Mean-Variance Portfolio Selection: A Stochastic LQ Framework. Applied Mathematics and Optimization 42, 19–33 (2000)
3. Chang, T.J., Meade, N., Beasley, J.E., Sharaiha, Y.M.: Heuristics for Cardinality Constrained Portfolio Optimisation. Computers & Operations Research 27, 1271 (2000)
4. Gaspero, L.D., Tollo, G.d., Roli, A., Schaerf, A.: Hybrid Metaheuristics for Portfolio Selection Problems. In: MIC 2007 - Metaheuristics International Conference, Montreal (2007)
5. Doerner, K., Gutjahr, W., Hartl, R., Strauss, C., Stummer, C.: Pareto Ant Colony Optimization: A Metaheuristic Approach to Multiobjective Portfolio Selection. Annals of Operations Research 131, 79–99 (2004)
6. Giovanis, E.: The Arbitrage Pricing Theory and the Capital Asset Pricing Models and Artificial Neural Networks Modeling with Particle Swarm Optimization (PSO). SSRN eLibrary (2009)
7. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: International Conference on Neural Networks, ICNN 1995, p. 1942 (1995)
8. Sharpe, W.F.: Mutual Fund Performance. The Journal of Business 39, 119–138 (1966)
9. Bratton, D., Kennedy, J.: Defining a Standard for Particle Swarm Optimization. In: Swarm Intelligence Symposium, SIS 2007, pp. 120–127. IEEE, Los Alamitos (2007)
10. Black, F.: Capital Market Equilibrium with Restricted Borrowing. The Journal of Business 45, 444–455 (1972)
11. Wang, K.: Applied Computational Intelligence in Intelligent Manufacturing Systems. Advanced Knowledge International (2005)

A New Particle Swarm Optimization Solution to Nonconvex Economic Dispatch Problem

Jianhua Zhang, Yingxin Wang, Rui Wang, and Guolian Hou

North China Electric Power University, Beijing, 102206 China
zhangwu@public.bta.net.cn

Abstract. This paper presents an optimal economic dispatch for power plants by using modified particle swarm optimization (PSO) algorithm. The economic dispatch problem in power systems is to determine the optimal combination of power outputs for all generating units in order that the total fuel cost can be minimized, furthermore, all practical constraints can be satisfied. Several key factors in terms of valve-point effects of coal cost functions, unit operation constraints and power balance are considered in the computation models. Consequently, a new adaptive PSO technique is utilized for solving economic dispatch problems. The proposed algorithm is compared with other PSO algorithms. Simulation results show that the proposed method is feasible and efficient.

Keywords: Economic dispatch, Particle swarm optimization.

1 Introduction

Economic load dispatch problem is allocating loads to available generation units for minimum cost while meeting the constraints. It can be formulated as an optimization problem of minimizing the total fuel cost of all committed plant while meeting the demand and losses. Modern power system is experiencing increased demand for electricity with related expansions in system size, which has resulted in higher number of generators and lower reserve margins making the economic dispatch (ED) problem more challenging and complicated.

There have been many algorithms proposed for economic dispatch. In the literature dedicated to the optimization of cost objective function, many researchers proposed different solutions [1]. Due to the nature of large scale, valve-point loadings, nonlinear generation cost and multiple constraints, ED problem combines a highly nonlinear, nonconvex and computationally difficult environment with a need for optimality [2]. In the traditional ED problem, the cost function for each generator has been approximately represented by a single quadratic function and solved by using mathematical programming, such as, nonlinear programming [3], linear programming, homogenous linear programming [4], dynamic programming [5], quadratic programming [6] and so on.

The basic ED considers the power balance constraint apart from the generating capacity limits, a practical ED must take ramp rate limits, prohibited operating zones and valve-point effects options into consideration so as to provide the completeness for the ED formulation. The resulting ED is a challenging problem. PSO is one of the modern heuristic algorithms, which can be used to solve nonlinear and non-continuous optimization problems [7]. PSO can be easily applied to nonlinear and non-continuous optimization problems. There are some modified particle swarm optimization algorithms, for example, learning factor is linearly decreased in velocity update relaxed (VUR) PSO so as to have a linear transition of search ability from global to local search.

This paper introduces a new adaptive PSO (APSO) applied to non-convex ED problems. Compared with the conventional PSO and VURPSO approaches, APSO has a better dynamic balance between global and local search abilities due to nonlinear and dynamic change in the inertia weight during the iterations.

2 The Economic Dispatch Model

2.1 Objective Function

The optimal ED problem can be modeled as an optimization problem. The objective of the ED problem is to minimize the total generation costs of a power system with the operating constraints of a power system. In all practical cases, the fuel cost of generator i with power output P_i can be formulated as a quadratic function of real power generation.

$$f(P_i) = \sum_{i}^{n} (aP_i^2 + bP_i + c) \cdot \tag{1}$$

Wire drawing effects occurs when each steam admission valve in a turbine starts to open, and at the same time a rippling effect on the unit curve is produced. Ignoring valve point effects, some inaccuracy would be introduced into the resulting dispatch. To model the effects of "valve-points", a recurring rectified sinusoid contribution is added to the above cost function. Since the valve point results in the ripples, a cost function contains higher order nonlinearity. Therefore, the cost function (1) should be replaced by (2) in order to consider the valve-point effects as follows:

$$f(P_i) = \sum_{i}^{n} (aP_i^2 + bP_i + c) + \sum_{i}^{n} E_i \tag{2}$$

$$E_i = \left| g_i \sin(h_i (P_i - P_{i\min})) \right| \tag{3}$$

where g_i and h_i are the valve-points coefficients. $P_{i\min}$ is the lower generation limit of unit i.

2.2 Operational Limitations and Constraint

In general, the operational limitations and constraints considered in this paper are listed below:

(1) Unit operation constraints

Generation output of each generator should be laid between maximum and minimum limits. The corresponding inequality constraints for each generator are:

$$P_{\min} \le P_i \le P_{\max} \tag{4}$$

where $P_{\min}$ and $P_{\max}$ are minimum and maximum output of generator i, respectively.

(2) Power balance
For power balance, an equality constraint should be satisfied. The total generated power should be the same as total load demand plus the total line loss

$$\sum_{i}^{n} P_i = P_L + P_D \tag{5}$$

where P_D is the total system demand and P_L the total line loss. Nevertheless, the transmission loss is not considered in this paper for simplicity.

2.3 Optimization of the Objective Function

Considering the above condition, the mathematical model of the economic dispatch for n units can be established as follows

$$\min F = \sum_{i}^{n} F_i(P_i) = \sum_{i}^{n} [f_i(P_i) + E_i] \tag{6}$$

$$\text{St} \quad P_{\min} \le P_i \le P_{\max}$$

$$\sum_{i}^{n} P_i = P_D$$

Therefore, the objective function is selected as follow:

$$f = \sum_{i}^{n} f_i(P_i) + \sum^{n} E_i + k \left| \sum_{i}^{n} P_i - P_{LD} \right| \quad i = 1, 2, \cdots n \tag{7}$$

where k is penalty function coefficient.

3 Particle Swam Optimization Algorithm

3.1 Conventional Standard PSO

Particle swam optimization is an intelligent optimization algorithm. It is similar to other intelligent optimization algorithms in that the algorithm is initialized with a population of random solutions. In the original form of PSO, each particle in a swarm population adjusts its position in the search space by tracking two extreme values, which are the best position ($Pbest$) found so far, and the position of the known best-fit particle ($Gbest$) in the entire population. Unlike other population-based evolutionary algorithms, i.e., genetic algorithms, instead of using genetic operators such as crossover and mutation, PSO each candidate solution is associated with a velocity. The candidate solutions, called particles, then "fly" through the search space. The velocity is constantly adjusted according to the corresponding particle's flying experience and its companions' flying experience. The updating formula for each particle's velocity v_{ij} and position x_{ij} in conventional standard PSO is written as:

$$v_{ij}(t+1) = v_{ij}(t)(\; c_1(Pbest_i(t) - x_{ij}(t)) + c_2(Gbest_i(t) - x_{ij}(t))) \tag{8}$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \tag{9}$$

In. (8) and (9), the learning factors c_1 and c_2 are nonnegative constants, c_1 and c_2 are random numbers uniformly distributed in the interval [0, 1], $v_{ij} \in [-v_{\max}, v_{\max}]$, where $v_{\max}$ is a designated maximum velocity which is a constant preset by users according to the objective optimization function.

It has been found that the "flying" particles velocities are only determined by their current positions and their best positions in history. The velocity itself is memoryless, it leads to the final solution is heavily dependent on the initial seeds (population). In this case, by adding the first term in Eq. (10), the particles obtain the ability to expand the search space, that is, they have a tendency to explore the new area. So that PSO more likely has global search ability by adding the first term. The updating formula for each particle's velocity and position in conventional standard PSO is expressed as follows:

$$v_{ij}(t+1) = wv_{ij}(t) + c_1 rand(Pbest_i(t) - x_{ij}(t)) + c_2 rand(Gbest_i(t) - x_{ij}(t)) \tag{10}$$

where $rand$ is a random number uniformly distributed in [0,1]. w plays the role of balancing the global search and local search.

3.2 Relaxed PSO

The concept of a varying inertia weight has been introduced to balance the global wide range exploitation and local nearby exploration abilities of the swarm, which performs much better than fixed inertia weight. The weight is updated as follows:

$$w = w_{\max} - gen * (w_{\max} - w_{\min}) / gen_{\max} \tag{11}$$

where $gen_{\max}$ is the maximum iterations set by the user. $w_{\max}$ and $w_{\min}$ are the maximum and minimum inertia weight respectively, which can be set by the user, for example, $w_{\max}$ =0.9, $w_{\min}$ =0.4.

3.3 Adaptive PSO

In our work, an adaptive PSO algorithm is proposed to improve its performance. Different particles are allocated with different tasks. The particles with better performance have larger inertia weight, which are in charge of searching better area. The particles with poor performance are weighted by smaller inertia, allowing them to quickly converge to a better area for detailed search. Moreover, a large inertia weight facilitates a global search while a small inertia weight facilitates a local search. The particles are sorted according to their individual optimal location from excellent to poor. The inertia weight and positive constant of the particle, which is in i place, are expressed as follows:

$$w_i = w_{\min} + (w_{\max} - w_{\min})(m-i)/(m-1) \tag{12}$$

$$c_{i1} = c_{i2} = (w_i + 1 + 2\sqrt{w_i})/2 \tag{13}$$

m is the population size, the inertia weight w_i is adjusted adaptively.

4 Case Study

In this section, some numerical examples are provided to highlight the main features of the proposed APSO approach.

4.1 Case 1

The test system is composed of 5 generating units and the registered capacity of units is $P_{\max}$ =600MW, $P_{\min}$ =240MW [8]. The parameter of inertia weight associated with PSO is 0.9 and VURPSO linearly decreasing inertia weight from 1.0 to 0.5 with iteration cycle. The APSO inertia weight values are between 0.4 and 1.0. Maximum allowed iteration cycles are taken as 350 with swarm size of 50 particles for all the PSO versions.

In this case, the fuel cost of the generating units are given as follows:

$$\begin{cases} F_1 = -1.614\times10^{-7}P_1^3 + 4.39\times10^{-4}P_1^2 - 0.3265P_1 + 384 \\ F_2 = 0.0003P_2^2 - 0.3651P_2^2 + 404.63 \\ F_3 = -2.7719\times10^{-7}P_3^3 + 5.6722\times10^{-4}P_3^2 - 0.40829P_3 + 411.8 \\ F_4 = 3.2044\times10^{-7}P_4^3 - 2.1458\times10^{-4}P_4^2 + 0.3265P_4 + 370 \\ F_5 = 7.9055\times10^{-7}P_5^3 - 7.07328\times10^{-4}P_5^2 + 5.1926P_5 + 361.2 \end{cases}$$

$$240MW < P_i < 600MW$$

The fitness curves of four different algorithms are shown in Fig. 1. It is clear that the APSO and VURPSO are almost similar in convergence and show their superiority over the PSO algorithm. APSO is slightly better than VURPSO, because APSO has better information sharing and conveying mechanism than the others.

The best power outputs obtained by three PSO strategies are compared with improved mutative scale chaotic optimization [8], the final results are shown in Table 1. It can be shown that APSO has the highest probability of achieving better solutions.

4.2 Case 2

This system consists of 3 generating units and the input data of 3-generator system are given in Table 2 [9]. The total demand for the system is set to 850MW. The simulation results compared with conventional PSO and modified PSO are shown in Table3. It is clear that the APSO has succeeded in finding a global optimal solution.

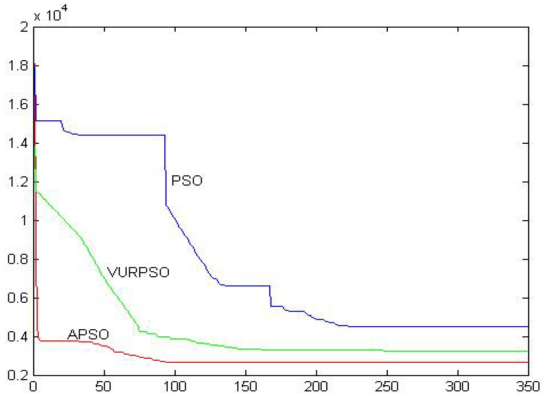


Fig. 1. Comparative results among three PSO strategies

Table 1. The comparison of statistical test results

Algorithm	Total power (MW)	P_1 (MW)	P_2 (MW)	P_3 (MW)	P_4 (MW)	P_5 (MW)	Fuel cost ($)
PSO		462.12	547.15	448.34	425.82	516.78	4792.9
VURPSO		560.84	466.73	600.00	600.00	241.42	2672.0
APSO	2469.64	456.14	572.85	600.00	600.00	240.00	2659.9
Literature[8]		396.16	512.96	450.00	598.24	512.56	4039.3
PSO		510.65	430.10	324.30	452.92	430.34	5941.6
VURPSO		600.00	600.00	388.08	546.54	240.00	2684.8
APSO	2374.63	467.82	554.71	512.06	600.00	240.00	2663.4
Literature[8]		350.49	544.06	499.23	450.63	530.40	4169.1
PSO		355.93	468.25	600.00	600.00	493.96	5884.5
VURPSO		600.00	600.00	600.00	600.00	318.88	3052.8
APSO	2718.88	600.00	600.00	600.00	600.00	318.88	3052.8
Literature[8]		418.85	600.00	583.20	575.35	539.66	4171.4

Table 2. Data for test case 2(3-unit system)

S.no	a	b	c	g_i	h_i	$P_{i\min}$ (MW)	$P_{i\max}$ (MW)
1	0.001562	7.92	561	300	0.0315	100	600
2	0.00194	7.85	310	200	0.042	100	400
3	0.00482	7.97	78	150	0.063	50	200

Table 3. Best simulation results from 3 kinds of PSO strategies

Output powers (MW)	Best result		
	PSO	VURPSO	APSO
P1(MW)	243.00	270.42	365.52
P2(MW)	357.48	331.94	319.26
P3(MW)	167.03	70.94	138.21
Fuel cost(MW)	8569.2	8385.1	8209.7

5 Conclusions

In this paper, a new modified PSO algorithm is presented for building the optimal economic dispatch strategy for generation companies in power markets. Economic dispatch is a multi-objective problem. To prove the ability of the proposed APSO in solving non-convex optimization problems, ED problems with non-convex solution spaces are considered and solved by three different PSO strategies (PSO, VURPSO, and APSO). The results show that the proposed APSO has obtained satisfactory solution for generation companies.

Acknowledgments. This work is supported by National Natural Science Foundation of China under grants (No. 60674051, 60974029). These are gratefully acknowledged.

References

1. Chowdhury, B.H., Rahman, S.: A Review of Recent Advances in Economic Dispatch. IEEE Trans. Power Systems 5, 1248–1259 (1990)
2. Lin, W.M., Cheng, F.S., Tsay, M.T.: Nonconvex Economic Dispatch by Integrated Artificial Intelligence. IEEE Trans. Power Systems 16, 307–311 (2001)
3. Nanda, J., Hari, L., Khotari, M.L.: Economic Emission Load Dispatch with Line Flow Constraints Using a Classical Technique. IEE Proc. of Generation Transmission And Distribution 141, 1–10 (1994)
4. Rabin, J., Coonick, A.H., Cory, B.J.: A Homogeneous Linear Programming Algorithm for the Security Constrained Economic Dispatch Problem. IEEE Trans. Power Systems 15, 930–936 (2000)
5. Ross, D.W., Kim, S.: Dynamic Economic Dispatch of Generation. IEEE Trans. Power Apparatus and Systems 99, 2060–2068 (1980)
6. Fan, J.Y., Zhang, L.: Real-Time Economic Dispatch with Line Flow and Emission Constraints Using Quadratic Programming. IEEE Trans. Power Systems 13, 320–325 (1998)
7. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: Proc. IEEE Int. Conf. Neural Networks, pp. 1942–1948 (1995)
8. Wang, S., Han, F., Zhu, H.: Economic Load Dispatch Base on Improved Mutative Scale Chaotic Optimization. Proceedings of the CSEE 25, 90–95 (2005)
9. Selvakumar, A.I., Thanushkod, K.: A New Particle Swarm Optimization Solution to Nonconvex Economic Dispatch Problem. IEEE Trans. Power Systems 22, 42–51 (2007)

Optimal Micro-siting of Wind Farms by Particle Swarm Optimization

Chunqiu Wan[1], Jun Wang[1,3,⋆], Geng Yang[1], and Xing Zhang[2]

[1] Department of Automation, Tsinghua University, Beijing, China
[2] School of Aerospace, Tsinghua University, Beijing, China
[3] Dept. of Control Science & Engn., Tongji University, Shanghai, China
jwang@ieee.org

Abstract. This paper proposes a novel approach to optimal placement of wind turbines in the continuous space of a wind farm. The control objective is to maximize the power produced by a farm with a fixed number of turbines while guaranteeing the distance between turbines no less than the allowed minimal distance for turbine operation safety. The problem of wind farm micro-siting with space constraints is formulated to a constrained optimization problem and solved by a particle swarm optimization (PSO) algorithm based on penalty functions. Simulation results demonstrate that the PSO approach is more suitable and effective for micro-siting than the classical binary-coded genetic algorithms.

Keywords: wind farm micro-siting, particle swarm optimization, penalty function.

1 Introduction

Micro-siting is one of the most fundamental problems of wind farm design. Unfortunately, only few studies were carried out. The most commonly adopted scheme, also known as the empirical method, was the staggered siting scheme, which is mainly suitable for a relatively flat wind farm with a dominant wind direction [1]. Mosetti, Poloni and Diviacco first systematically optimized turbine positions in a wind farm by a genetic algorithm (GA) [2]. Their work was further improved by Grady, Hussaini and Abdullah [3], Marmidis, Lazarou and Pyrgioti [4], and Wan, Wang, Yang, et al. [5].

In the above studies, a wind farm was partitioned into square cells, the width of which is usually five times of the diameter of the turbine rotor for operation safety. The turbines could only be placed in the center of each cell. Although this kind of "discret" siting is convenient for the realization of optimal methods, some freedom was lost so was the performance of the wind farm. Wan et al. [6] proposed an algorithm to allow each turbine to be freely adjusted inside its cell in order to increase the energy generated by the wind farm. Unfortunately, as the constraints on turbine-minimum-distance were not considered, the turbines could be placed quite closely especially when the wind direction was dominant.

⋆ Corresponding author.

The particle swarm optimization (PSO) algorithm is easy to implement and converges quickly. Furthermore, the PSO algorithm is especially suitable for real-valued optimization problem [7,8]. In this paper, a PSO algorithm based on penalty functions is proposed for the optimal micro-siting of a wind farm in a spatially-continuous manner.

2 Formulation of Wind and Turbine Models

For completeness and clarity, this section presents the models of wind directions and speeds, the wake effects among turbines and the power evaluation of turbines.

2.1 Models of Wind Direction and Speed

The wind energy rose map is used to describe the characteristics of wind direction and speed variations of a wind farm. The length of each segment of the rose map represents the frequency of the corresponding wind speed in a certain direction.

2.2 Models of Wake Effects

In a wind farm, wake effects among turbines could be approximated by a linear wake model [9]. According to the momentum balance theorem and the Betz theory, the wake speed at a given downstream distance d is [9,10]

$$\widetilde{u}_0(d) = u_0 \left(1 - \left(1 - \sqrt{1 - C_T}\right)\left(\frac{D}{D + 2\alpha d}\right)^2\right) \tag{1}$$

where u_0 is the freedom wind speed, C_T is the turbine thrust coefficient, D is the turbine rotor diameter, and α is the wake spreading constant.

For a wind farm with N turbines, the wind speed of the ith turbine, u_i, can be computed according to the theory of kinetic energy balance [9,10].

2.3 Models of Turbine Power

The different wind directions and speeds are not considered in the above wake models for clarity. Suppose a wind farm has N turbines and the wind has M directions. Moreover, each wind direction is partitioned to P speeds. The total power of the wind farm could be calculated by [2]

$$\mathcal{P}(Z) = \sum_{i=1}^{M} \sum_{j=1}^{P} f_{i,j} \left(\sum_{k=1}^{N} 0.3 u_{i,j,k}^3\right) \tag{2}$$

where $f_{i,j}$ is the percentage of the jth wind speed in the ith wind direction, $u_{i,j,k}$ is the actual wind speed on the kth wind turbine under condition of the jth wind speed in the ith wind direction.

3 Optimization Methods

Suppose the allowed minimum distance between turbines is $d_{\min}$. The optimal micro-siting problem is to maximize the power produced by the wind farm, i.e.

$$\max \mathcal{P}(Z) \tag{3}$$

subject to the constraints on turbine positions

$$\begin{cases} g_k(Z) = (x_m - x_n)^2 + (y_m - y_n)^2 - d_{\min}^2 \geq 0 \\ 0 \leq x_m \leq X_{\max}, \qquad 0 \leq y_m \leq Y_{\max} \end{cases} \tag{4}$$

$$(k \in \{1, 2, ..., C\}, \ \forall m \neq n, \ m, n \in \{1, 2, ..., N\})$$

where (x_m, y_m) represents the the Cartesian coordinates of the mth wind turbine, $g_k(Z)$ represents the position constraint between the mth and the nth turbines, $C = N(N-1)/2$ is the number of constraints.

The wind farm micro-siting problem is a constrained optimization problem with a large number of constraints.

3.1 Penalty Function Method

The penalty function method is most commonly utilized to handle constraints in optimization problems. It transforms a constrained optimization problem into an unconstrained one by penalizing the infeasible solution based on the amount of constraint violations [11].

According to Equations (3) and (4), the present optimization problem is a maximization problem. The fitness of a potential solution can be evaluated by

$$\mathcal{F}(Z) = \mathcal{P}(Z) - \gamma\phi(Z) \tag{5}$$

where γ is the penalty coefficient. $\phi(Z)$ is the constraint violation value of the solution Z and can be evaluated by

$$\phi(Z) = \sqrt{\sum_{k=1}^{C} |\min\{0, g_k(Z)\}|} \tag{6}$$

It represents the distance between the infeasible solution and the feasible region. $\phi(Z) = 0$ indicates that the solution is feasible, while $\phi(Z) > 0$ infeasible.

In this paper, the penalty function method transforms the constrained optimization problem into an unconstrained one so that the computational complexity is reduced.

3.2 Particle Swarm Optimization

The PSO algorithm is a population-based, global and stochastic optimization algorithm. It was inspired by social behavior of fish schooling and bird flocking, and developed by Kennedy and Eberhart in 1995 [12]. It is easier to implement and has a faster convergence rate than the traditional evolutionary algorithms [7,8]. Additionally, it is computational inexpensive and especially suitable for the present complex continuous problem.

The PSO algorithm starts with a population of particles whose positions $\mathbb{Z}$ and velocities $\mathbb{V}$ are randomly initialized in the search space. In the present study, $\mathbb{Z} = [Z_1^T \ Z_2^T \ \cdots \ Z_{PS}^T]^T$ represents the position matrix of turbines for all the potential solutions in the swarm, "PS" represents the population size, and $\mathbb{V} = [V_1^T \ V_2^T \ \cdots \ V_{PS}^T]^T$ represents the modification to the position $\mathbb{Z}$. The search for optimal position is carried out by updating the velocities and positions of the particles iteratively. The search of particles is focused toward promising regions by biasing the particles' velocities vector toward both their own historical best positions $\mathbb{Z}^p$ and the swarm's historical best position $\mathbb{Z}^g$. Here, the *best* position of a particle or the swarm is corresponding to the smallest fitness value defined in Equation (5) .

The velocities and positions of particles are usually updated by the following equations [7,8,13]:

$$\mathbb{V}(t+1) = w\mathbb{V}(t) + c_1 R_1 .* (\mathbb{Z}^p(t) - \mathbb{Z}(t)) + c_2 R_2 .* (\mathbb{Z}^g(t) - \mathbb{Z}(t)) \tag{7}$$

$$\mathbb{Z}(t+1) = \mathbb{Z}(t) + \mathbb{V}(t+1) \tag{8}$$

where t is the generation index, w is the *inertial weight* which balances the global and local search ability, c_1 and c_2 are *cognitive* and *social* parameters respectively, R_1 and R_2 are random matrices of a dimension $PS \times 2N$ whose elements are uniformly distributed within $[0, 1]$, and "$.*$" represents the element-by-element product.

In order to improve the convergence ability of the PSO, a constriction factor K can be added to the velocity update rule. The effect of the constriction factor is to reduce the velocity of the particles as the search progresses and thereby contract the overall swarm diameter. This in turn results in a progressively smaller domain being searched. In this case, the velocities of particles are updated as follows [14]:

$$\mathbb{V}(t+1) = K(\mathbb{V}(t) + c_1 R_1 .* (\mathbb{Z}^p(t) - \mathbb{Z}(t)) + c_2 R_2 .* (\mathbb{Z}^g(t) - \mathbb{Z}(t))) \tag{9}$$

where $K = \frac{2}{|2-c-\sqrt{c^2-4c}|}$, $c = c_1 + c_2 > 4$. Experimental results pointed out that $c_1 = 2.8$ and $c_2 = 1.3$ yielded good results for the test problems [8,15]. In this paper, this type of PSO algorithm is used to solve the present problem.

The maximum velocity $V_{\max}$ of the particle is limited to the dynamic range of the search space [14]. The convergent criterion is that the relative improvement in consecutive 100 generations is less than 0.01% and the algorithm has proceeded at least 1000 generations.

4 Results and Discussions

In this paper, computer simulations are carried out by MATLAB. The allowed minimum distance between turbines is set as $d_{\min} = 4D$, where $D = 50$m is the turbine diameter [10]. As the Grady's results [3] are the most classical ones, the present optimal results are compared with Grady's under the same wind farm conditions in order to demonstrate the effectiveness of the proposed approach.

In Grady's study [3], a square wind farm was partitioned into a 10×10 grid. The turbines were only allowed to be installed in the center of each cell with the objective of minimizing the cost per unit energy. Genetic algorithms were used to solve the optimization problem.

The characteristics of the wind turbines in Grady's paper are as follows: The hub height of the turbines is 60m, rotor diameter 40m, thrust coefficient 0.88. The turbine thrust coefficient is considered constant throughout the process. The farm is a 2000m $\times$ 2000m site and the ground roughness is $z_0 = 0.3$m.

In Grady's study, three cases with different complexities were investigated: (a) uniform wind direction with a speed 12m/s; (b) 36-direction evenly distributed wind with a speed 12m/s; (c) 36-direction distributed wind with varied probability of the speeds 8, 12, and 17m/s. The wind roses for three cases are shown in Fig. 1.

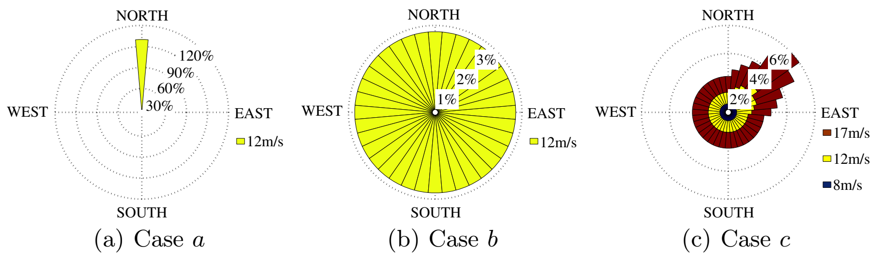


Fig. 1. The wind roses for the three cases in Grady's paper

The optimal wind farm configurations of Grady's study for the three cases are shown in Fig. 2 and the numbers of installed turbines are 30, 39 and 39, respectively. The limitation of Grady's study is that the positions of turbines could not been adjusted freely in the continuous space.

In the present study, the PSO algorithm with penalty functions is applied to the same cases. The wind farm model and parameters are the same as given in [3] while the objective function is to maximize the total power extracted from the wind farm. The same type of wind turbines as in [3] are used. The numbers of turbines for all three cases are set to be the same as those of Grady's. In Grady's study, the wind farm is a square of 2000m $\times$ 2000m and the width of each cell is 200m. Since the turbines are only installed in the center of each cell, the effective domain of the wind farm in Grady's study is 1800m$\times$1800m. Thus, the computational domain of the present study is set as 1800m$\times$1800m so that the optimal results could be fairly compared with Grady's study.

The population size for the optimization is set to the twice of the total turbine number. The penalty coefficients γ for the Cases a, b and c are chosen to be 500, 500 and 1000 respectively by trial-and-error tunings. The wind farm configurations are shown in Fig. 3.

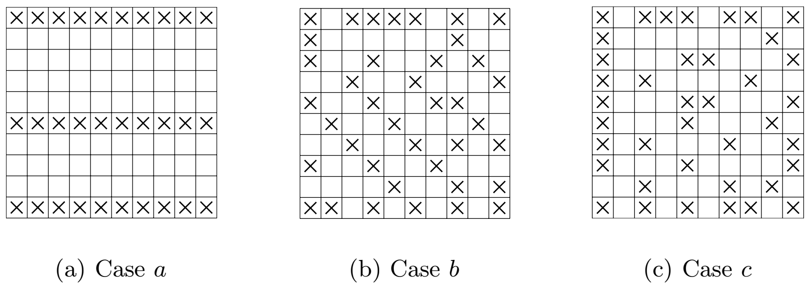

(a) Case *a* (b) Case *b* (c) Case *c*

Fig. 2. Optimal wind farm configurations of Grady's study [3]

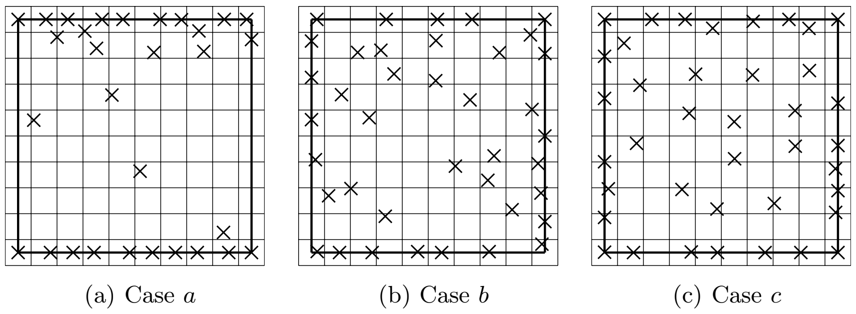

(a) Case *a* (b) Case *b* (c) Case *c*

Fig. 3. Optimal wind farm configurations of the present study

In Fig. 3, the area in the black thick square represents the effective domain of the wind farm. The turbine positions in the present study have been adjusted within the boundary of the wind farm effective domain. Since there are position constraints in the optimization, the turbines have been positioned to satisfy the constraints on minimum distances. In Case *a*, the turbines assemble on the upper and lower boundary of the wind farm, which is in coincidence with the uniform wind condition. In Case *b* and Case *c*, since the wind is distributed among 36 wind directions, the turbines "scatter" around the outer perimeter of the site.

Table 1 compares the wind farm performances by the two methods. Since the positions of turbines of the present study could be adjusted more freely, the wind speed loss has been further reduced. The total power output $\mathcal{P}$ and the wind farm efficiency η are respectively improved by 6.34%, 4.13% and 4.02% in three cases. It is obvious that wake effects in the wind farm, as well as energy loss, have been greatly reduced by optimizing the positions of turbines in the continuous space, and the wind resource could be more effectively exploited. However, since the number of turbines of the latter two cases are larger than that of the first case and the wind conditions are more complex, their relative free space to adjust is lower than that of the first case. As a result of that, the relative improvement is also lower than the first case.

Table 1. Comparison of optimal results

	Caes *a*		Caes *b*		Caes *c*	
	$\mathcal{P}$ (kW)	η (%)	$\mathcal{P}$ (kW)	η (%)	$\mathcal{P}$ (kW)	η (%)
Grady's study	14312	92.03	17241	85.28	31649	85.54
Present study	15220	97.87	17953	88.80	32921	88.98
Improvement	6.34%		4.13%		4.02%	

5 Conclusions

This paper introduces a novel approach to optimize the positions of turbines in the continuous space. The goal is to extract the maximal energy when the wind resource, topography and the number and types of turbines are given. The key question to solve the formulated problem is the large number of constraints and the high complexity of the optimization problem. A particle swarm optimization algorithm based on a penalty-function method is employed to solve the present problem. Simulation results demonstrate that more energy could be extracted from the wind farm by the proposed approach under the same conditions.

In the application of the penalty-function method, the selection of penalty coefficient is problem dependent and time consuming. Determining an appropriate value of penalty coefficients is itself a difficult optimization problem. It will be of importance to propose an adaptive fitness-evaluation method in the future work.

Acknowledgements

This work was supported in part by the National High Technology Research and Development Program of China (863 Program) under Grant No. 2007AA05Z426 and the Natural Science Foundation of China under Grant No. 60674096.

References

1. Ammara, I., Leclerc, C., Masson, C.: A viscous three-dimensional differential/actuator-disk method for the aerodynamic analysis of wind farms. Solar Energy Engineering 124, 345–356 (2002)
2. Mosetti, G., Poloni, C., Diviacco, B.: Optimization of wind turbine positioning in large wind farms by means of a genetic algorithm. Wind Engineering Industrial Aerodynamic 51(1), 105–116 (1994)
3. Grady, S.A., Hussaini, M.Y., Abdullah, M.M.: Placement of wind turbines using genetic algorithms. Renewable Energy 30(2), 259–270 (2005)
4. Marmidis, G., Lazarou, S., Pyrgioti, E.: Optimal placement of wind turbines in a wind park using monte carlo simulation. Renewable Energy 33(7), 1455–1460 (2008)

5. Wan, C., Wang, J., Yang, G., Li, X., Zhang, X.: Optimal micro-siting of wind turbines by genetic algorithms based on improved wind and turbine models. In: Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference, Shanghai, P.R. China, pp. 5092–5096 (2009)
6. Wan, C., Wang, J., Yang, G., Zhang, X.: Optimal siting of wind turbines using real-coded genetic algorithms. In: Proceedings of European Wind Energy Association Conference and Exhibition. Marseille, France (2009)
7. Eberhart, R.C., Shi, Y.: Particle swarm optimization: developments, applications and resources. In: Congress on Evolutionary Computation 2001, Kennedy (2001)
8. Schutte, J.F., Groenwold, A.A.: A study of global optimization using particle swarms. Global Optimization 31(1), 93–108 (2005)
9. Jensen, N.O.: A note on wind turbine interaction. Tech. rep. Riso National Laboratory, Denmark (1983)
10. Katic, I., Hojstrup, J., Jensen, N.: A simple model for cluster efficiency. In: Proceedings of European Wind Energy Association Conference and Exhibition, Rome, Italy, pp. 407–410 (1986)
11. Coello Coello, C.A.: Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Computer Methods in Applied Mechanics and Engineering 191(11-12), 1245–1287 (2002)
12. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the 1995 IEEE International Conference on Neural Networks, Perth, Australia, vol. 4, pp. 1942–1948 (1995)
13. Shi, Y., Eberhart, R.: A modified particle swarm optimizer. In: Conference on Evolutionary Computation, Anchorage, USA, pp. 69–73 (1998)
14. Eberhart, R.C., Shi, Y.: Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of the 2000 Congress on Evolutionary Computation, California, USA, vol. 1, pp. 85–88 (2000)
15. Charlisle, A., Dozier, G.: An off-the-shelf pso. In: Proceedings of the Workshop on Particle Swarm Optimization. Purdue School of Engineering and Technology, Indianapolis, USA (2001)

PSO Applied to Table Allocation Problems

David A. Braude[1,2] and Anton van Wyk[2]

[1] Information Security, Modeling and Digital Sciences, CSIR, Pretoria, South Africa
david.braude@students.ee.wits.ac.za

[2] School of Electrical and Information Engineering, University of the Witwatersrand, Johannesburg, South Africa
anton.vanwyk@wits.ac.za

Abstract. Table allocation is a type of assignment problem. The aim of table allocation is to assign multiple people to a single table in such a way that it minimizes a cost function. While particle swarm optimization (PSO) is normally used for continuous variables it has been adapted to solve this problem. Each particle represents an entire seating arrangement, and the velocity is the amount of times people swap tables during each iteration. In an example application PSO shows a significant improvement in fitness compared to the initial conditions, and has a low runtime. It also performs better in fitness improvement and runtime compared to choosing as many random samples as PSO generated. The use of PSO allows for generalized cost functions, and is simple to implement.

1 Introduction

At many functions or events it is necessary to seat people at a table according to certain constraints. These may include keeping apart certain people, or ensuring that certain people are together. Another important consideration is the maximum and minimum size of a table. For instance at a wedding it is desirable to keep family members together and to keep people who dislike one another apart. Other examples as to what may be taken into consideration are the gender ratios, the age of the people, and the languages that the guests speak.

Table allocation as a separate problem has not been formally defined. However, it is very similar to the assignment problem[1]. The analogy would be that a table is a task and that a person would be the agent. The difference in the two problems is that inherently multiple agents are assigned to one task, and it is the nature of the interaction of the agents that creates the cost. For the purposes of this paper, the definition of the table allocation problem will be: To assign all people to tables in such a way as to minimize costs and satisfy rules. To accommodate cases where people can be unassigned the creation of a no table allocation could be used.

It is clear that as the amount of types of costs rise the difficulty of assigning people to tables becomes increasingly more complicated. In this paper, Particle Swarm Optimization(PSO)[2] is used to deal with the table allocation problem. The primary purpose of this methodology is to decouple the process from the calculation of the 'goodness' or fitness of a table. Thus if more cost types are added then the rest of the optimization is not affected in terms of computational efficiency. It also becomes easier to modify or add constraints.

Neither linear nor quadratic programming can be used to solve the table allocation problem in general because the cost function can assume any form. Population based methods are applicable because of the general structure of the fitness function. This study focuses on PSO, however, a Genetic Algorithm (GA) can also be applied[2]. The most appropriate crossover function is not readily apparent, thus PSO is easier to adapt to the problem. GA, unlike PSO, is memoryless. It is theorized that in this problem memory will enhance the final solution.

The No Free Lunch Theorem[3] is a general heuristic which implies that some search methods are better suited to certain problems. To test if PSO is an efficient solution for the table allocation problem, the example will be compared to the process of randomizing a valid seating arrangement and then calculating the cost. Each iteration represents generating as many new seating plans as the PSO swarm size and keeping the best arrangement. Both methods are averaged over 100 trials. The PSO has higher computational costs and thus to be considered a viable solution to the table allocation problem it should have a large increase in performance. If PSO cannot exceed the performance of the random approach then its computational costs are not justified, and hence it cannot be considered a reasonable methodology.

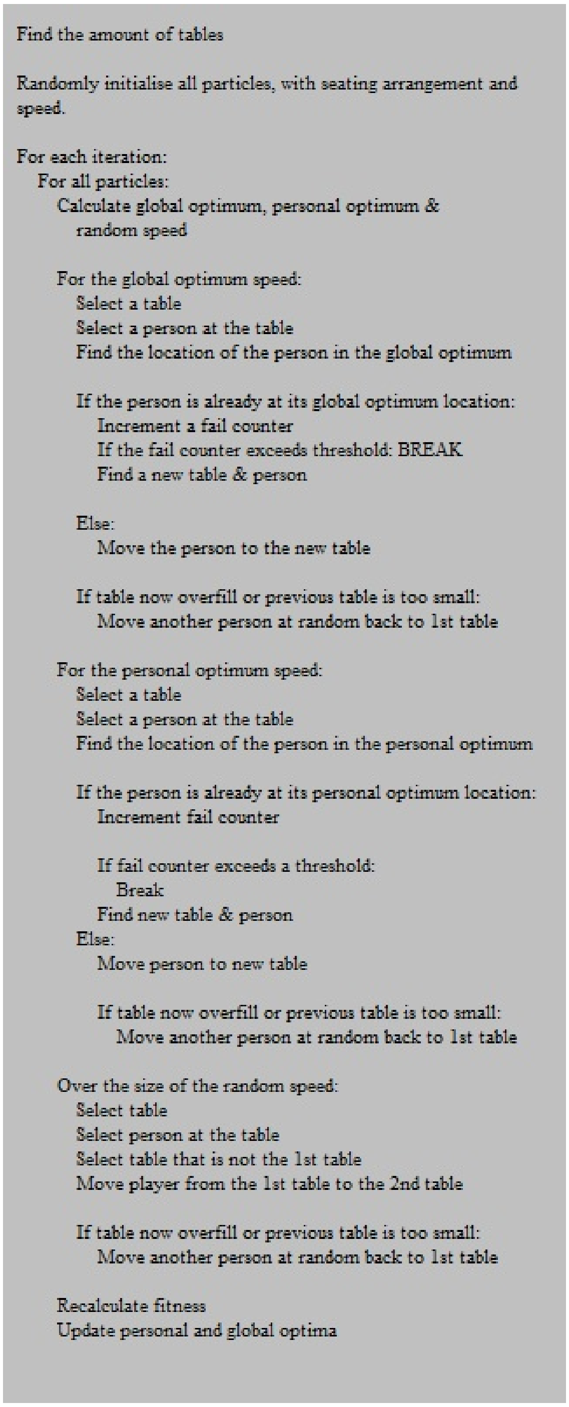
The rest of the paper is structured as follows: The general algorithm is defined; an example is used to test performance; the key points are summarized in the conclusion.

2 General Algorithm

PSO was first introduced by James Kennedy and Russell Eberhart in 1995[4]. In this technique, each particle at each iteration moves in three different ways, referred to as speeds - towards the global optimum; towards its personal optimum; and randomly. Speed is defined relative to the application. Each speed is determined by previous states - it will increase the movement toward the global optimum if the current iteration is highly dissimilar; movement toward its own optimum is determined the same way; and its random movement is proportional to the total speed of the previous iteration. Each speed is also scaled by a constant. The speeds due to the global optimum and the personal optimum are also multiplied by a uniform random number on the unit range[2].

There is a modification where the particles will work within neighborhoods[2]. This will only add to the complexity and not the comprehension of the proposed solution to the problem. Hence it was not used for this study.

In the case of table allocation each particle represents a possible way of seating the people. The movement is done by moving people from one table to another. In the table allocation problem movement towards optima means moving people to the table they are assigned to in the optimum. Through experimentation (an



```
Find the amount of tables

Randomly initialise all particles, with seating arrangement and
speed.

For each iteration:
  For all particles:
    Calculate global optimum, personal optimum &
      random speed

    For the global optimum speed:
      Select a table
      Select a person at the table
      Find the location of the person in the global optimum

      If the person is already at its global optimum location:
        Increment a fail counter
        If the fail counter exceeds threshold: BREAK
        Find a new table & person

      Else:
        Move the person to the new table

      If table now overfill or previous table is too small:
        Move another person at random back to 1st table

    For the personal optimum speed:
      Select a table
      Select a person at the table
      Find the location of the person in the personal optimum

      If the person is already at its personal optimum location:
        Increment fail counter

        If fail counter exceeds a threshold:
          Break
        Find new table & person
      Else:
        Move person to new table

        If table now overfill or previous table is too small:
          Move another person at random back to 1st table

    Over the size of the random speed:
      Select table
      Select person at the table
      Select table that is not the 1st table
      Move player from the 1st table to the 2nd table

      If table now overfill or previous table is too small:
        Move another person at random back to 1st table

    Recalculate fitness
    Update personal and global optima
```

Fig. 1. PSO Table Allocation Algorithm

example is shown in Section3) it was found that the best order was to move towards the global optimum, then the personal optimum, and then randomly. An important constraint on the movement is that a table does not get overfilled or have too few people. To counter this problem if moving an person to a new table causes the table capacity to be exceeded, or causes the source table to hold too few people, a random person at the destination table is moved to the source table. In this way, table capacity constraints are preserved through updating scenarios. The PSO algorithm for table allocation is shown in Fig. 1.

Fail thresholds were introduced to stop the update process from going into an infinite loop. The speed is initialized randomly with a normal distribution. In the example application, in 3% of the trails, the swarm converged on a local optimum, within five iterations, when using uniform random numbers. The use of the normal distribution stopped this problem by ensuring that most particles did not have near zero initial speeds. Details of the advantages of various distributions are given in[5]. An acceptable range of the speed was empirically determined by running the algorithm once with arbitrary values and setting the range to be approximately double the mean speed. In the example application the range was 0 to 40 movements.

2.1 Parallel Processing

PSO has many components that can be processed in parallel[6]. The initialization of the particles is embarrassingly parallel. The update is almost also embarrassingly parallel, with the exception that after each update the global solution must be shared. To accommodate this sharing, when implementing the optimization, it was efficiently achieved by flagging particles that improve on the global optimum as candidates. The fitness of the candidates is then bubble sorted with only one iteration so the fittest is known[6]. This also makes it easy to use the multi-elitist modification to PSO proposed in[7]. Because the example is to be used as a benchmark the modification was not implemented.

3 Example Application

In South Africa there are a number of conventions for Role Playing Games (RPGs). At these conventions there are two categories of participants: players and Game Masters (GM). Participants may enter in teams of up to seven people, or individually.

During the course of the convention the ideal would be that participants are:

1. Never seated with each other twice
2. Team mates and family members are kept apart
3. There is mix of experience at a table
4. There is an ideal ratio of gender
5. There is an ideal ratio of players to GM's

Table 1. Cost Table

Description	*Cost*
Rejoining of the players with each other	15 per Round
Rejoining of the players with a GM	35 per Round
Family members at the same table	100 per violation
Team mates players at the same table	300 per violation
Players at the same table as a team GM	1000 per violation
Incorrect ratios of players to a GM	15 per Player
Incorrect ratio of genders	5 per Player
Experience variation	0.05

The importance of each criterion is different and all of the above may be changed between games. A typical cost matrix is given in Table 1. The experience costs represent a multiplier for the standard deviation. A negative multiplier then favors grouping players together which have similar experience.

3.1 Test Conditions

The test conditions for the example are as follows (the complete data set is available on request):

1. There are 65 participants
2. 9 participants are registered as GM's, 8 male, 1 female
3. 12 participants have registered to either play or be a GM, 8 male, 4 female. During the allocation process their role becomes fixed, there are no players that are concurrently GM's.
4. There are 10 female players and 34 male players
5. There are 4 full teams of 7
6. There have been 2 previous rounds (which were allocated at random)
7. There are 4 families the sizes are 2, 3, 4, and 5
8. The size of the table must be between 4 to 6 players, and there must be a single GM
9. Ideally there are 6 players, 2 female, 4 male at each table
10. The experience was uniformly randomized for each player between 0 and 100

3.2 PSO Implementation Details

The PSO has a swarm size of 30 and was run to 50 iterations, these are typical of PSO[2] and kept it to a reasonable run time. The global optimum scaling factor was 0.9, the personal optimal scaling factor was 0.3, and the random speed scaling factor is 0.1. The scaling factors where chosen using[8] as a guide and finalized empirically. The initial speed was a uniform random number between 0 and 50. And the failure threshold was 50. The initial speed and failure threshold was found empirically.

4 Results

Using Stackless Python2.5.4 on Intel 2GHz Core2 Duo a notebook computer, the time it took to initialize the swarm was 5.8s. The time per iteration was 4.15s. Thus the total run time for PSO was 213.3s. The random approach described in the introduction took 5.8s for initialization and per iteration, as it is the same process as initializing the PSO. Thus the total run time was 290s which is longer than the PSO. Per iteration it takes the random approach 1.39 times longer than PSO.

In the following, lower fitness values are better. Fig. 2 shows the mean fitness per iteration of both approaches. While both tend towards convergence, PSO converges faster to a usually better fitness.

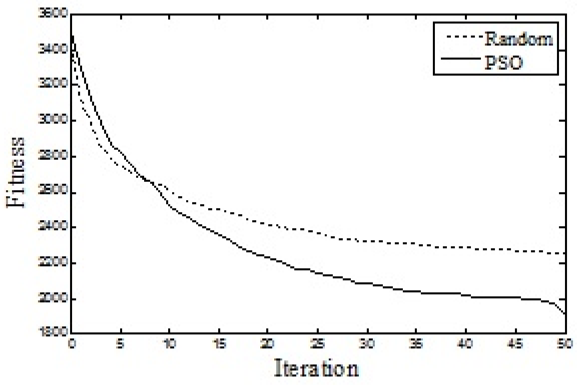


Fig. 2. Fitness v. iteration

The important statistics are given in Table 2. While they both start with the same fitness, the PSO outperforms the random approach by a relatively large margin, in both mean and minimum measures. The mean final fitness of PSO is in fact lower than the minimum final fitness of the random approach, which implies that on average the PSO will do better than the best that the random approach is capable of, however, this is theorized to be an anomaly as the random approach could generate the same table.

Table 2. Fitness Comparisons

	Cost	*PSO*
Mean Initial Fitness	3498.2	3465.43
Minimum Initial Fitness	2088	2011
Mean Final Fitness	1903.93	2243.36
Minimum Final Fitness	1201	1925
Time per iteration	4.15s	5.8s
Total run time	213.3s	290s

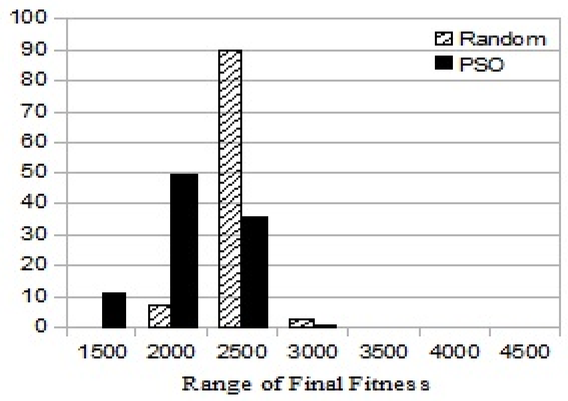


Fig. 3. Histogram of final fitness

The histogram in Fig. 3 shows the bulk of the PSO results are the same or better than 90% of the random approach results. Less than 5% of the PSO results were worse than the bulk of the random approach results. Which shows that PSO will reliably outperform the random selection

5 Conclusion

The table allocation problem and the assignment problem are very similar. The Particle Swarm Optimization(PSO) approach developed here could be adapted for the assignment problem.

Unlike other approaches PSO is simple to implement and can handle generalized fitness functions. Movement within this application of the PSO is representative of swapping allocations.

In the example PSO outperforms randomized results in every criteria. Thus PSO is considered to be a valid approach to solving the table allocation problem.

References

1. Shmoys, D.B., Tardos, E.: An Approximation Algorithm for the Generalized Assignment Problem Mathematical Programming, vol. 32, pp. 461–474 (1993)
2. Kennedy, J., Eberhart, R.C.: Swarm Intelligence. Morgan Kaufmann, San Francisco (2001)
3. Wolpert, D.H., Macready, W.G.: No Free Lunch Theorems for Optimization. IEEE Transactions on Evolutionary Computation 1(1), 67–82 (1997)
4. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proceedings of the IEEE International Conference on Neural Networks, November 1995, pp. 1942–1948 (1995)
5. Krohling, R.A., dos Santos Coelho, L.: PSO-E: Particle Swarm with Exponential Distribution. In: 2006 IEEE Transactions on Evolutionary Computation, July 2006, pp. 1428–1433 (2006)

6. Hunt, A., Thomas, D.: The Pragmatic Programmer. Addison-Wesley Professional, Indianapolis (1999)
7. Das, S., Abraham, A., Konar, A.: Automatic Kernel Clustering with a Multi-elitist Particle Swarm Optimization Algorithm. Pattern Recognition Letters 29(5), 688–699 (2008)
8. Shi, Y., Eberhart, R.C.: Parameter Selection in Particle Swarm Optimization. In: Porto, V.W., Waagen, D. (eds.) EP 1998. LNCS, vol. 1447, pp. 591–600. Springer, Heidelberg (1998)

Finding the Maximum Module of the Roots of a Polynomial by Particle Swarm Optimization

Liangdong Qu and Dengxu He

College of Mathematics and Computer Science,
Guangxi University for Nationalities,
Nanning, China
quliangdong@163.com

Abstract. After the theorem which is used to determine whether all roots of a polynomial are in unit circle is given, and two particle swarm optimizations for finding the maximum module of the roots of a polynomial based on the theorem are proposed. Finally, several computer simulation results show that using these algorithms to find the maximum module of roots of a polynomial are more efficient and feasible, the convergent speed is much faster and the accuracy of results is much higher.

Keywords: polynomial, the maximum module, unit circle, particle swarm optimization, parallel.

1 Introduction

In many theories and practical engineering applications, we often need to estimate the maximum module of the roots of polynomials, such as analyzing the stability of control system. The traditional estimation methods are mainly Cauchy estimate, Carmichael-Mason estimate [1], Farmer-Loizou estimate [2], Kittaneh estimate [3] and so on. However their accuracy is not high. In [4], a new type of iterative algorithm is given, the accuracy has increased, but not only more complicated calculation, but also not has parallel.

Particle swarm optimization (PSO) [5] is a population-based, self-adaptive search optimization method motivated by the observation of simplified animal social behaviors such as fish schooling, bird flocking, etc. It is becoming very popular due to its simplicity of implementation and ability to quickly converge to a reasonably good solution [6, 7].

This paper presents two particle swarm optimizations for finding the maximum module of the roots of a polynomial. These algorithms can actualize parallel in the feasible space and converge to the global optimal solution, which efficiently overcome the defect that the traditional methods can not parallelly compute in engineering technique. Several computer results show that they are more efficient and feasible.

2 The Basic Theories of Polynomial

2.1 Polynomial Deformation

Suppose that $f(x) \in C[x]$ with $\deg(f(x)) = n$) as follows:

$$f(x) = a_0x^n + a_1x^{n-1} + \cdots + a_{n-1}x + a_n, \; a_0 \neq 0 \tag{1}$$

Suppose that r is a real number. Let $x = ry$. Then, polynomial (1) can be deformed as follows:

$$f(x) = f(ry) = g(y) = a_0(ry)^n + a_1(ry)^{n-1} + \cdots + a_{n-1}(ry) + a_n \tag{2}$$

Let $a_k r^{n-k} = b_k$ ($k = 0, 1, \cdots, n$). Then, polynomial (1) can be deformed as follows:

$$f(x) = g(y) = b_0 y^n + b_1 y^{n-1} + \cdots + b_{n-1} y + b_n \tag{3}$$

If let $r = \max\{|x_k| \,|\, f(x_k) = 0, k = 1, 2, \cdots, n\}$. Then, the maximum module of the roots of the deformed polynomial (3) is 1, i.e., for polynomial (3), its roots are all in the unit circle (where in the unit circle included on the circle). Therefore, as long as finding the minimum parameter r makes the roots of polynomial (3) are all in the unit circle, and then we can find the maximum module of the roots of polynomial (1).

2.2 Module Bounds Selection

Now suppose that $r_{lower}^{(0)}$ and $r_{upper}^{(0)}$ are two real numbers. They are the lower and upper bounds of the maximum module of roots of polynomials (1), respectively. Let the lower bound $r_{lower}^{(0)} = 0$. For the upper bound of the selection, there are many ways [8], here we choose $r_{upper}^{(0)} = 1 + \max\{|a_k| / |a_0| \,|\, k = 1, 2, \cdots, n\}$.

2.3 Decision Theorem

Let $g^*(y) = \overline{b_n} y^n + \cdots + \overline{b_1} y + \overline{b_0}$. (4)

Easy to know that the roots of polynomial (3) and polynomials (4) are inverse points on the unit circle each other. We first introduce the following lemmas:

Lemma 1. [9]: If the roots of polynomial (3) are all in the unit circle, then $|b_n| < |b_0|$.

By Lemma 1, we can get Lemma 2:

Lemma 2. If $|b_n| \geq |b_0|$, then the roots of polynomial (3) are incomplete in the unit circle.

Let

$$\begin{aligned}
g_0(y) &= g(y) = b_0 y^n + \cdots + b_{n-1} y + b_n, \\
g_0^*(y) &= g^*(y) = \overline{b_n} y^n + \cdots + \overline{b_1} y + \overline{b_0}, \\
g_k(y) &= b_{k,0} y^{n-k} + \cdots + b_{k,n-k-1} y + b_{k,n-k}, \\
g_k^*(y) &= \overline{b_{k,n-k}} y^{n-k} + \cdots + \overline{b_{k,1}} y + \overline{b_{k,0}}
\end{aligned}$$

$$g_{k+1}(y) = \begin{cases} \dfrac{g_k(y)}{y} & b_{k,n-k} = 0 \\ \overline{\dfrac{b_{k,0} g_k(y) - b_{k,n-k} g_k^*(y)}{y}} & b_{k,n-k} \neq 0 \end{cases} \tag{5}$$

where $k = 0,1,\cdots,m-1, m \le n$. Obviously, when $k = 0$, there $b_{0,j} = b_j$ ($j = 0,1,\cdots,n$). From (5), we can construct $g_k(y)$.

Lemma 3. $\deg(g_k(y)) = n-k, (k = 0,1,\cdots m, m \le n)$.

Lemma 4. Suppose that N_k is the number of the roots of the polynomial $g_k(y)$ $(k = 0,1,\cdots,m)$ in the unit circle. Then, $N_0 = N_m + m$, $N_{k+1} = N_k - 1$. Where $k = 0,1,\cdots,m-1$.

Theorem (decision theorem). For a given polynomial (3), constructed sequence (5), then for polynomial (3), its roots are all in the unit circle if and only if $m = n$.

3 Particle Swarm Optimization

Particle swarm optimization provides a method for finding the global optimum of a function of several real variables. As in the case of genetic algorithms it starts with a population of potential solutions. The population is called a swarm and the members are called particles. The particles belong to a group which may be the population as a whole or may be a subpopulation. The particles change (evolve, learn) over time in response to their own experience and the experience of the other particles in their group. As this interpretation suggests, the principles underlying the algorithm can be applied not only to the solution of optimization problems, but also to the representation of social behavior, including economic behavior.

A swarm consists of Np particles moving around in an n-dimensional search space. The ith particle at the tth iteration has a position $x_i^{(t)} = (x_{i1}, x_{i2}, \cdots, x_{in})$, a velocity $v_i^{(t)} = (v_{i1}, v_{i2}, \cdots, v_{in})$, the best solution achieved so far by itself (pbest) $pb_i = (pb_{i1}, pb_{i2}, \cdots, pb_{in})$. The best solution achieved so far by the whole swarm (gbest) is represented by $pg = (pg_1, pg_2, \cdots, pg_n)$. The position of the ith particle at the next iteration will be calculated according to the following equations:

$$v_i^{(t+1)} = w \cdot v_i^{(t)} + c_1 \cdot r_1 \cdot (pb_i - x_i^{(t)}) + c_2 \cdot r_2 \cdot (pg - x_i^{(t)}), \tag{6}$$

$$x_i^{(t+1)} = x_i^{(t)} + v_i^{(t+1)}, \tag{7}$$

where c_1 and c_2 are two positive constants, called cognitive learning rate and social learning rate, respectively; r_1 and r_2 are two separately generated uniformly distributed random numbers in the range [0,1]; w is inertia weight factor. Empirical results showed the linearly decreased setting of inertia weight can give a better performance, such as linearly decreases from 1.4 to 0, and 0.9 to 0.4 through the search process (LDPSO) [10]. In addition, the velocities of the particles are confined within $[-v_{\max}, v_{\max}]$. If an element of velocities exceeds the threshold $-v_{\max}$ or $v_{\max}$, it is set equal to the corresponding threshold.

4 The Algorithms for Finding the Maximum Module of the Roots of a Polynomial by PSO

4.1 The Principle of Finding the Maximum Module of the Roots of a Polynomial by PSO

Because finding the maximum module of the roots of polynomial (1) can be converted finding the minimum parameter r makes the roots of polynomial (3) are all in the unit circle, this is an optimization problem, and we can use PSO.

4.2 Fitness Function Design

In PSO, fitness function plays an important role. It is used to evaluate the advantages and disadvantages of particles. In this particular problem on finding the maximum module of the roots of polynomial (1), we evaluate the advantages and disadvantages of two particles r_1 and r_2 are following:

Suppose that $r_1 \le r_2$. Then,

(i) If both r_1 and r_2 make that the roots of polynomial (3) are all in the unit circle, and then r_1 is superior to r_2.

(ii) If both r_1 and r_2 make that the roots of polynomial (3) are incomplete in the unit circle, and then r_2 is superior to r_1.

(iii) If both (i) and (ii) can not be satisfied, then if $(r_1 + r_2)/2$ make that the roots of polynomial (3) are all in the unit circle, then, r_1 is superior to r_2, otherwise, r_2 is superior to r_1.

4.3 The Process of Finding the Maximum Module of the Roots of a Polynomial by PSO

Here, we present two methods for finding the maximum module of the roots of a polynomial by PSO: one is using LDPSO directly and the other is improved LDPSO (ILDPSO). Their steps are illustrated as follows:

4.3.1 Finding the Maximum Module by LDPSO

Step1. Setting evolution parameters;

Step2. Initializing search space $[r_{lower}^{(0)}, r_{upper}^{(0)}]$ based on section II

Step3. Initializing the position vector and velocity vector of each component of each particle in $[r_{lower}^{(0)}, r_{upper}^{(0)}]$, determining the personal best location and the best location among the entire swarm;

Step4. Calculate the weight value in current iteration according to Eq. (8):

$$w = w_{\max} - iter \cdot \frac{w_{\max} - w_{\min}}{maxiter}, \tag{8}$$

where $w_{\max}$ and $w_{\min}$ are the maximum value and minimum value for weights, respectively, $iter$ is the current iteration, and $maxiter$ is the maximum iteration;

Step5. Evaluate all particles according to subsection 2 in Section 4, and derive the best particle of current iteration, and the best position that every particle has reached so far.

Step6. Update velocity and position of every particle according to Eq. (6) and (7).

Step7. If the stop criteria are satisfied, output the obtained best solution and the algorithm is ended; otherwise, go to Step 4.

4.3.2 Finding the Maximum Module by ILDPSO

In the method of the above, we only improve Eq. (7) in Step6 as follows:

If $x_i^{(t)}$ make that the roots of polynomial (3) are all in the unit circle, then

$$x_i^{(t+1)} = x_i^{(t)} - | v_i^{(t+1)} |,$$

otherwise

$$x_i^{(t+1)} = x_i^{(t)} + | v_i^{(t+1)} |.$$

These make particles searching direction is much clearer. Its convergence speed is much faster than LDPSO.

5 Simulation Experiments

Several experiments have been conducted to evaluate the performance of the proposed algorithms. In the algorithms, the number of particles of the swarm Np was set at 50 and the maximum number of iterations $maxiter$ was set at 100. Other parameters were set at $w_{\max} = 1.4$, $w_{\min} = 0$, $c_1 = 2$, $c_2 = 2$ and $v_{\max} = 2$. The experiment conditions are: Intel(R) Core (TM) 2 Duo 2.20GHz CPU, 1G Memory, and Windows XP operation system. The programs are realized in Matlab7.

Example 1: Let find the maximum module of the roots of the polynomial $f(x) = x^6 - 9x^5 - 16x^4 - 8x^3 - 5x - 7$. Its exact solution is 10.58363864836489. Using basic line decreases PSO (LDPSO), the result is 10.583638648408163, and using improved line decreases PSO (ILDPSO), the result

Table 1. Experimental results for Example 1

Methods	the Maximum Module r	Error $\mid r - r^* \mid$
Carmichael-Mason	21.81	11.226361351635109
Kittaneh	21.7	11.116361351635110
Farmer-Loizou	18	7.416361351635111
Cauchy	17	6.416361351635111
Method 1 in Literature[11]	15.4	4.816361351635111
Method 2 in Literature[11]	14.554	3.970361351635111
Method 1 in Literature[12]	12	1.416361351635111
Method 2 in Literature[12]	11.045	0.461361351635110
Literature[4]	10.58353	1.086483648897740e-004
LDPSO	10.583638648408163	4.327382896462950e-011
ILDPSO	10.583638648364893	3.552713678800501e-015

is 10.583638648364893. In Table1, the results of several methods were given. From Table1, We can see that using the two proposed methods in this paper, the accuracy of results is very high.

In order to more intuitively see convergence process of the maximum module of the roots of Example 1 by PSOs, we gave the error evolution curve in Fig.1. In Fig.1, the horizontal axis is the number of iterations; the vertical axis is the common logarithm value of the error. From Fig.1, we can see that the algorithms for finding the maximum module of the roots of a polynomial convergence speeds are very fast. ILDPSO is superior to LDPSO.

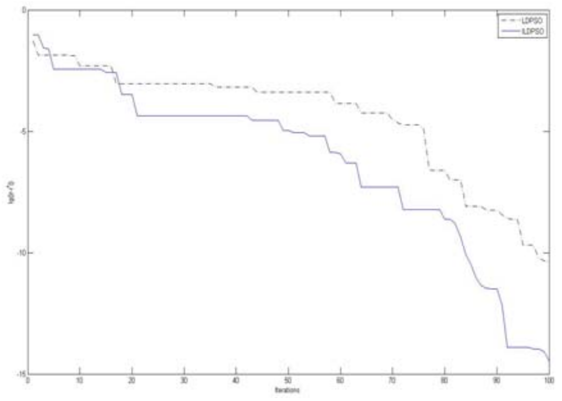


Fig. 1. Error evolution curve for Exmple 1

Example 2: Let find the maximum module of the roots of the polynomial $f(x) = x^{100} + 1$. Its exact solution is 1. Table2 and Fig.2 show the results by LDPSO and ILDPSO.

Table 2. Experimental results for Example 2

Methods	**the Maximum Module** r	**Error** $\mid r - r^{*} \mid$
LDPSO	0.999999999986375	1.362499002510731e-011
ILDPSO	0.999999999999994	5.773159728050814e-015

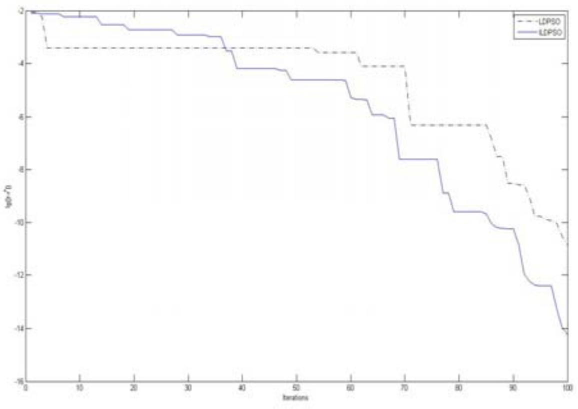


Fig. 2. Error evolution curve for Exmple 2

From Table2 and Fig.2, we can see that for high-order polynomial, LDPSO and ILDPSO convergence speed quickly and the accuracy is also very high.

6 Conclusions

In this paper, we present two PSOs for finding the maximum module of the roots of a polynomial such as group search and global convergence. The algorithms avoid the trivial deviate operation in finding the maximum module by traditional methods and overcome the defect that the convergence of traditional methods has much relationship with the selection of initial values, whose parallelism satisfies the need of parallel finding the maximum module in engineering technique. Experimental results show that the algorithms can converge to the best solution, and they have high accuracy, high convergence speed and parallelism.

Acknowledgments. This work is supported by the Natural Science Foundation of Guangxi of China (No. 0728054).

References

1. Horn, P.A., Johnson, C.H.: Matrix Analysis. Cambridge University Press, Cambridge (1990)
2. Farmer, M.R., Loizou, G.: Locating Multiple Zeros Interactively. Comput. Math. Appl. 11(6), 595–603 (1985)
3. Kittaneh, F.: Singular Values of Companion Matrices and Bounds on Zeros of Polynomials. Society for Industrial and Applied Mathematics (1995)
4. Guan, Y.: An iterative arithmetic for the largest module of roots of a polynomial. Journal of Hangzhou Teachers College (Natural Science Edition) 4(1), 22–24 (2005) (in Chinese)
5. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, pp. 1942–1948 (1995)
6. Shen, H.Y., Peng, X.Q., Wang, J.N., Hu, Z.K.: A mountain clustering based on improved PSO algorithm. In: Wang, L., Chen, K., S. Ong, Y. (eds.) ICNC 2005. LNCS, vol. 3612, pp. 477–481. Springer, Heidelberg (2005)
7. Eberhart, R.C., Shi, Y.: Extracting rules from fuzzy neural network by particle swarm optimization. In: Proceedings of IEEE International Conference on Evolutionary Computation, Anchorage, Alaska, USA (1998)
8. Ralston, A., Will, H.S.: Mathematical Methods for Digital Computer. John Wiley & Sons Inc., Chichester (1960)
9. Cheng, J.S.: A Parallel Algorithm for Finding Roots of Complex Polynomial. J. of Comput. Sci. & Technol. 5(1), 71–81 (1990)
10. Shi, Y., Eberhart, R.C.: A modified particle swarm optimizer. In: Proceedings of the IEEE International Conference on Evolutionary Computation, Anchorage, Alaska, USA, pp. 69–73 (1998)
11. Song, Y.Z.: The existence the region of polynomial zeros. ACTA Mathematic Sinica 36(2), 254–258 (1993) (in Chinese)
12. Zhao, W.J.: Bounds for the zeros of polynomials. Journal of Qingdao University 13(2), 14–19 (2000) (in Chinese)

Research on the Ant Colony Optimization Algorithm with Multi-population Hierarchy Evolution

Xuzhi Wang[1], Jing Ni, and Wanggen Wan

[1] School of Communication and Information Engineering, Shanghai University, Shanghai 200072, China
wangxzw@shu.edu.cn

Abstract. The ant colony algorithm (ACA) is a novel simulated evolutionary algorithm which is based on observations to behavior of some ant species. Because of the use of positive feedback mechanism, ACA has stronger robustness, better distributed computer system and easier to combine with other algorithms. However, it also has the flaws, for example mature and halting. This paper presents an optimization algorithm by used of multi-population hierarchy evolution. Each sub-population that is entrusted to different control achieves respectively a different search independently. Then, for the purpose of sharing information, the outstanding individuals are migrated regularly between the populations. The algorithm improves the parallelism and the ability of global optimization by the method. At the same time, according to the convex hull theory in geometry, the crossing point of the path is eliminated. Taking advantage of the common TSPLIB in international databases, lots of experiments are carried out. It is verified that the optimization algorithm effectively improves the convergence rate and the accuracy of reconciliation.

Keywords: Traveling Salesman Problem (TSP), ACOA, Multi-population Hierarchy, Individual Migration, Eliminating-cross.

1 Introduction

Complexity science has risen in the mid-80s. And the main research is about the theory of complex system, such as natural phenomena, engineering, biology, management, social, etc.. Many problems need to be processed with the natural characteristics of combinatorial optimization in the complex social system. For instance, how to guide vehicles to find way, how to effectively allocate radio frequency link, and so on. It is shown that the scale of mathematical model of these practical problems is so large that the accurate solution can not be completed within an acceptable time. However, these problems still must be effectively solved. Therefore, the sub-optimal solution need to be studied, which is an acceptable solution should be searched within an acceptable period of time.

It is found that a series of bionic algorithm belong to the scope of computational intelligence. According to the calculating process and outcome, it can spontaneously adjust algorithm parameters to achieve optimal results for solving the problem. As the swarm intelligence can accomplish the task which is so hard that a number of individual insects can not independently complete. This algorithm has no special requirements to the objective function, so it is especially applied to solve the optimization problems of large-scale complex structure. And ACA is a novel bionic algorithm which is based on observations to foraging behavior of the ant species. It has already demonstrated excellent performance in resolving the optimization of complex systems. It is widely applied in vehicle scheduling, traveling salesman problem, job-shop scheduling, route planning, power systems, wireless networks, data mining, image processing, and so on. For instance, multicast routing is solved with multi-constraints QoS in the ad hoc network [1]. Network bandwidth resources can be saved and network overload can be reduced in point-to-multipoint network communications. [2] proposed a novel team algorithm approach based on ACA for the resolution of a set of multi-objective optimization problems. And robustness and balance are achieved. And the test points are optimally selected in analog circuit design [3]. This can greatly reduce the test cost by eliminating redundant measurements. Moreover, [4-5] introduced the application in power forecasting and robot control. At present, ACA has become an international research focus and cutting-edge topics in the field of intelligent computing.

However, ACA also has the defects, such as trapped in a local optimal solution, searched for a long time, etc.. In response to these shortcomings, in recent years, many scholars at home and abroad have done a lot of research work to improve the performance of ACA. Dorigo M firstly proposed an improved ant colony system (AS) [6], which only updated the pheromone of the shortest path in each cycle. Thus the feedback of optimal pheromone is strengthened. And German scholars Stutzle T and Hoos H proposed the max-min ant system (MMAS) [7]. Through the limit of the value of the upper and lower of pheromone, pheromone in all paths is approximation and the divergence of the algorithm can be avoided. A new approach to update the pheromone trails, denominated learning levels, is presented and used to solve the vehicle routing problem [8]. [9] proposed a novel cloud-based fuzzy self-adaptive ant colony system (CFSACS), in which cloud model is used as the fuzzy membership function and a self-adaptive mechanism is constructed. According to the advantages of the combination with other algorithms, the hybrid algorithms are presented with particle swarm optimization (PSO) and genetic algorithm (GA) [10-11].

Meanwhile, this paper presents an optimization algorithm with multi-population hierarchy evolution (MHEACA). Each sub-population, which is entrusted to different control parameters, carries out respectively a different search independently. In view of the purpose of sharing information, the outstanding individuals are migrated regularly between the populations. Through the method, the parallelism and the ability of global optimization should be improved.

2 Ant Colony Algorithm [12-13]

Italian scientist Dorigo M. etc. firstly presented the heuristic algorithm – ACA in 1991, which is inspired by real ants foraging in natural world. One of the problems studied by ethologists was to understand how almost blind animals like ants could manage to establish shortest route paths from their colony to feeding sources and back. It was found that the medium used to communicate information among individuals regarding paths, and used to decide where to go, consists of pheromone trails.

Through the pheromone, the entire ant colony finally finds the optimal path. Meanwhile, the colony can also adapt to changes in the environment. And the ants can quickly skip the obstacles on the path and find the shortest path again.

ACA is the model derived from the study of real ant colonies. It is not interested in simulation of ant colonies, but in the use of artificial ant colonies as an optimization tool. The basic formula is as follows.

$$p_{ij}^{k}(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}\bullet[\eta_{ik}(t)]^{\beta}}{\sum\limits_{s\subset allowed_k}[\tau_{is}(t)]^{\alpha}\bullet[\eta_{is}(t)]\beta}, & \text{if } \mathrm{j}\in \mathrm{allowed}_{\mathrm{k}}\\ 0, & \text{otherwise}\end{cases} \tag{1}$$

Where, $p_{ij}^{k}(t)$ is the probability from city i to city j ; $\tau_{ij}(t)$ is the intensity of pheromone trail; $\eta_{ij}(t)$ is the visibility of edge (i, j) ; $allowed_k$ is the set of cities to be visited by ant k positioned on city i ; α and β are two parameters that tunes the relative importance of trail versus visibility. Ants routing tendency can be adjusted by choosing the value of α and β . And n is the city amount. Meanwhile, in order to satisfy the constraint that an ant visits all the n different towns, each ant is associated with a data structure called the tabu list, that saves the towns already visited up to time t and forbids the ant to visit them again before n iterations have been completed.

3 The Multi-population Optimization Algorithm

If swarm intelligence is effective, it should have the capability of local and global exploration. ACA has demonstrated its own advantages and disadvantages. In general, for improving the convergence rate, the role of positive feedback should be necessarily enhanced, so that the pheromones of a few superior paths rapidly increase. Moreover, for improving halting, it is important that the role of positive feedback is reduced. At the same time, this encourages ants to explore new paths and expands the scope of the choose path. So it is obvious that improvement of convergence speed and reduction of halting influence each other. Therefore, it is necessarily that the balance between convergence speed and the exploration of solution space is discovered.

And symbiotic evolution, which refers to biological phenomenon is living together and interdependent, is a beneficial way of life. Moreover, the information of exchanging and sharing between individual and individual, or between individuals and groups, often exist in nature. Thus, based on the symbiotic strategy, this paper proposes the assumptions of optimization model - multi-population hierarchy evolution. The MHEACA is composed of several separate sub-populations which can independently evolve in parallel. After the evolution in several generations, the information will be migrated between sub-populations. And all sub-populations can share optimal information.

The researchers have been paid more and more attention to collaborative work between populations. And the idea has been applied in GA [14-15]. Here, through the interaction between multiple ant colony, it can further enhance time performance and effectively improve the stagnation.

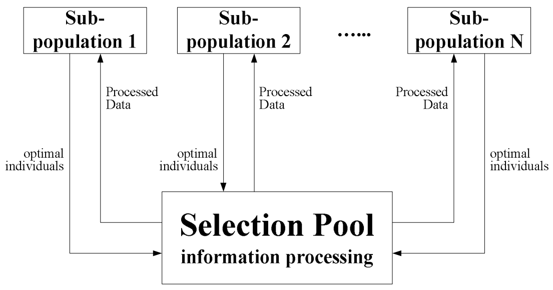


Fig. 1. Framework of information exchange between sub-populations

According to the different parameter settings, each sub-population can separately search by different movement pattern. It will be as much as possible to explore the solution space in the evolutionary process, so that the algorithm escapes from local optimum to global search. Through the selection pool, which is used to save the optimal individuals, all populations exchange information and improve the performance.

In order to effectively expand search space of populations, the size of selection pool is determined according to the scale of sub-populations. Through specific exchange mechanism, the change of information within a sub-population can rapidly lead to the response of other sub-populations. Thus the optimal information is selected and the performance of algorithm is ensured.

3.1 Information Exchange Mechanism

It is shown that the mechanism of information exchange is a key in the MHEACA. According to the analysis about the principle of ACA, how to properly deal with positive feedback is the focus of improvement of the performance. From the above, it is interaction between the convergence speed and local optimal solution by the change of positive feedback. Moreover, the idea of multi-population hierarchy evolution is just a reference to solve this problem. Firstly, the definition of optimal solution should be accurately understood. For different specific problems, the answer is different. In order to get rid of the phenomena of early-maturing and stagnation, it is necessarily that the influence of local optimal value must be reduced in ACA. Therefore, a simple method, which is weighted average method, is adopted as the mechanism of exchange between populations. That is to say, the local optimal individual of each sub-population is took out, and then the weighted average of these values is again assigned to original sub-population. Thus the algorithm can expand the scope of search and it will not be trapped into local optimal solution. The formula is as follows:

$$F_{selection} = \frac{Sub_1 * K_1 + Sub_2 * K_2 + \cdots\cdots + Sub_{N-1} * K_{N-1} + Sub_N * K_N}{N} \tag{2}$$

Where $F_{selection}$ is the final average result in selection pool; Sub_N $(N = 1,2,\cdots,n)$ is the value of optimal individuals in each sub-population; K_N $(N = 1,2,\cdots,n)$ is the weight; N is the number of sub-populations.

3.2 Parameters Selection

So the information is exchanged by pheromone and the convergence should be a dynamic process. It is extremely important that the parameters are properly selected, which will directly affect the final performance of algorithm. At present, the optimal selection of parameters is yet not a theoretical basis. The values are set based on experience in most conditions.

Given the specific issues for the traveling salesman problem (TSP), $\eta_{ij}(t)$ is computed from distance D_{ij} between cities, that is

$$\eta_{ij}(t) = \frac{1}{D_{ij}} \tag{3}$$

Obviously, the smaller the distance, the greater the expectation $\eta_{ij}(t)$. Secondly, the trail levels are updated as follows. Through analysis to nature, part of the pheromone trail evaporates, i.e. existing trails are reduced before new trails are laid. This is regulated by a parameter ρ. On basis of these updated trail levels，the next iteration $t+1$ can be started. The updating equation of track pheromone concentration as follows:

$$\tau_{ij}(t+n) = (1-\rho) \bullet \tau_{ij}(t) + \Delta\tau_{ij}(t) \tag{4}$$

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t) \tag{5}$$

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{Q}{L_k}, & \text{if k - th ant uses edge (i, j) in its tour} \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where, ρ is the parameter to regulate the reduction of τ_{ij}; $1-\rho$ is used to stand for the information eclipse degree; $\Delta\tau_{ij}$ is the total increase of trail level on edge (i, j); Q is the quantity of pheromone laid by per-tour; L_k is the total length of the paths in current tour by ant k.

From above, it is very important that the scale of populations and migration cycle are properly set in MHEACA. If the scale is too large, the data are so much that the step of calculation becomes more intricacy and the speed of computation will drop. Meanwhile, if the migration cycle is smaller, it is favor to make the integration of sub-populations and spread best individual. But it can also increase the communication and reduce the diversity of individual. So the proper intervals should be considered. These values may not only draw lessons from the existing results of other scholars, but also be determined by the continuous adjustment of experiments.

3.3 Execution Steps

In order to prove the feasibility of the MHEACA, the algorithm is appropriately simplified about the number of populations. So there are only two sub-populations in this paper, A and B. The sub-population A is defined as elite colony which focuses on improving of the convergence speed. Only the pheromones of a few optimal paths are allowed to update in each iteration. And the more outstanding ants, the more pheromone releases. Thus the positive feedback of algorithm is enhanced so as to improve execution speed. The other sub-population B, which is defined as detection colony, is used to expand the scale of search space and provide new super-plane and the diversity of individual.

Firstly, at the 1st level, two sub-populations of MHEACA independently parallel compute by the implementation steps of ACA. After several iterations, the more excellent individual can be searched out and saved. Secondly, at the 2nd level, the information is shared and migrated in the selection pool at the same interval. The selected outstanding individuals are deal with by weighted average method. And the result is again assigned to original sub-population. Thus the local optimal can be balanced and the performance of algorithm can be improved. Finally, if the precision requirement or the number of iteration is met, the global optimal solution is get. The specific flowchart of MHEACA is as follows.

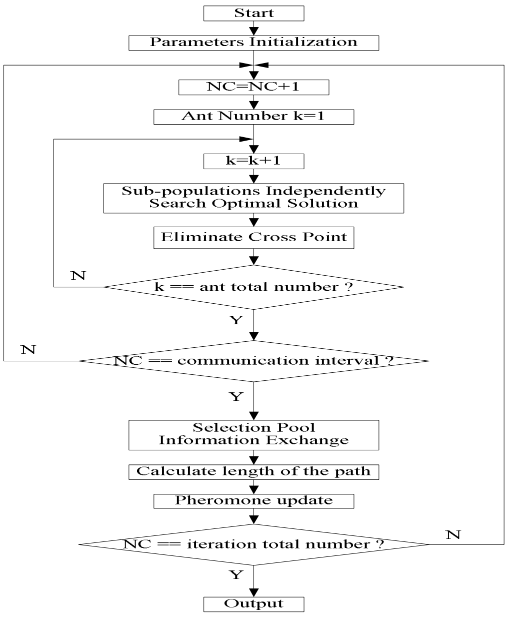

Start
Parameters Initialization
NC=NC+1
Ant Number k=1
k=k+1
Sub-populations Independently Search Optimal Solution
Eliminate Cross Point
N
k == ant total number ?
Y
N
NC == communication interval ?
Y
Selection Pool Information Exchange
Calculate length of the path
Pheromone update
N
NC == iteration total number ?
Y
Output

Fig. 2. The software flowchart of MHEACA

3.4 Eliminate Cross Point

It is inevitable that the experimental results may be cross. Therefore, it is necessary that the cross point must be eliminated between cities in order to form a regular closed curve. Convex hull vertex, which is one of the basic theories in geometry, is used to optimize the path. Here, the points of curve are pre-processed in TSP.

Suppose S is a non-empty set of points in a plane, z_1 and z_2 are any two points in S, t is an arbitrary real number between 0 and 1. If the formula (7) is satisfied, then the point z belongs S and S is convex.

$$z = tz_1 + (1-t)z_2 \qquad 0 \le t \le 1 \tag{7}$$

Where convex hull, which contains S, is the smallest convex polygon perimeter in convex set S. And convex hull must contain convex set. In the pre-processing algorithm for TSP, from the beginning of a vertex of convex hull to the next adjacent vertex, the two points need to be determined in the same straight line. If it is not, then this line will be eliminated. And so the vertices else of convex hull are similar treatment.

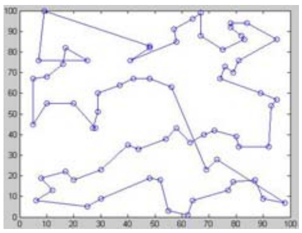
4 Experiment results

TSP is a typical combinatorial optimization problem in mathematics field. As ACA is very suitable for NP - problem, it has been applied to solve TSP [16]. The objective is to determine a minimum distance of a tour passing through each city once and only once. For some classic urban data in TSPLIB, such as berlin52, smith70 and so on, from the accuracy, robustness and execution time, MHEACA is compared with ACA. The specific data of parameter in MHEACA are as follows: $\alpha_A = 1$, $\beta_A = 3$, $\rho_A = 0.2$, $Q_A = 200$, $K_A = 1.2$, $\alpha_B = 1$, $\beta_B = 3$, $\rho_B = 0.2$, $Q_B = 200$, $K_B = 0.9$. Because of the same work, the number of ants is 30 in two sub-populations. In addition, the interval of communication is 10 and the total number of iterations is 100. Table 1 shows the optimal results of two algorithms. It is obvious that MHEACA has better performance, especially in the execution time.

Table 1. Comparison between MHEACA and ACA

Algorithm	Smith70			churritz150		
	Shortest Distance	Convergence Number	Time	Shortest Distance	Convergence Number	Time
ACA	728.85	45	1.436s	6967.8	30	6.551s
MHEACA	713.83	65	0.722s	6834.5	56	2.226s

At the same time, the ability of planning routes is also compared. When the scale of cities is relatively smaller, the route of ACA is well. However, as the increase in the number of cities, the route of ACA gradually has some cross point and becomes non-standard. Through the cross pre- processing, the redundant cross points are effectively eliminated. Fig.3 (a) and Fig.4 (a), which are the results by ACA, have more cross points. However, through eliminating cross, the result is standard closed curve in Fig.3 (b) and Fig.4 (b).


(a) Result by ACA

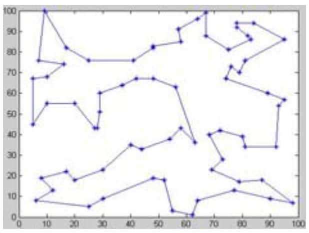
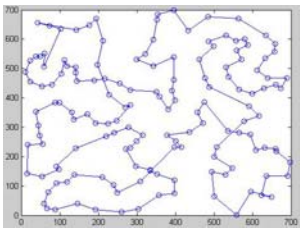

(b) Result by MHEACA

Fig. 3. The path curve with smith70


(a) Result by ACA

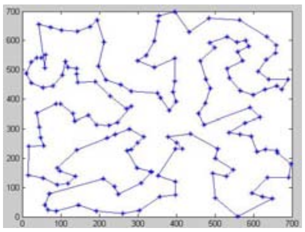

(b) Result by MHEACA

Fig. 4. The path curve with churritz150

5 Conclusion

In summary, this paper proposes a MHEACA based on the social division and symbiotic evolution in natural world. According to the different search purpose, many sub-populations respectively parallel search optimal individual. And through migration strategy, the useful information is transmitted at regular intervals. So these individuals of next generation can inherit more information from the ancestors and the neighboring sub-populations by sharing. Moreover, because of the adoption of migration operator, the performance of algorithm is improved. At the same time, through lots of experiments in TSPLIB, it is verified that the MHEACA has better parallelism, stability and global optimization. Compared with single population, this method can effectively improve the convergence speed, maintain the local characteristics of sub-populations and avoid the premature phenomenon. According to the actual need, the number of sub-population may be appropriately expanded and the information exchange mechanism of selection pool can be changed. It is obvious that the MHEACA has good prospects for engineering applications. It is better able to solve the engineering optimum design and provide a new idea for the optimal solution of complex structure.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No.60902086) and the Leading Academic Discipline Project of Shanghai Municipal Education Commission (No.J50104).

References

1. Wang, H., Shi, Z.: An Ant Colony Algorithm Based on Orientation Factor for QoS Multicast Routing in Ad Hoc Networks. In: Third International Conference on Communications and Networking in China, pp. 321–326 (2008)
2. Lezcano, C., Pinto, D., Barán, B.: Team Algorithms Based on Ant Colony Optimization – A New Multi-Objective Optimization Approach. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 773–783. Springer, Heidelberg (2008)
3. Zhang, C.-j., He, G., Liang, S.-h.: Test Point Selection of Analog Circuits Based on Fuzzy Theory and Ant Colony Algorithm. In: IEEE AUTOTESTCON 2008, Salt Lake City, UT, pp. 164–168 (2008)
4. Li, W., Han, Z.-h., Li, F.: Clustering Analysis of Power Load Forecasting based on Improved Ant Colony Algorithm. In: Proceedings of the 7th World Congress on Intelligent Control and Automation, Chongqing, China, pp. 7492–7495 (2008)
5. Gao, M., Xu, J., Tian, J.: Mobile Robot Global Path Planning Based on Improved Augment Ant Colony Algorithm. In: Second International Conference on Genetic and Evolutionary Computing, pp. 273–276 (2008)
6. Dorigo, M., Gambardella, L.M.: A study of some properties of Ant-Q. In: Ebeling, W., Rechenberg, I., Voigt, H.-M., Schwefel, H.-P. (eds.) PPSN 1996. LNCS, vol. 1141, pp. 656–665. Springer, Heidelberg (1996)
7. Thomas, S., Holger, H.H.: MAX-MIN ant system. Future Generation Computer Systems, 889–914 (2000)
8. Laura Cruz, R., Juan, J., Gonzalez, B., Orta, J.F.D., et al.: A new approach to improve the ant colony system performance: Learning levels. In: Corchado, E., Wu, X., Oja, E., Herrero, Á., Baruque, B. (eds.) HAIS 2009. LNCS, vol. 5572, pp. 670–677. Springer, Heidelberg (2009)
9. Li, Z., Wang, Y., et al.: A Novel Cloud-based Fuzzy Self-adaptive Ant Colony System. In: Fourth International Conference on Natural Computation, pp. 460–465 (2008)
10. Xin, Z., Yu-zhong, Z., Ping, F.: An Improved Ant Colony Algorithm. MultiMedia and Information Technology, 98–100 (2008)
11. Nonsiri, S., Supratid, S.: Modifying Ant Colony Optimization. In: IEEE Conference on Soft Computing in Industrial Applications, Japan, pp. 95–100 (2008)
12. 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)
13. Dorigo, M., Caro, G.D., Gambardella, L.M.: Ant algorithms for discrete optimization. Artificial Life, 137–172 (1999)
14. Chang-chun, D., Ru-ming, Z., Yong-xia, L., Bo, X.: A Multi-Group Parallel Genetic Algorithm for TSP. Computer Simulation, 187–190 (2008)
15. Liu, X.-j., Huang, G.-l., Lin, Z.-x., Guo, W.-h.: Interaction Force Coefficients Estimation of Ship Maneuvering Based on Multi-Population Genetic Algorithm. Journal Of Shanghai Jiaotong University, 945–948 (2008)
16. Zhang, X., Tang, L.: A New Hybrid Ant Colony Optimization Algorithm for the Traveling Salesman Problem. In: Huang, D.-S., Wunsch II, D.C., Levine, D.S., Jo, K.-H. (eds.) ICIC 2008. LNCS (LNAI), vol. 5227, pp. 148–155. Springer, Heidelberg (2008)

Graph Partitioning Using Improved Ant Clustering

M. Sami Soliman and Guanzheng Tan

School of Information Science and Engineering
Central South University
Changsha 410083, P.R. China
mssas_sami@yahoo.com, tgz_csu@yahoo.com.cn

Abstract. Parallel computing, network partitioning, and VLSI circuit placement are fundamental challenges in computer science. These problems can be modeled as graph partitioning problems. A new Similarity carrying Ant Model (SCAM) is used in the ant-based clustering algorithm to solve graph partitioning problem. In the proposed model, the ant will be able to collect similar items while it moves around. The flexible template mechanism had been used integrated with the proposed model to obtain the partitioning constrains. Random graph has been used to compare the new model with the original ant model and the model with short-term memory. The result of the experiments proves the impact of the SCAM compared with other models. This performance improvement for ant clustering algorithm makes it is feasible to be used in graph portioning problem.

Keywords: graph portioning, ant-based clustering, similarity, template.

1 Introduction

Clustering data sets into disjoint groups is a problem arising in many domains. From a general point of view, the goal of clustering is to find groups that are both homogeneous and well separated, that is, entities within the same group should be similar and entities in different groups dissimilar. These data sets can represent graphs, where nodes correspond to the entities to be clustered and edges correspond to a similarity measure between those entities. The partitioning problem arises in many areas of computer science, like parallel computing, network partitioning, and VLSI circuit placement. In particular, the partitioning problem models the placement of data onto a multiprocessor computer with distributed memory, where the computation load has to be balanced among the processors and the amount of communication has to be minimized [1], [2] and [3].

The graph partitioning problem is NP-hard for general graphs, as well as for bipartite graphs, and even finding good approximation solutions for general graphs or arbitrary planar graphs is NP-hard [4]and [5]. Using traditional methods, it may not result in an optimum solution of practical problems. There exist many variations of this approach aimed at improving its performance. However, these techniques do not present a universal solution to partitioning problems since there are certain graphs for which each version of these methods performs poorly [6]. Heuristic algorithms can help us to find a better optimum solution with reasonable computation.

Ant-based clustering is a heuristic clustering method that draws its inspiration from the behavior of ants in nature [7]. In particular, it models the clustering and sorting that can be observed in some ant species: where real ants gather corpses to clean up their nest, or transport larvae to order them by size, artificial ants transport data items that are laid out in an artificial environment, and arrange them in a sorted fashion.

2 Graph Partitioning Problem Statement

Let $G = \{V, E\}$ be a graph, where $V = \{v_i, i = 1, \cdots, N\}$ is the set of N vertices and E the set of edges. We assume that G is non-directed, that is, E is symmetric. Let the adjacency matrix be denoted by $A = [a_{ij}]; a_{ij} \neq 0$ if and only if $(v_i, v_j) \in E$. We shall only treat the cases where $a_{ij} = 0 \; or \; 1$, which correspond to $(v_i, v_j) \notin E$ and $(v_i, v_j) \in E$, respectively.

The graph partitioning problem on a graph G consists of dividing the vertices into disjoint subsets such that the number of edges whose endpoints are in different subsets is minimized. We consider the balanced partitioning problem, where the difference of cardinalities between the largest and the smallest subset is at most one.

The precise formulation of the usual problem of graph partitioning is to find k nonempty components such that the total inter-component weight is minimized. For arbitrary k, the decision problem is NP-complete, and for fixed k, a solution can be found in $O(|V|^{k^2})$ steps, where $|V|$ denotes the number of elements in V. If the additional constraint of having the same number of vertices in every cluster is added, the problem is NP-complete for fixed k [8].

3 Basic KLS Algorithm for Graph Partition Problem

The first ant-based clustering algorithm was introduced by Deneubourg et al. [9], to produce clustering behavior in simple groups of robots. Then, Kuntz, Layzell, and Snyers [10] and [11] (hereafter KLS) have proposed an interesting application of Lumer and Faieta's idea [10] to use ant clustering in graph partitioning. The advantage of using KLS instead of classical methods is handling for the most complex graphs without any more computation. For simplicity of the application we choose to use two- dimensions space. In this space the dissimilarity between two nodes in the graph $\delta(v_i, v_j)$ can be calculated by equation:

$$\delta(v_i, v_j) = \frac{\sum_{k=1}^{n} |a_{ik} - a_{jk}|}{\sum_{k=1}^{n} |a_{ik}| + \sum_{k=1}^{n} |a_{jk}|} \quad . \tag{1}$$

The original algorithm begins with an initial random placement of objects on the grid. Next, each ant picks up an object; the ant randomly selects an object v_i that is not being carried by another ant. The ant then computes a measure of similarity between v_i and its neighbor objects $f(v_i)$, and a pick-up probability $P_{pick}(v_i)$ based on the measure of similarity. The ant generates a random number $R \in [0,1]$. If $R < p_{pick}(v_i)$ the ant moves to the location of object v_i and picks up the object. Otherwise, the ant repeats the process. The measure of similarity or local density $f(v_i)$ is the summation of the difference between object v_i and all other objects in v_i's neighborhood L, it calculated as

$$f(v_i) = \max\left(0, \frac{1}{s^2}\sum_{d_j \in L}\left[1 - \frac{\delta(v_i v_j)}{\alpha}\right]\right) . \tag{2}$$

Object dissimilarity is scaled by the constant α, which is data dependent, and provides a threshold that separates similar objects from dissimilar objects. The neighborhood consists of the grid cells residing in a square centered on object v_i's current location. The probability of picking up an object, $P_{pick}(v_i)$ is given by

$$P_{pick}(v_i) = \left(\frac{k_1}{k_1 + f(v_i)}\right)^2 . \tag{3}$$

where the parameter $k_1 \in [0,1]$ controls the pick-up sensitivity. Once each ant is holding an object, the main loop of the algorithm begins. Each iteration of the main loop proceeds as follows. For all ants, using its current position on the grid, the ant computes the similarity measure, $f(v_i)$ for the object it is holding v_i, using equation (2). The ant generates a random number $R \in [0,1]$ and drops its object if $R < p_{drop}(v_i)$,where $p_{drop}(v_i)$ is calculated as

$$P_{drop}(v_i) = \left(\frac{f(v_i)}{k_2 + f(v_i)}\right)^2 . \tag{4}$$

If the cell is empty, the ant drops the object in its current cell. Otherwise, the ant does no thing. The drop probability proposed in [9] provides a smooth response similar to the probability of picking up an object. The drop probability $p_{drop}(v_i)$ is higher when the carried object is similar to those in the ant's neighborhood and lowers when dissimilar. The constant k_2 governs the drop sensitivity. If the ant dropped its object, the ant goes through the process of picking up another object. If the object was not dropped, then the ant randomly moves to another cell in the grid. Only a single agent and/or a single item are allowed to occupy any one cell at a time. Once all time steps are complete or some other termination condition has been met, the main loop ends and all ants drop their objects at their current locations.

The algorithm suffers from convergence problems and an unfavorable runtime behavior, and several attempts to overcome these limitations have therefore been proposed [12]. This ends the basic ant clustering algorithm that is the starting point for our enhancements in the next section.

4 Proposed Algorithm

In this section, we build upon the work of [12] to develop a general and robust version of ant-based clustering. In particular, we describe how template construction for the algorithm can be simply derived from the clustering constrains, and we introduce new model for the ant agents that improve the quality of the clustering.

By adding a template mechanism to the KLS algorithm particular problems can be solved. Inclusion of the template is straightforward. Equations (3) and (4) in section 3 are transformed into:

$$P_{pick}(v_i) = 0.7\left(\frac{k_1}{k_1 + f(v_i)}\right)^2 + 0.3(1 - P_t(x_i, y_i)) \cdot \tag{5}$$

$$P_{drop}(v_i) = P_t(x_i, y_i)\left(\frac{f(v_i)}{k_2 + f(v_i)}\right)^2 \cdot \tag{6}$$

Where x_i, y_i represent the location coordinates of vertex v_i on the grid. The template function -will be discussed later - allows us to solve the problem of partitioning the graph into known number of clusters. The proposed algorithm consists of setup phase and free-running phase. The setup phase represents 1% of the total number of iterations. At the setup phase, to make sure that the ants can discover the entire cluster regions; the dropping probability should be more affected with the template function. So the dropping probability will be as equation (7) at the beginning and then it will be change in the second phase to the original form as equation (6).

$$P_{drop}(v_i) = 0.5 \times P_t(x_i, y_i) + 0.5 \times \left(\frac{f(v_i)}{k_2 + f(v_i)}\right)^2 \cdot \tag{7}$$

4.1 The Template Function Construction

Some species of the ants which are found in nature, they utilize the information related to the temperature and humidity in their surroundings to build their nests and to distribute their brood. This concept of self-organizing with templates has been used in [7] for data analysis and graph partitioning. With such mechanisms, the result is that the final structures would closely follow the configuration defined by the templates. The template we have used in [13], in the form of a Gaussian Probability Surface (GPS) guides the multi-agents to form clusters within a plane working space.

We propose a universal template structure. The centroid of clusters is defined based on the user requirements and not dependent on the feature space. In order to achieve focal points for the template; a circle shape is chosen to be the working area. A circle can be easily partitioned into equal circular sector by dividing its 2π angle over the number of desired clusters. Then after getting a circular sector simply find the centroid for them. Using the geometric centroid for each sector as the center for the cluster; the probability function can be constructing for the whole templates using the GPS equation, $P_t(x, y)$ as shown in equation (8).

$$P_t(x, y) = P_{\max} \sum_{i=1}^{n} \left[e^{-\left(\frac{(x-\bar{x}_{oi})^2 + (y-\bar{y}_{oi})^2}{\sigma_i^2} \right)} \right] + \delta \cdot \tag{8}$$

where, $0 \le P_t(x, y) \le 1$. $P_{\max}$ is the maximum value of probability, $0 \le P_{\max} \le 1$. δ is offset value. $\bar{x}_{oi}, \bar{y}_{oi}$ are coordinates of the geometric centroid for each cluster. x, y are coordinates on any single point in the workspace. n is the number of clusters, $1 < n$.

Fig. 1 shows 50×50 grid example of the GPS model with four humps. As the value of the probability surface increases, the probability of dropping the item by the ant is higher. The balanced and pre-configured clusters has a importance for some applications of graph partitioning problem, such as design of VLSI circuits and dynamic load balancing for distributed systems.

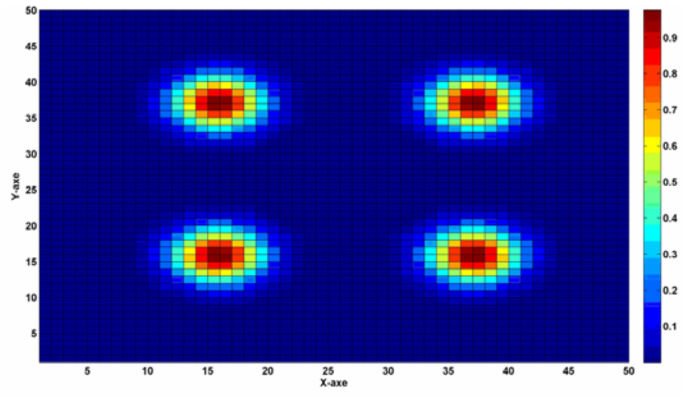


Fig. 1. Template functions

4.2 Similarity Carrying Ant Model (SCAM)

The ants clustering algorithm has been inspired from the biological life of ants, so the modification of it should be also inspired form their life. One of the well-known facts about ants, that an ant can lift 20-50 times its own body weight. So this fact has been used to enhance the performance of the ant clustering. In the proposed model for the ant, it will be able to carry more than one item while it moves around. The group

Algorithm 1. High-level algorithm for KLS-SCAM

```
/* Initialization */
Compute distance matrix $\delta(v_i, v_j)$ for all vertexes in the graph
Compute $P_t$ equations (8-12)
For every vertex $v_i$ Place $v_i$ randomly on grid End For
For all agents Place agent at randomly selected site End For
/* Main loop */
For $t$ = 1 to $t_{\max}$ do
    For all agents do
        If ((agent can carry more) and (site occupied by vertex $v_i$ )) then
            Compute $f(v_i)$ using equation (2), $P_{pick}(v_i)$ using equation
            (5) and $s(v_i)$ using equation (12)
            Draw random real number R between 0 and 1
            If ( $R < P_{pick}(v_i)$ ) then
                If (agent is empty  or (agent already carries items and
                $s(v_i) \le$ similarity threshold)) then
                        Pick up item $v_i$
                End if
            End If
        Else If ((agent carrying items) and (site empty)) then
            For every carried item $v_j$ do
              Compute $f(v_i)$ using equation (2)
              If ( $t < 0.01 \times t_{\max}$ )  //setup phase
               Compute $P_{drop}(v_i)$ using equation (7)
              Else
               Compute $P_{drop}(v_i)$ using equation (6)
              End If
            End for
               Draw random real number R between 0 and 1
               If ( $R < Max(P_{drop}(v_i))$ ) then
                  Drop item $v_i$ where $P_{drop}(v_i) = Max(P_{drop}(v_i))$
               End If
        End If
        Move to randomly selected neighboring site not occupied by other agent
    End For
End For
   For all agents Drop all carried items in agent location or nearest free location End For
   Mask clusters with the constructed template to get the final clusters
   Reconstruct the graph from the clusters of vertexes
```

picking and dropping behavior reduces the number of ant moves, with the purpose of reducing the time for the complete clustering process. Under SCAM, an ant has some probability of carrying more items. The decision is governed by the similarity of carried items and the candidate item. The similarity can be measured by the average distances between the candidate item and the carried items as

$$S(v) = \frac{\sum_{v_k \in C} \delta(v_k, v)}{N_c} . \qquad (9)$$

where C is the carried items set and N_c is the number of carried items by the ant.

If the similarity value $S(v)$ is less than similarity threshold, then the ant carries the nominated item with other items. The maximum number of elements that ant can carry simultaneously should be selected with realistic value based on the total number of the items. Also the similarity threshold should be extracted from the items intra-distances. The value of similarity threshold can be calculated by sorting the intra-distances between all items, then select the value of the item with index equal to 25% of the total number of the elements. In this paper, we compare the performance of the similarity carrying ant model to original ant model and the ant model having memory.

The proposed algorithm shown in Algorithm 1 is the basic ant clustering algorithm with our SCAM and template system extensions.

5 Experiment

Random graphs $F(n, c, p_i, p_e)$ have been used in the context of VLSI to test iterative improvement partitioning algorithms. [14]. The KLS algorithm was tested on these random graphs [7]. So, we will use the same graph in our experiments. Such graphs are composed of c clusters of n vertices, p_i is the probability that two vertices within a cluster are connected, and p_e is the probability that two vertices that belong to different clusters be connected. The proposed algorithm has been tested for different instance of random graphs and the results constantly insure its superiority. Due to the limited space we will include only one example. Fig. 2 (a) shows the initial random distribution of vertices on the plane for a random graph of the type RG (25, 4, 0.8, 0.01). Fig. 2 (b) illustrates the fact that the proposed algorithm is able to find "natural" clusters in a graph RG. It shows that, after $T = 2\times10^5$ iterations, vertices distribute themselves in space in such a way that exactly 4 clusters of vertices appear, which correspond to the four clusters of the graph. The original KLS can reach this result within $T = 2\times10^6$ according to [7]. KLS algorithm suffer from unfeasibility in the time needed to complete clustering. The new model takes only 10% of this time compared with the original KLS and achieves the same result.

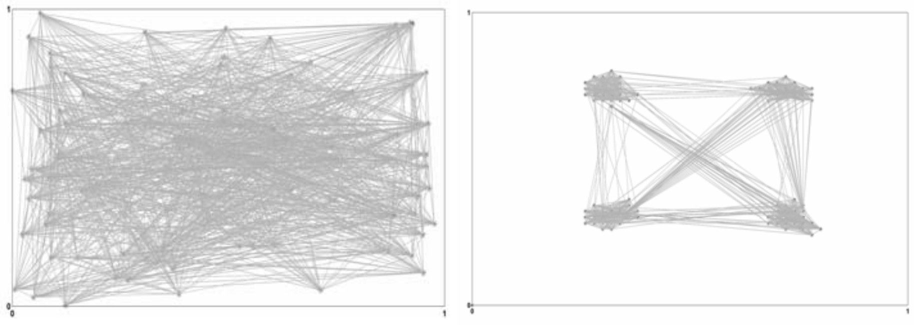


Fig. 2. (a)random graph of type F (25, 4, 0.8, 0.01),(b) the result after clustering

In order to evaluate the performance of the proposed SCAM model, as the comparison objects, other KLS algorithms, including the original KLS algorithm and the modified version with memory, were also used to portioning the same random graph RG. With the purpose to compare the quality of our clustering algorithm, two metrics are used to judge the clustering quality. These metrics are F-measure and The Adjusted Rand index. The same metrics has been used in [15]. The F-measure represents the reasonable balance between the precision and recall of the clustering for all classes. The F-measure is bound by the interval [0, 1], with higher values indicating a better clustering [16]. Also the Adjusted Rand Index (ARI) became one of the most successful cluster validation indices and in [17] it is recommended as the index of choice for measuring agreement between two partitions in clustering analysis. ARI is bound by the interval [0, 1] such as F-measure.

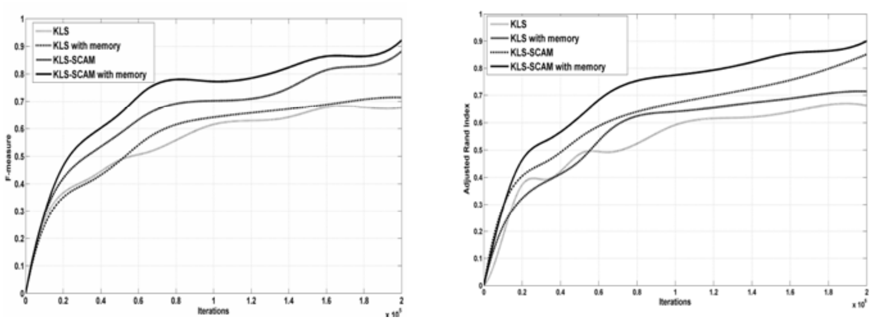
5.1 SCAM Model Performance Evaluation

For the all experiments, the same template been used. For all algorithms, the clustering is executed for 30 times for the same random graph RG and the clustering results are averaged for each of the metrics used. The average performance is created by calculating the mean of each metric every 500 iteration. All algorithms were executed using 10 agents within 50×50 grid space. The main algorithm parameters k_1, k_2, α and s^2 have values 0.3 ,0.1,1 and 9 respectively . These parameters are commonly used for KLS algorithm. The maximum number of items that ant can carry (for SCAM model) is three. The maximum memory length (for ants with memory model) is four. All simulations are implemented using Matlab 7.4.0 (R2007a).

The affect of merging the new model with other modification of KLS is experienced. We decide to examine the ants with memory because it seems to be the best modification for this KLS and it is one of the most attractive to be studied [24].

Fig. 3 shows the average for both metrics F-measure and ARI. It is clear that the new SCAM model for ants has a great effect on the performance of the KLS. Both metrics confirm that the clustering operation using the proposed model can achieve better clusters within less time. The effect of the memory feature on the performance of the original KLS is not so clear. The reason of limited improvement is nature of that effect; the role of the sort-term memory is to reduce the number of equivalent clusters which is constant in our case. In other hand, the SCAM feature is more effective; because it can reduce the number of tours which ants need to transport items. By merging SCAM with memory feature the result of clustering becomes better; which means that SCAM has no side effects on other KLS modification. . This performance improvement for KLS makes it is feasible to be used in graph portioning problem and its applications.

The template used also as has a role for the improvement of the clustering by fixing the number of clusters and separate them with sufficient distance. The most important effect of the template is the load balance insurance, which has importance mainly for graph partitioning problem.



(a) (b)

Fig. 3. The average values of (a) F-measure and (b) ARI

6 Conclusions

The graph partitioning problem is very important for such application as designing of VLSI circuits and dynamic load balancing for distributed systems. The need for finding good approximation solutions for it is NP-hard. One of the methods that had been used to solve this problem using ants is the KLS algorithm .The original KLS algorithm and its modification is not very efficient in terms of computation time. The new SCAM ant model is capable to achieve excellent result in acceptable computing time. The F-measure and ARI metrics demonstrate the performance enhancement that the new model is able to get compared with original model and its memory feature modification.

The new template construction method had very flexible features. It used minimal information about the involved clusters to construct the full template. The integration of the new model and the template has been done. This integration allowed the proposed algorithm to realize separated and balanced portions for the graph in practical computation time.

References

1. Teresco, J.D., Faik, J., Flaherty, J.E.: Hierarchical Partitioning and Dynamic Load Balancing for Scientific Computation. In: Dongarra, J., Madsen, K., Waśniewski, J. (eds.) PARA 2004. LNCS, vol. 3732, pp. 911–920. Springer, Heidelberg (2006)
2. Boman, E.G., Catalyurek, U.V., Chevalier, C., Devine, K.D., Safro, I., Wolf, M.M.: Advances in Parallel Partitioning, Load Balancing and Matrix Ordering for Scientific Computing. Journal of Physics: Conference Series 180(1), 12008–12013 (2009)
3. Kumar, S., Das, S.K.: Graph Partitioning for Parallel Applications in Heterogeneous Grid Environments. In: 16th International Parallel and Distributed Processing Symposium, IPDPS 2002, Lauderdale, Florida, USA, pp. 66–71 (2002)
4. Schaeffer, S.E.: Graph Clustering. Computer Science Review 1(1), 27–64 (2007)
5. Bui, T.N., Jones, C.: Finding Good Approximate Vertex and Edge Partitions Is NP-Hard. Information Processing Letters 42, 153–159 (1992)
6. Peiravi, A., Ildarabadi, R.: Complexities of Using Graph Partitioning in Modern Scientific Problems and Application to Power System Islanding. Journal of American Science 5(5), 1–12 (2009)
7. Bonabeau, E., Dorigo, M., Theraulax, G.: Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, New York (1999)
8. Garey, M.R., Johnson, D.S., Stockmeyer, L.: Some Simplified NP-Complete Graph Problems. Theoretical Computer Science 1, 237–267 (1976)
9. Deneubourg, J.L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., Chrétien, L.: The dynamics of collective sorting robot-like ants and ant-like robots. In: The first international conference on simulation of adaptive behavior on from animals to animats, pp. 356–363. MIT Press, Cambridge (1990)
10. Lumer, E., Faieta, B.: Diversity and Adaptation in Populations of Clustering Ants. In: Proceedings of Third International Conference on Simulation of Adaptive Behavior: From Animals to Animats, vol. 3, pp. 499–508. MIT Press, Cambridge (1994)
11. Kuntz, P., Layzell, P., Snyers, D.: A Colony of Ant-Like Agents for Partitioning in VLSI Technology. In: Husbands, P., Harvey, I. (eds.) Fourth European Conference on Artificial Life, pp. 417–424. MIT Press, Cambridge (1997)
12. Handl, J., Knowles, J., Dorigo, M.: Strategies for the increased robustness of ant-based clustering. In: Di Marzo Serugendo, G., Karageorgos, A., Rana, O.F., Zambonelli, F. (eds.) ESOA 2003. LNCS (LNAI), vol. 2977, pp. 90–104. Springer, Heidelberg (2004)
13. Ong, S.L., Lai, W.K., Tai, T.S.Y., Hoe, K.M.: Application of ant-based template matching for web documents categorization. Informatica 29, 173–181 (2005)
14. Garbers, J., Promel, H.J., Steger, A.: Finding Clusters in VLSI Circuits. In: IEEE International Conference on Computer-Aided Design, pp. 520–523. IEEE Computer Society Press, Los Alamitos (1990)
15. Peterson, G.L., Mayer, C.B.: Ant clustering with locally weighted ant perception and diversified memory. Swarm Intelligence 2(1), 43–68 (2008)
16. Handl, J.: Ant-based methods for tasks of clustering and topographic mapping: extensions, analysis and comparison with alternative methods. Masters Thesis, Chair of Artificial Intelligence, University of Erlangen-Nuremberg, Germany (2003)
17. Santos, J.M., Embrechts, M.: On the Use of the Adjusted Rand Index as a Metric for Evaluating Supervised Classification. In: Alippi, C., Polycarpou, M., Panayiotou, C., Ellinas, G. (eds.) ICANN 2009. LNCS, vol. 5769, pp. 175–184. Springer, Heidelberg (2009)

A Knowledge-Based Ant Colony Optimization for a Grid Workflow Scheduling Problem

Yanli Hu, Lining Xing, Weiming Zhang, Weidong Xiao, and Daquan Tang

College of Information System and Management
National University of Defense Technology
Changsha, 410073 P.R. China
smilelife1979@hotmail.com,
{xinglining,wmzhang,wdxiao,dqtang}@nudt.edu.cn

Abstract. Service-oriented grid environment enables a new way of service provisioning based on utility computing models, where users consume services based on their QoS (Quality of Service) requirements. In such "pay-per-use" Grids, workflow execution cost must be considered during scheduling based on users' QoS constraints. In this paper, we propose a knowledge-based ant colony optimization algorithm (KBACO) for grid workflow scheduling with consideration of two QoS constraints, deadline and budget. The objective of this algorithm is to find a solution that minimizes execution cost while meeting the deadline in terms of users' QoS requirements. Based on the characteristics of workflow scheduling, we define pheromone in terms of cost and design a heuristic in terms of latest start time of tasks in workflow applications. Moreover, a knowledge matrix is defined for the ACO approach to integrate the ACO model with knowledge model. Experimental results show that our algorithm achieves solutions effectively and efficiently.

Keywords: grid workflow scheduling, quality of service, ant colony optimization, knowledge model.

1 Introduction

Service-oriented grid computing enables users to consume services transparently over world-wide network environment based on their QoS requirements. Many grid applications require workflow processing in which interdependent tasks are executed based on their control or data dependencies. One of the most challenging problems in service-oriented grid computing is to map service instances to tasks in workflow applications to achieve users' various QoS requirements.

A workflow application consists of a collection of tasks that are processed in a specific order to accomplish a complicated goal. A workflow application is usually represented as a Directed Acyclic Graph (DAG), in which the nodes represent individual tasks and the directed arcs stand for precedence relations between the tasks. As scheduling in a DAG is NP-hard in general[1], approaches to grid workflow scheduling are proposed based on different heuristics[2-4], which are applicable to a set of parallel independent tasks based on the performance estimation for task execution.

Moreover, meta-heuristic approaches have been applied for solving workflow scheduling problems[5-7]. Since meta-heuristics provide both a general structure and strategy guidelines for developing a heuristic for solving computational problems, they are generally applied to a large and complicated problem, and provide an efficient way to achieve better performance. Recently, ant colony optimization (ACO) [8], a meta-heuristic in the light of the foraging behavior of ants, has been proposed to solve various intractable combinatorial optimization problems. With the advantage of making full use of instance-based heuristic information for scheduling problems, some grid workflow scheduling algorithms based on ACO have been proposed. However, most of them can only tackle the problems with a single QoS parameter or with small scale workflows.

Among various QoS requirements arising in practice, makespan and cost of workflow applications are of particular concern to users. Although traditional scheduling algorithms achieved some interesting results, most approaches figure out grid workflow scheduling by minimizing makespan. In "pay-per-use" grids, workflow execution cost must be considered during scheduling based on users' QoS constraints. In fact, approaches that minimize execution cost while meeting the deadline of workflow applications are of users' interests.

In this paper, a knowledge-based ant colony optimization (KBACO) algorithm for grid workflow scheduling is proposed with consideration of two QoS constraints, deadline and budget. Our proposed approach aims to find solutions that minimize execution cost while meeting the deadline of workflow applications.

This paper is organized as follows. In Section 2, the framework of knowledge-based ant colony optimization approach for grid workflow scheduling is proposed, and the KBACO algorithm is described. In Section 3, computational experiments are reported. Some concluding remarks are made in Section 4.

2 A Knowledge-Based ACO Approach for a Grid Workflow Scheduling Problem

In this section we first give the formal definition of a grid workflow scheduling problem. Second, we propose a framework of knowledge-based ant colony optimization approach. Next the implementation of KBACO algorithm for grid workflow scheduling is described. We then illustrate how to improve solutions effectively. Finally, rules are designed for knowledge learning for scheduling.

2.1 Problem Formulation

We model a workflow application as a DAG $G=(V, E)$ as follows.

(1) Let V be the finite set of tasks T_i ($1 \le i \le n$) of a workflow application, and E the set of directed arcs of the form (T_i, T_j), where T_i is called a parent task of T_j, and T_j is called a child task of T_i. We denoted by $Pred(T_i)$ and $Succ(T_i)$ the set of parent tasks and the set of child tasks of T_i, respectively.

(2) Let S_i be the service set consisting of service instances $\{s_i^1, s_i^2, \cdots, s_i^{m_i}\}$ available for implementing task T_i. Each service instance s_i^j is associated with a label

$(s_i^j.id, s_i^j.t, s_i^j.c)$, where $s_i^j.id$ is the unique *id* indicating the service instance s_i^j, $s_i^j.t$ and $s_i^j.c$ indicates the capacity and the cost per unit of s_i^j, respectively.

(3) Let D be the time constraint (deadline) specified by users for workflow execution. That is, the total execution time of the workflow must not be larger than D.

We assume *wlog* that setting up time in the scheduling is negligible. In a feasible solution each task T_i must be allocated to one available service instance s_i^j in S_i.

2.2 Framework of KBACO for Grid Workflow Scheduling Problem

Based on the knowledge-based heuristic searching architecture[9], we proposed a knowledge-based ACO algorithm as shown in Fig. 1, which can be briefly sketched as follows.

(1) The genotypic knowledge was initialized;
(2) A group of feasible solutions were constructed using ACO algorithm guided by the existing knowledge;
(3) The genotypic knowledge was updated by the optimization of current iteration.
(4) These processes were repeated till the stopping criterion is satisfied.

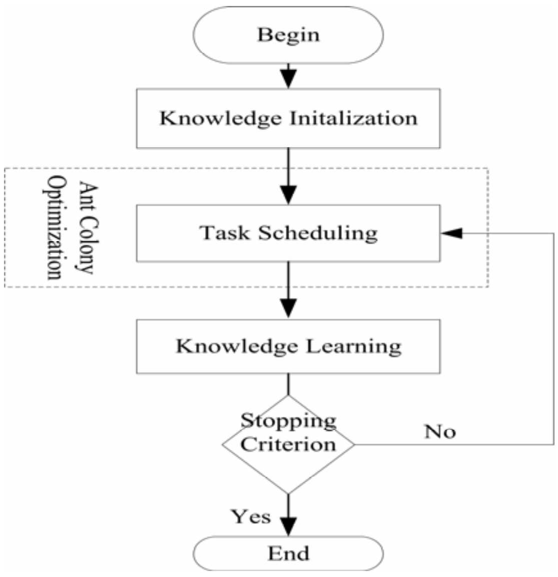


Fig. 1. The framework of the Knowledge-Based Ant Colony Optimization for Grid Workflow Scheduling

In KBACO, an ACO algorithm is applied to solve workflow scheduling with assistance of accumulated knowledge of scheduling.

(1) Elite solution knowledge. It consists of a fixed number of selected best solutions. In the initialization phase, the elite solution knowledge will be initialized

as null. In the knowledge learning phrase, the elite solution knowledge will be updated by the best solution in each iteration.

(2) Task scheduling knowledge (TSK). It is the accumulative knowledge of mapping service instances to tasks in workflow applications. A knowledge matrix TSK with size $|V| \times |V| \times |S|$ is defined for the TSK, where |*V*| is the cardinality of the set of tasks, and |*S*| is the maximum value of $|S_i|$. Each element of the TSK matrix is initialized using formula (1).

$$TSK(i,j,k) = \begin{cases} 0, & task\ \mathrm{T_i}\ cannot\ be\ processed\ on\ service\ instance\ s_j^k \\ s_j^k.t / s_j^k.c, & task\ \mathrm{T_i}\ can\ be\ processed\ on\ service\ instance\ s_j^k \end{cases} \tag{1}$$

Elite solution and task scheduling knowledge are extracted from the global best solutions of workflow scheduling.

2.3 Solution Construction

In the schedule construction phase, each artificial ant *a* will choose service instances for tasks and constructs a feasible solution. More specifically, the steps for ant *a* is as follows.

Step 1. Determine the allowed processing task set, referred to as *allowed*, consisting of all tasks whose predecessors have been scheduled.

Step 2. At each step ant *a* maps some service instance s_i^j to task *Ti* in terms of a probability p_{ij}^a and $TSK(i,i,j)$.

$$p_{ij}^a \times TSK(i,i,j) \tag{2}$$

where p_{ij}^a denotes the probability of mapping the service instance s_i^j to task *Ti* in terms of pheromone and heuristic information, as will be elaborated shortly, $TSK(i,i,j)$ denotes the task scheduling knowledge of mapping the service instance s_i^j to task *Ti*.

Step 3. Repeat step 1 to step 2 till all the tasks are mapped to the appropriate service instances.

The probability of mapping service instances to tasks is computed in terms of pheromone and heuristic information as shown by the formula (3).

$$p_{ij}^a = \begin{cases} \dfrac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{s_i^k \in S_i} [\tau_{ik}]^\alpha [\eta_{ik}]^\beta}, & if \quad s_i^k \in S_i \\ 0, & otherwise \end{cases} \tag{3}$$

where τ_{ij} is the pheromone value, and η_{ij} the heuristic information value for evaluating the possible decisions.

At the beginning of the algorithm, a pheromone value τ_{ij} is initialized using the cost for service instance s_i^j to accomplish task T_i, i.e.,

$$\tau_{ij} = \frac{t(T_i)}{s_i^j.t} \times s_i^j.c, \quad 1 \le i \le n, 1 \le j \le m_i \tag{4}$$

where $t(T_i)$ is the processing units of task T_i.

To minimize the cost while meeting the time constraint, we define a heuristic in terms of the latest start time (*LST*) of tasks. More specifically, the heuristic values η_{ij} is computed for available service instances in terms of the absolute differences to the maximum latest start time of an eligible task, that is,

$$\eta_{ij} = 1 \Big/ (\max_{s_i^k \in S_i} LST_{ik} - LST_{ij}) + 1 \tag{5}$$

After each ant has completed a schedule, the best solution found so far is then used to update the local scheduling pheromone in each generation. More specifically, if mapping service instance s_i^j to task T_i in the best solution found so far, some amount of pheromone is added to the pheromone τ_{ij}, that is, τ_{ij} is incremented by $\vartriangle\tau_{ij}$.

$$\tau_{ij} = (1-\rho)\tau_{ij} + \rho\vartriangle\tau_{ij} \tag{6}$$

where $\vartriangle\tau_{ij} = \frac{1}{C^*}$, C^* is the cost of the best found schedule so far, and parameter ρ denotes the evaporation rate ($0 \le \rho < 1$).

The workflow scheduling will be based on the mappings of ant *a*. *M* artificial ants build *M* solutions to the problem. Each ant maintains a building process and all ants construct their solutions in parallel. The best solution is one which minimizes the cost while meeting the deadline.

For global scheduling pheromone updating, only the global best ant, which constructed the global optimal schedule from beginning, is allowed to deposit knowledge to its corresponding solution. When the global optimal schedule is improved, the knowledge matrix *TSK* is updated correspondingly.

2.4 Knowledge Learning

(1) The elite solution knowledge will be updated by the predefined numbers of best solutions of every iteration.
(2) Adopts the concepts of the Max-Min Ant System (MMAS)[10], knowledge on each solution component is limited to an interval $[\tau_{\min}, \tau_{\max}]$ to avoid stagnation state. The *TSK* matrix will be updated by following rules.

• Knowledge depositing rule

$$TSK(i,i,j) = TSK(i,i,j) \times Q_1 \tag{7}$$

where Q_1 denotes the incremental coefficient of *TSK*.

• Knowledge evaporating rule

$$TSK(i,i,j) = \min\{\tau_{\max}, \max\{\tau_{\min}, (1-\rho)TSK(i,i,j)\}\} \tag{8}$$

where ρ is the evaporation rate.

Note that the knowledge depositing rule is only applied to the optimal schedule found so far, and the knowledge evaporating rule is performed after every trial.

3 Experimental Analysis

In this section we experimentally evaluated the performance of KBACO algorithm, and implemented comparison study of KBACO and published ACO algorithms for grid workflow scheduling.

3.1 Test Instances and Parameters Setting

We evaluate the KBACO algorithm in ten workflow applications (see Table 1). The first two workflows, including the neuroscience application [functionalMRI (fMRI)] with 15 tasks[11], and the e-protein workflow with 15 tasks[12], are derived from real-life applications. The other eight workflows are generated based on the networks in the project scheduling problem library (PSPLIB) [13], which is a library for project scheduling problems. These networks include j301_1 and j301_2 with 30 tasks, j601_1 and j601_2 with 60 tasks, j901_1 and j901_2 with 90 tasks, and j1201_1 and j1201_2, among which 120 problem instances are the largest contained in the PSPLIB.

Table 1. Test Instances

Workflow	Number of Tasks	Topology
fMRI	15	[16]
e-Protein	15	[17]
j301_1	30	j301_1(PSPLIB)
j301_2	30	j301_2(PSPLIB)
j601_1	60	j601_1(PSPLIB)
j601_2	60	j601_2(PSPLIB)
j901_1	90	j901_1(PSPLIB)
j901_2	90	j901_2(PSPLIB)
j1201_1	120	j1201_1(PSPLIB)
j1201_2	120	j1201_2(PSPLIB)

The QoS parameters (execution time and cost) of all service instances are randomly generated, but they follow the rule that for the same task, a service instance with higher shorter execution time may cost more money and vice versa.

There are mainly three parameters in the algorithm: α and β in the probability selection rule [Formula (3)] and ρ in the pheromone updating rule [Formula (6)]. In terms of the configuration given by the ACO algorithms for the workflow scheduling problem and other general scheduling problems[14-17], we set ρ = 0.1, α=1 and β=1 in our experiments.

3.2 Experiment Results

Our KBACO was implemented on an IBM XSERIES226 server with an Intel(R) XEON(TM) 3.00GHZ processor, and 2.00GB RAM. The KBACO terminates when one of the following conditions is satisfied.

(1) The global best solution is not improved in successive predefined number of iterations;
(2) The maximum preset search iterative is exhausted.

To these workflow applications, the final experimental results of our KBACO averaged over 100 runs were displayed as Table 2. Three indexes were applied to describe the experimental results.

(1) Best: the best cost among these 100 runs.
(2) Avg.: the average cost of these 100 runs.
(3) Avg. time: the average the average computational time of these 100 runs.

Table 2. Experimental results of kbaco for workflow applications

Workflow	Cost		Avg. time (s)
	Best	Avg.	
fMRI	101.5	106.5	210
e-Protein	123	124	220
j301_1	197	201	308
j301_2	213	218	335
j601_1	342	350	429
j601_2	458	465	475
j901_1	731	745	498
j901_2	796	812	510
j1201_1	1120	1134	535
j1201_2	1208	1215	546

The experimental results of Table 2 suggest that KBACO performed well on all workflow applications. It can obtain the near-optimal (optimal) solutions with a quick computational speed.

4 Conclusions

In this paper we have proposed a KBACO algorithm for grid workflow scheduling problem, which integrated the ACO model with knowledge model. In KBACO, knowledge was learned from the optimization of ACO by the knowledge model, at the same time, the existing knowledge was applied to guide the current heuristic searching of ACO. We have experimentally evaluated the KBACO algorithm in benchmark workflow applications with the number of tasks varying from 15 to 120. Experimental results demonstrate that the KBACO algorithm is capable of achieving solutions effectively and efficiently.

Acknowledgments. This work is supported in part by the National Natural Science Foundation of China under Grant No.60902094, 60903225, 60903206, 60172012.

References

1. Garey, D.: Computers and Intractability: A Guide to the Theory of NP Completeness. W. H. Freeman and Company, New York (1979)
2. Braun, T.D.: A Comparison Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems. J. Parallel Distrib. Comput. 61, 810–837 (2001)
3. XiaoShan, H., XiaoHe, S.: QoS Guided Min-Min Heuristic for Grid Task Scheduling. J. Comput. Sci. Technol. 18(4), 442–451 (2003)
4. Lopez, M.M., Heymann, E., Senar, M.A.: Analysis of Dynamic Heuristics for Workflow Scheduling on Grid Systems. In: 5th Int. Symp. Parallel Distrib. Comput. (ISPDC 2006), pp. 199–207 (2006)
5. Wang, L., Siegel, H.J., Roychowdhury, V.P., et al.: Task Matching and Scheduling in Heterogeneous Computing Environments Using a Genetic-Algorithm-based Approach. J. Parallel Distrib. Comput. 47, 8–22 (1997)
6. Martino, V.D., Mililotti, M.: Scheduling in a Grid Computing Environment Using Genetic Algorithms. In: Int. Parallel Distrib. Process. Symp. (IPDPS 2002), pp. 235–239 (2002)
7. Kim, J.-K.: Dynamically Mapping Tasks with Priorities and Multiple Deadlines in a Heterogeneous Environment. J. Parallel Distrib. Comput. 67, 154–169 (2007)
8. Dorigo, M., Stützle, T.: Ant Colony Optimization. MIT Press, Cambridge (2004)
9. Xing, L.N., Chen, Y.W., Wang, P., et al.: A Knowledge-Based Ant Colony Optimization for Flexible Job Shop Scheduling Problems. Applied Soft Computing (2009) (to be published)
10. Stützle, T., Hoos, H.: MAX-MIN Ant System and Local Search for the Traveling Salesman Problem. In: IEEE Int. Conf. Evol. Comput. (ICEC 1997), pp. 309–314. IEEE Press, New York (1997)
11. Zhao, Y.: Grid Middleware Services for Virtual Data Discovery Composition, and Integration. In: 2nd Workshop Middleware Grid Comput., Toronto, ON, Canada, pp. 57–62 (2004)
12. O'Brien, A., Newhouse, S., Darlington, J.: Mapping of Scientific Workflow within the E-Protein Project to Distributed Resources. In: UK e-Sci. All Hands Meet, Nottingham, UK, pp. 404–409 (2004)
13. Kolisch, R., Sprecher, A.: PSPLIB-A Project Scheduling Problem Library: OR Software-ORSEP Tasks Research Software Exchange Program. Eur. J. Oper. Res. 96(1), 205–216 (1997)
14. Chen, W.N., Zhang, J.: An Ant Colony Optimization Approach to a Grid Workflow Scheduling Problem With Various QoS Requirements. IEEE Transactions on Systems and Cybernetics—PART C: Applications and Reviews 39(1), 29–44 (2009)
15. Liu, J., Chen, L., Dun, Y., et al.: The Research of Ant Colony and Genetic Algorithm in Grid Task Scheduling. In: International Conference on Multi-Media and Information Technology, pp. 47–49 (2008)
16. Lo, S.T., Chen, R.M., Huang, Y.M., et al.: Multiprocessor System Scheduling with Precedence and Resource Constraints Using an Enhanced Ant Colony System. Expert Systems with Applications 34, 2071–2081 (2008)
17. Merkle, D., Middendorf, M., Schmeck, H.: Ant Colony Optimization for Resource-Constrained Project Scheduling. IEEE Trans. Evol. Comput. 6(4), 53–66 (2002)

An Improved Parallel Ant Colony Optimization Based on Message Passing Interface

Jie Xiong[1], Xiaohong Meng[1], and Caiyun Liu[2]

[1] Key Laboratory of Geo-detection (China University of Geosciences, Beijing), Ministry of Education, 100083 Beijing, China
[2] Freshman Education Department of Yangtze University, 434023 Jingzhou, Hubei, China
mxh@cugb.edu.cn

Abstract. Ant Colony Optimization (ACO) is recently proposed metaheuristic approach for solving hard combinatorial optimization problems. Parallel implementation of ACO can reduce the computational time obviously. An improved parallel ACO algorithm is proposed in this paper, which use dynamic transition probability to enlarge the search space by stimulating ants choosing new path at early stage; use polymorphic ant colony to improve convergence speed by local search and global search; use partially asynchronous parallel implementation, interactive multi-colony parallel and new information exchange strategy to improve the parallel efficiency. We implement the algorithm on the Dawn 4000L parallel computer using MPI and C language. The Numerical result indicates the algorithm proposed in this paper can improve convergence speed effectively with the fine solution quality.

Keywords: Parallel, Ant colony optimization, Dynamic transition probability, Polymorphic ant colony.

1 Introduction

Ant Colony Optimization(ACO) is a constructive population based meta-heuristic algorithm[1]. It has been applied to traveling salesman problem[2](TSP), quadratic assignment problem[3], mesh-partition problem[4], routing algorithm [5] and other combinational optimization(CO) problems. ACO has some disadvantages. Its' convergence speed becomes slow due to the random transition strategy when constructing the solution. It is easily premature and fall into local optimization solution due to the positive feedback. It needs considerable computational time and resources when the complexity of the problem increases.

Some strategies for parallelization have been reported recently. Buklnheimer[6] proposed two synchronous and asynchronous parallelization strategies. In the synchronous strategy, each processor exchanges information every iteration, while in the asynchronous strategy, each processor exchanges information after a certain iteration regularly. Talbi[7] presented a synchronous fine grained parallel ant colony algorithm in master/slave fashion combined with local tuba search, and applied this algorithm to solve quadratic assignment problem(QAP). Piriyakumar DAL[8] introduced an asynchronous parallel Max-Min ant colony algorithm associated with the local search strategy. Randal M[9] introduce five parallelization strategies and implement one of them named parallel ants. Blum C[10] advanced a parallel ant colony optimization on the hyper-cube architecture. By modifying the classical ACO, Merkle D[11] proposed a parallel ant colony algorithm on reconfigurable processor arrays.

This paper is organized into 4 sections. In section 1, a brief description of improvement and parallel implementation of ACO is introduced. Then, an improved parallel ACO algorithm named IPACO is proposed, which improve the ACS algorithm with the characteristic of polymorphic ant colony, dynamic transition probability and new parallel strategy in section 2. We evaluate the IPACO algorithm by study the convergence speed, parallel size scalability and problem size scalability of it in section 3, and draw a conclusion in section 4.

2 Improved Parallel Ant Colony Optimization (IPACO) Algorithm

2.1 Polymorphic Ant Colony

The study of real ants indicates the ant colony is complex. Different kind of ants has different kind of pheromone. Xu[12] introduce the concept of polymorphic ant colony into ACO. We divide the ant colony into scout ant colony and search ant colony enlightened by it.

There are n scout ants are scattered throughout n cities, one per city. Each scout ant explores $n-1$ neighbor cities and sorts them by distance from the city located, then releases scouting pheromone at the path linked to NN closest neighbor cities. The scouting pheromone is described in formulate (1) as follow:

$$s[i][j]=\begin{cases}\tilde{d}_{ij}/d_{ij},\text{if city j is NN closest neighbor of city i}\\ 0, otherwise\end{cases} \tag{1}$$

$\tilde{d}_{ij}$ is the shortest distance from city i, NN is the parameter indicating the closest neighbor number. According to the scouting pheromone, initialize the common pheromone in formulate (2) as follow:

$$\tau_{ij}(0)=\begin{cases}C*s[i][j], if\left(s[i][j]\neq 0\right)\\ C*\tilde{d}_{ij}/\hat{d}_{ij}, \text{otherwise}\end{cases} \tag{2}$$

$\hat{d}_{ij}$ is the longest distance from city i, C is the initial pheromone level of ACS algorithm. It can increase the convergence speed due to modify the initial common pheromone level differently.

Search ant colony is work like the normal ant colony in ACO. It chooses the next city not only by the information of common pheromone and distance, but by the scout pheromone. The transition probability of search ant colony is described in formulate (3) as follow:

$$p_{ij}^{k}(t)=\begin{cases}\dfrac{\left[\tau_{ij}(t)\right]^{\alpha}\bullet\left[\eta_{ij}(t)\right]^{\beta}}{\sum\limits_{k\in allowed_k}\left\{\left[\tau_{ik}(t)\right]^{\alpha}\bullet\left[\eta_{ik}(t)\right]^{\beta}\right\}}, j\in allowed_k, s[i][j]\neq 0\\ 0, otherwise\end{cases} \tag{3}$$

It can reduce the scale of search region when search ant choosing the next city because of the guide of scout pheromone. The search ants release the common pheromone on the path they passed every iteration only if the path's scout pheromone is not zero.

2.2 Dynamic Transition Probability

It is reported that the dynamic transition probability can avoid premature effectively[13]. The global optimization solution may be ignored as too strong pheromone level produced by the local optimization solution. If we stimulate more ants to attempt the new paths which little ants passing before at the early stage, the premature can avoid effectively. We proposed a new dynamic transition probability strategy more simply but more effective according to the ideal of it[13]. So the transition probability is modified as follow in formulate (4):

$$P_{ij}^{k}(t)=\begin{cases}\arg\max\left\{\left[\tau_{ij}(t)\right]^{\alpha}\bullet\left[\eta_{ij}(t)\right]^{\beta}\right\}, q\leq q_0, j\in allowed_k, s[i][j]\neq 0\\ \dfrac{\left[\tau_{ij}(t)\right]^{\alpha}\bullet\left[\eta_{ij}(t)\right]^{\beta}\bullet\delta_{ij}(t)}{\sum\limits_{k\in allowed_k}\left\{\left[\tau_{ik}(t)\right]^{\alpha}\bullet\left[\eta_{ik}(t)\right]^{\beta}\bullet\delta_{ik}(t)\right\}}, q>q_0, j\in allowed_k, s[i][j]\neq 0\\ 0, Otherwise\end{cases} \tag{4}$$

$\delta_{ij}=\dfrac{m\bullet N_c}{m\bullet N_c+\mu\bullet Q_c(i,j)}$,it is the adjust parameter to the transition probability of path (i,j) , m is the number of ants, N_c is the number of iterations, $Q_c(i,j)$ is the number of ants passed path (i,j) , and μ is a weighted parameter.

At the early stage of the algorithm, N_c is small. δ_{ij} decreases when $Q_c(i,j)$ increases. It stimulate more ant to explore new paths by restrain more ant choose the path (i,j) . At the later stage of algorithm, the $Q_c(i,j)$ of the fine solution is approximate equal to N_c , so δ_{ij} is approximate equal to $\dfrac{m}{m+\mu}$, and approximate equal 1 because of m is more larger than μ . Other path's δ_{ij} is approximate equal to 1 because $Q_c(i,j)$ is approximate equal to zero. So the algorithm is convergent.

2.3 Parallel Strategy

(1) Coarse-granularity interacting multi ant colonies [14]

It is reported that coarse-granularity is more suitable for the fine-granularity[6]. We adopt the ideal of coarse-granularity interacting multi ant colonies [14], same number ants are assigned to each processor. We regard the ants in the same processor as an independent colony. The ants in the same processor (sub-colony) can construct the solution independently guided by the local information. Different sub-colony interact each other by exchanging the information.

(2) Some results [6] [15] indicate that partially asynchronous parallel implement(PAPI) is better than synchronous implement. So we adopt the PAPI strategy that is the computational nodes (colony) exchange information each other every N iterations.

(3) Information exchange strategy

We design a new information exchange strategy, which exchanges the global best solution every N iterations and exchange the pheromone matrix every $k \bullet N$ iterations.

```
Step 1: Receive($\beta, \gamma, \rho, seed$), pheromone, distance matrix and N from
        master;
Step 2: If termination condition is not met, repeat Step 2 to Step 7, else
        terminate.
Step 3: Each node builds solution for N iterations by local information.
Step 4: Send the local-best-solution to master
Step 5: Receive the node's number, which constructs the Iteration-best-cost
        solution from master;
Step 6: Receive the pheromone matrix from the node, which constructs
        Iteration-best-cost solution;
Step 7: Receive the termination information signal from the master;
```

Fig. 1. The pseudocode of IPACO

We modify the ACS algorithm by introducing the characteristic of Polymorphic and dynamic transition probability discussed in 2.1 and 2.2, and propose an improved parallel ant colony optimization (IPACO) following the parallel strategy discussed in 2.3. The pseudocode of the IPACO is shown in Figure 1.

3 Numerical Result

3.1 General Description

The numerical experiments are performed on the platform of the Dawn 4000L parallel computer using the C language and the Message Passing Interface (MPI). A number of TSP problem instances from TSPLIB[16] have been selected with which to test the algorithm proposed in this paper. The program is running with the parameters as follows: $\alpha = 1, \beta = 5, m = 25, \rho = 0.9, \mu = 0.5, \lambda = 10, NN = 20$.

We evaluate the IPACO algorithm proposed in this paper by study the convergence speed, parallel size scalability and problem size scalability of it.

3.2 Convergence Speed Analysis of the IPACO Algorithm

In order to study the convergence speed of the IPACO algorithm, we use the Sequence ACS and IPACO (2 nodes) to solve the same TSP instance rl5915 as an example. We plot the relationship of solution value to iterations in Figure 2. From the Figure 2, we found the IPACO converge faster than the ACS, and the IPACO can find a better solution than the ACS at last.

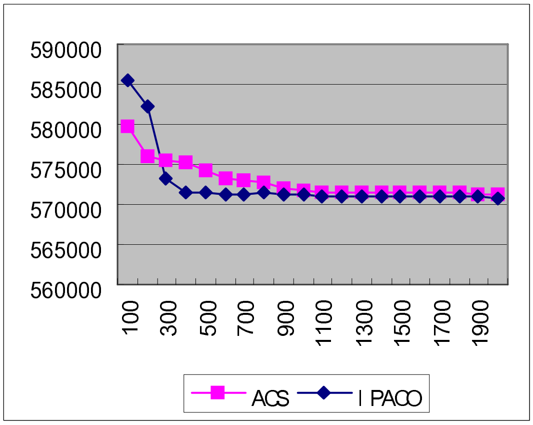


Fig. 2. Convergence speed analysis of IPACO algorithm

3.3 Parallel Size Scalability Analysis of the IPACO Algorithm

In order to study the parallel size scalability of the IPACO algorithm, we use different computational nodes such as 1,2, 4, 8 and 16 to solve the same TSP instance fl1577 as an example. We run the experiments 20 times and list the result in the Tab.1. sol_{avg} is average for solution. t_{avg} is average time for getting all the best solutions. e_{avg} is the excess from optimal of sol_{avg} .

Table 1. Experimental results of parallel size scalability analysis of IPACO algorithm

instance	Nodes num	sol_{avg}	t_{avg}	e_{avg} (%)	*speedup*	*efficiency*
	1	22307.3	251.83	0.2620	1.0000	1.0000
	2	22307.9	146.45	0.2647	1.7196	0.8598
fl1577	4	22304.7	101.82	0.2503	2.4733	0.618325
	8	22304.2	85.61	0.2481	2.9416	0.3677
	16	22309.4	78.53	0.2715	3.2068	0.200425

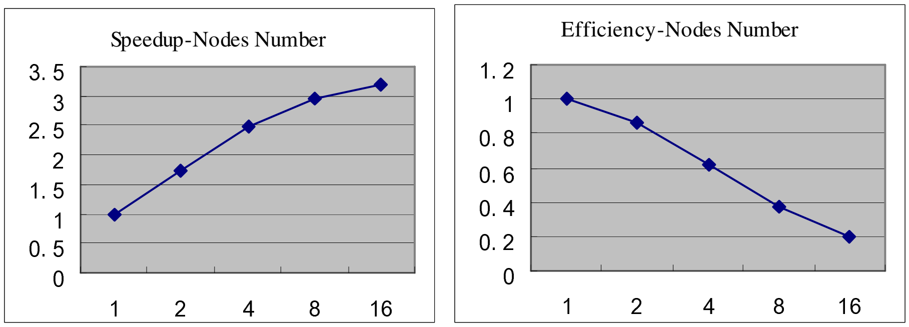


Fig. 3. The relationship of speedup and efficiency to parallel size of IPACO algorithm

We plot the relationship of speedup to nodes number (left) and efficiency to nodes number (right) in Figure 3. From Figure 3, we have found that there is a steady decrease in computational time and a steady increase in speedup with the increasing of nodes number. It indicates more computing nodes can reduce the computational time obviously and obtain significant speedup.

3.4 Problem Size Scalability Analysis of the IPACO Algorithm

We use 2 nodes to solve the different scale TSP instances, and compare them with sequential algorithms. We run the experiments 20 times using sequence ant colony algorithm (SACS) and improved parallel ant colony algorithm (IPACO), then list the result in the Table 2. $t_{\max}$ is maximum time allowed for running. i_{avg} is average iterations for getting all the solutions. The sol_{avg} , t_{avg} , e_{avg} are defined in 3.3.

Table 2. Experimental results of problem size scalability analysis of IPACO algorithm

instance	algorithm	sol_{avg}	$t_{\max}$	t_{avg}	e_{avg} (%)	speedup	efficiency
D198	IPACO	15780.1	20	4.16	0.0006	0.8125	0.4063
	SACS	15780.1	20	3.38	0.0006		
Gr666	IPACO	294685.2	100	38.23	0.1112	1.487575	0.7438
	SACS	294620.4	100	56.87	0.0891		
Pcb1173	IPACO	56901.8	200	67.3	0.0172	1.583655	0.7918
	SACS	56908.3	200	106.58	0.0287		
fl1577	IPACO	22307.9	400	146.45	0.2647	1.719563	0.8598
	SACS	22307.3	400	251.83	0.262		
pr2392	IPACO	379258.7	1000	252.74	0.3245	1.877912	0.9390
	SACS	379264.8	1000	504.67	0.3261		
rl5915	IPACO	570874.7	2000	942.09	0.9451	1.913915	0.9570
	SACS	571134.6	2000	1803.08	0.991		

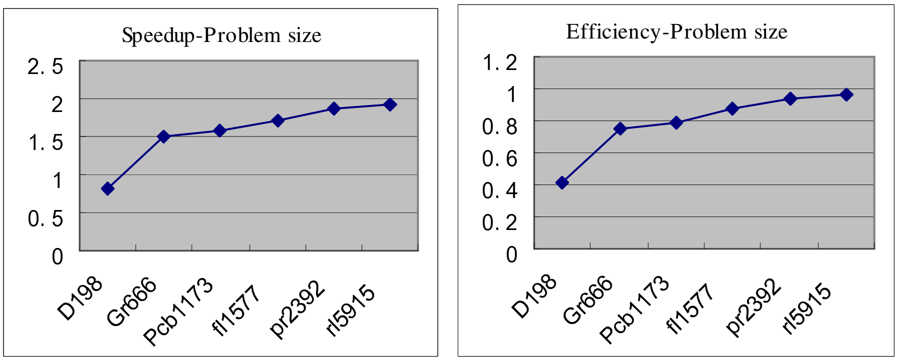


Fig. 4. The relationship of speedup and efficiency to problem size of IPACO algorithm

We calculate *speedup* and *efficiency* and plot them in Figure 4. From Fig. 4, we have found that there is a steady increase in speedup and parallel efficiency with the increasing of the scale of the instance. The speedup and efficiency are closed to the limit 2 and 1 with the large instance of rl5915. It indicates that the parallel ant colony optimization algorithm is more efficient for the large scales problem.

4 Conclusion

In this paper, we propose an improved parallel ant colony optimization algorithm with polymorphic, dynamic transition probability and new parallel strategies. The code is written in C and MPI to implement this IPACO algorithm and has been executed on the dawn 4000L parallel computer. We evaluate the IPACO algorithm proposed in this paper by study the convergence speed, parallel size scalability and problem size scalability of it. The numerical results indicate that (1) the IPACO algorithm can construct solution better than the sequential ACO (SACS) algorithm and converge faster then SACS; (2) more computing nodes can reduce the computing time obviously and obtain significant speedup; (3) the IPACO algorithm is more efficient for the large scale traveling salesman problem with fine quality of solution.

References

1. Dorigo, M., Stützle, T.: Ant Colony Optimization. MIT Press, Cambridge (2004)
2. Dorigo, M., Gambardella, L.M.: Ant colonies for the travelling salesman problem. BioSystems 43, 73–81 (1997)
3. Gambardella, L.M., Taillard, E.D., Dorigo, M.: Ant colonies for the quadratic assignment problem. Journal of the Operational Research Society 50, 167–176 (1999)
4. Korosec, P., Silc, J., Robi, B.: Solving the mesh-partitioning problem with an ant-colony algorithm. Parallel Computing 30, 785–801 (2004)
5. Sun, Z.-x., Xia, Y.-a.: Research on QoS muticast routing algorithm based mixed AntNet algorithm. Journal of Communications 30, 6 (2009)

6. Bullnheimer, B., Kotsis, G., Strauss, C.: Parallelization strategies for the Ant System. Technical Report POM 9-97. Vienna University of Economics and Business Administration (1998)
7. Talbi, E.G., Roux, O., Fonlupt, C., Robillard, D.: Parallel Ant Colonies for the quadratic assignment problem. Future Generation Computer Systems 17, 441–449 (2001)
8. Piriyakumar, D.A.L., Levi, P.: A new approach to exploiting parallelism in ant colony optimization, pp. 237–243 (2002)
9. Randall, M., Lewis, A.: A Parallel Implementation of Ant Colony Optimization. Journal of Parallel and Distributed Computing 62, 1421–1432 (2004)
10. Blum, C., Roli, A., Dorigo, M.: HC–ACO: The hyper-cube framework for Ant Colony Optimization. In: Proceedings of MIC 2001–Meta–heuristics International Conference, Porto, Portugal, vol. 2, pp. 399–403 (2001); Also available as technical report TR. IRIDIA/2001-16, IRIDIA, Universite Libre de Bruxelles, Brussels, Belgium (2004)
11. Merkle, D., Middendorf, M.: Fast Ant Colony Optimization on Runtime Reconfigurable Processor Arrays. Genetic Programming and Evolvable Machines 3, 345–361 (2004)
12. Xu, J.-m., Cao, X.-b., Wang, X.-f.: Polymorphic Ant Colony Algorithm. Journal of University of Science and Technology of China 35, 7 (2005)
13. Zheng, S., Hou, D.-b., Zhou, Z.-k.: Ant colony algorithm with dynamic transition probability. Control and Decision 23, 4 (2008)
14. Xiong, J., Liu, C., Chen, Z.: A New Parallel Ant Colony Optimization Algorithm Based On Message Passing Interface (2008)
15. Manfrin, M., Birattari, M., Stützle, T., Dorigo, M.: Parallel Ant Colony Optimization for the Traveling Salesman Problem. In: Dorigo, M., Gambardella, L.M., Birattari, M., Martinoli, A., Poli, R., Stützle, T. (eds.) ANTS 2006. LNCS, vol. 4150, pp. 224–234. Springer, Heidelberg (2006)
16. TSPLIB, `http://www.aco-metaheuristic.org/aco-code`

Research on Fault Diagnosis Based on BP Neural Network Optimized by Chaos Ant Colony Algorithm

Liuyi Ling, Yourui Huang, and Liguo Qu

School of Electric and Information Engineering,
Anhui University of Science and Technology, Huainan 232001, China
lyling@aust.edu.cn

Abstract. In view of shortcomings of BP neural network, which is slow to converge and tends to trap in local optimum when applied in fault diagnosis, an approach for fault diagnosis based on BP neural network optimized by chaos ant colony algorithm is proposed. Mathematical model of chaos ant colony algorithm is created. Real-coded method is adopted and the weights and thresholds of BP neural network are taken as ant space position searched by chaos ant colony algorithm to train BP neural network. Training result of chaos ant colony algorithm is compared with that of conventional BP algorithm and from both results it is can be seen that chaos ant colony algorithm can overcome the shortcomings of BP algorithm. It is proved that mathematical model of chaos ant colony algorithm is correct and optimization method is valid through experimental simulation for machinery fault diagnosis of mine ventilator.

Keywords: chaos ant colony algorithm, fault diagnosis, BP neural network, optimization algorithm.

1 Introduction

BP neural network, which has functions of self-learning, self-adaptation and nonlinear pattern recognition, is applied widely to fault diagnosis. However, shortcomings of BP network are that it is slow to converge and tends to trap in local optimum. Ant colony algorithm is a new search algorithm in combinatorial optimization field. At initialization, information element of each path adopts the same value, and this makes ants very difficult to find a better path in short time from a mass of disorderly and unsystematic paths. So search efficiency and convergence speed of ant colony algorithm are also low and algorithm tends to trap in local optimum. In recent years, the biologist Cole has found that behavior of the whole ant colony is a kind of periodic behavior and that of the single ant is chaos behavior [1-2]. So chaos theory is used to improve ant colony algorithm through introducing the chaos perturbation operator into the algorithm and then parameters of the algorithm are initialized by use of ergodicity and randomicity of chaos theory. This method increases the diversity of the solution and the better path can be selected from a mass of paths. So it leads ants to find the fastest and optimum path.

Chaos ant colony algorithm is applied to fault diagnosis in this paper, and the algorithm is used to train the weights and thresholds of BP neural network. Then the trained neural network is applied to the fault diagnosis of mine ventilator and the better diagnosis results are acquired through experiment simulation.

2 Fault Diagnosis Principle Based on BP Neural Network

When BP neural network is used to diagnose fault, first of all, the network need to be trained through a group of training samples. Input of the samples is fault character, and the desired output is corresponding fault type. After BP neural network has been trained, it can be tested using test samples and the most possible fault type is informed.

Suppose BP neural network adopts three-layer structure of input layer, single hidden layer and output layer. The input layer has m nodes, and corresponding input vector is $\mathbf{X}=(x_1,x_2,\cdots,x_m)$, the hidden layer has p nodes, and corresponding output vector is $\mathbf{Y}=(y_1,y_2,\cdots,y_p)$, and the output layer has n nodes, and corresponding output vector is $\mathbf{O}=(o_1,o_2,\cdots,o_n)$. Suppose connection weights between any node x_i of the input layer and any node y_k of the hidden layer is ω_{ik}, thresholds of No. k neuron of the hidden layer is θ_k, connection weights between any node y_k of the hidden layer and any node o_j of the output layer is ω_{kj}, and thresholds of No. j neuron of the output layer is θ_j. And then the neuron output of the hidden layer and output layer are as following:

$$\begin{cases} y_k = f\left(\sum_{i=1}^{m}\omega_{ik} - \theta_k\right), & k=1,2,\cdots,p \\ o_j = f\left(\sum_{k=1}^{p}\omega_{kj}y_k - \theta_j\right), & j=1,2,\cdots,n \end{cases} \tag{1}$$

In formula (1), $f(\cdot)$ is the activation function of the neuron, and here is an S-type function as following:

$$f(x)=\frac{1}{1+e^{-x}} \tag{2}$$

Formula (3) is taken as error output of BP algorithm when BP neural network is trained through fault samples.

$$E=\frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{n}(o_{ji}^{d}-o_{ji})^2 \tag{3}$$

In formula (3), M is sample quantity of training samples set, n is quantity of network output neurons; o_{ji}^{d} is the desired output of No. j output node of No. i sample; o_{ji} is the actual output of No. j output node of No. i sample.

From formula (1) and formula (3), it is can be seen that the error output E of BP algorithm is a function related to the connection weights and thresholds of all layers. So, the error output E can be changed through adjusting the connection weights and thresholds.

3 Chaos Behavior and Ant Self-organization Behavior

The first research of chaos phenomenon of insect behavior originated from the experiment result of observing Leptothax allardycei (one kind of ant) [3-7]. Against conventional prediction, ants do not work at all times but rest in fact in most time. Herbers's research on two different kinds of ants Leptothrax longispinosus and L.Ambiguus shows that ants do nothing in two-thirds life time [8-9]. Further research Cole does on Leptothrax longispinosus shows that behavior of ant colony is regular and periodic.

Ant colony consisting of many ants, which behaviors are sample and affect each other, can self-adaptively find the place where food is abundant. Two consecutive dynamics processes happen during looking for food. First is a non-cooperative process and in this process ant organization capability is very poor. Single ant will help itself and other ants to look for better food path through leaving pheromone. First process continues till ant individual behavior is enough affected and here second process, which is a cooperative process among ants, starts. In the whole self-organization process, every ant exchanges optimum position information with its neighbor and compares the information with before and memorizes it.

4 Mathematical Model of Chaos Ant Colony Algorithm

Chaos ant colony algorithm will search the optimum resolution in the l demention real number space R^l. Suppose that ant quantity of ant colony is n. All ants are put in the search space S and they will minimize a function $f : S \to R$. Each point s in space S is one feasible solution of the given problem. The algebra variable sign $s_i = (z_{i1}, z_{i2}, ..., z_{il})$ denotes the position of No. i ant, here $i = 1,2,...,n$. Naturally, each variable can be arbitrarily finite dimensional.

During the movement of the ants, each ant will be influenced by the organization of the whole ant colony. On mathematics expression, the movement strategy of single ant is a function related to own present position, optimum position found at present between itself and own accompanier, and organization variable[10]. The function is as following:

$$z_{id}(t) = g(z_{id}(t-1), p_{id}(t-1), y_i(t)) \tag{4}$$

In formula (4), t is the moment of present step, $t-1$ is the moment of last step, $z_{id}(t)$ is the present status of d dimension variable of No. i ant, here $d = 1,2,...,l$, l is the dimension number of the search space, $y_i(t)$ is the current status of the organization variable, $p_{id}(t-1)$ is optimum position that is found by No. i ant and its neighboring ant in No. $t-1$ step, g is a nonlinear function. Through introducing a continuously-changed organization variable $y_i(t)$ to realize the chaos behavior of ants, the system dynamics model of chaos ant colony optimization algorithm is as following:

$$\begin{cases} y_i(t) = y_i(t-1)^{(1+r_i)} \\ z_{id}(t) = z_{id}(t-1)\exp((1-\exp(-ay_i(t)))(3-\psi_d z_{id}(t-1))) \\ \qquad + \exp(-2ay_i(t)+b)(p_{id}(t-1)-z_{id}(t-1)) \end{cases} \tag{5}$$

In formula (5), a is an enough large positive constant, b is a constant and $0 \le b \le 2/3$, ψ_d decides the search range of No. d element in the search space, r_i is a positive constant less than 1 and nominated as the organization factor of No. i ant, and $y_i(0) = 0.999$. r_i and ψ_d are two important parameters of the given optimization algorithm. r_i will influence convergence speed of chaos ant colony algorithm, generally selected as $0 \le r_i \le 0.5$. ψ_d will influence the search range of chaos ant colony algorithm.

5 BP Neural Network Optimized Flow by Chaos Ant Colony

It is required that a suitable chaos ant colony search pattern should be created and fitness function and search spaces should be defined when chaos ant colony algorithm is used to train BP neural network. Here, the weights and thresholds of BP learning algorithm are taken as ant space position searched by chaos ant colony algorithm. That is, learning process of optimized BP neural network is in nature that ant searches optimum position within search space. The detailed flow chaos ant colony algorithm optimizes BP neural network is shown in Fig.1.

Based on BP neural network optimized flow by chaos ant colony algorithm, the detailed steps of fault diagnosis are given as follows:

Step1: Define BP neural network structure according to input and output of fault samples of diagnosed object.

Step2: Initialize parameters of chaos ant colony algorithm, which mainly includes defining ant quantity, cycle iterations and search space. Then, randomly generate a ant colony.

Step3: Take fault samples as training samples of BP neural network and start using chaos ant colony algorithm to train BP neural network.

Step4: Compute fitness function value of chaos ant colony algorithm with formula (3) and judge it is ant optimum position or not according to the principle that fitness function value is minimal. In the procedure of optimization iteration cycles, if computed fitness function value is less than current minimum correspond to optimum position for any ant then update own optimum fitness function value, and then assign current optimum value in ant colony to $p_{id}(t)$.

Step5: Update $y_i(t)$ and $z_{id}(t)$ with formula (5). Then judge whether iteration is over. If it is, continue Step6, or else back to Step4 and go on iteration.

Step6: Use test samples to test fault diagnosis system after BP neural network is trained by chaos ant colony algorithm and the most possible fault type is informed.

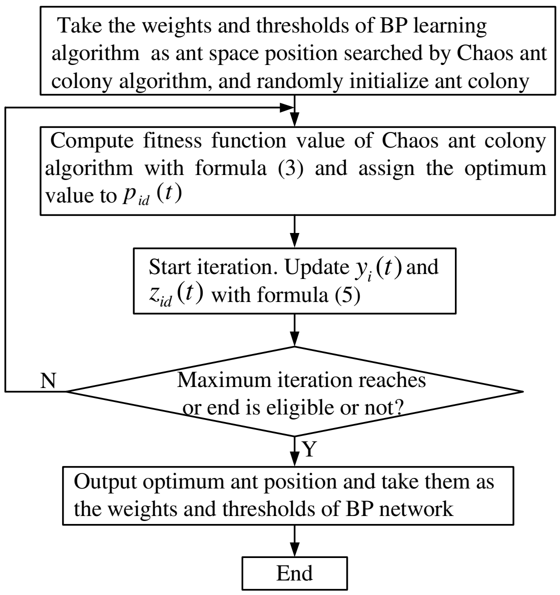

Take the weights and thresholds of BP learning algorithm as ant space position searched by Chaos ant colony algorithm, and randomly initialize ant colony
Compute fitness function value of Chaos ant colony algorithm with formula (3) and assign the optimum value to $p_{id}(t)$
Start iteration. Update $y_i(t)$ and $z_{id}(t)$ with formula (5)
Maximum iteration reaches or end is eligible or not?
N
Y
Output optimum ant position and take them as the weights and thresholds of BP network
End

Fig. 1. Flow diagram chaos ant colony algorithm optimizes BP neural network

6 Experimental Simulation and Result Analysis

The above-mentioned fault diagnosis method is simulated with MATLAB software through taking mine ventilator machinery fault diagnosis for example. It is a precondition to analyze mechanical vibration character of mine ventilator for getting fault character and diagnosing fault. Table 1 shows relationship between fault character and fault type of mine ventilator, and data in table 1 are taken as training samples of BP neural network. In table 1, f_1 is vibration frequency caused by balance fault, f_2 is vibration frequency caused by blade fault, f_o is bearing outer ring character frequency, f_i is bearing inner ring character frequency, f_b is roller character frequency and f_f is hold-shelf character frequency. It is to be noted here that x_i in table 1 is a normalized value and the normalized formula is as follow:

$$x_i = \frac{x_i' - x_{\min}}{x_{\max} - x_{\min}}, \quad i = 1,2,\cdots,8 \tag{6}$$

In formula (6): x_i is a normalized fault feature value, x_i' is an energy value every kind of character frequency has, $x_{\max}$ and $x_{\min}$ are respectively maximum and minimum in energy values all kinds of character frequency have.

Table 1. Training samples for mine ventilator fault diagnosis

ventilator status	fault character (sample input)								desired output								
	f_1	$2f_1$	f_2	$(2\sim5)f_1$	f_0	f_i	f_b	f_f									
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8	o_9
no fault	0.2	0.2	0.4	0.1	0	0	0	0	1	0	0	0	0	0	0	0	0
imbalance	1	0.2	0.4	0.1	0	0	0	0	0	1	0	0	0	0	0	0	0
disalignment	0.2	1	0.4	0.7	0	0	0	0	0	0	1	0	0	0	0	0	0
machinery loosen	0.2	0.2	0.4	0.9	0	0	0	0	0	0	0	1	0	0	0	0	0
blade fault	1	0.2	0.9	1	0	0	0	0	0	0	0	0	1	0	0	0	0
bearing outer ring fault	0.2	0.2	0.4	0.1	1	0	0	0	0	0	0	0	0	1	0	0	0
bearing inner ring fault	0.2	0.2	0.4	0.1	0	1	0	0	0	0	0	0	0	0	1	0	0
roller fault	0.2	0.2	0.4	0.1	0	0	1	0	0	0	0	0	0	0	0	1	0
hold-shelf fault	0.2	0.2	0.4	0.1	0	0	0	1	0	0	0	0	0	0	0	0	1

In this fault diagnosis example, 3-layer BP neural network consists of eight input neurons, twenty hidden neurons and nine output neurons. Error convergence factor of BP algorithm is 0.001 and activation functions of hidden layer and output layer are both Sigmoid-function. Parameters of chaos ant colony algorithm are as following: ant quantity is 20, $a = 200$, $b = 2/3$, $\psi_d = 3$, $y(0) = 0.999$, $r_i = 0.1 + 0.2rand$ and maximum of iterative times is 80. Take data in table 1 as training samples to train BP neural network by use of chaos ant colony algorithm and the relationship curve between training times and training error of BP neural network optimized by chaos ant colony algorithm is shown in Fig.2. Using same parameters, training error curve of conventional BP neural network is shown in Fig.3.

From Fig.2 and Fig.3, it can be seen that system error has reached 0.001 when carrying out 24th iteration using chaos ant colony algorithm to train neural network and however, required accuracy has not reached yet when carrying out last iteration using

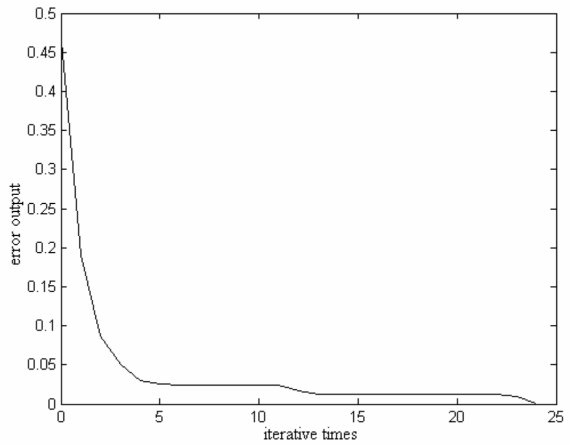


Fig. 2. Training curve of BP network optimized by chaos ant colony algorithm

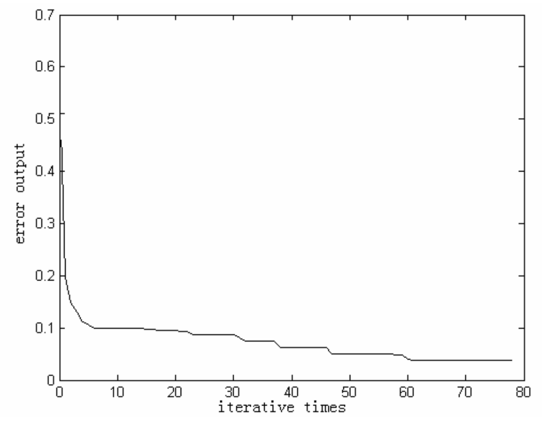


Fig. 3. Training curve of conventional BP neural network

Table 2. Fault test samples of mine ventilator

ventilator status	test samples input							
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
1. no fault	0.289	0.242	0.328	0.107	0.087	0.122	0.006	0.003
2. imbalance	0.986	0.245	0.568	0.096	0.065	0.056	0.012	0.002
3. machinery loosen	0.146	0.248	0.356	0.867	0	0.136	0.058	0
4. blade fault	0.976	0.363	0.923	0.995	0.105	0.016	0.112	0.009
5. roller fault	0.224	0.178	0.443	0.125	0	0.045	0.965	0.078

Table 3. Network output of test samples and diagnosis results

	network output of test samples									diagnosis results
	o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8	o_9	
1.	0.9968	0.0032	0.0006	0.0015	0.0009	0	0.0001	0.0003	0.0007	no fault
2.	0.0006	0.9957	0.0005	0.0010	0.0008	0.0005	0.0012	0.0007	0.0009	imbalance
3.	0.0003	0	0.0001	0.9969	0.0001	0.0002	0.0001	0	0	machinery loosen
4.	0	0.0002	0.0001	0.0001	0.9964	0.0011	0.0002	0.0001	0.0001	blade fault
5.	0	0.0001	0.0004	0.0001	0	0.0001	0.0013	0.9962	0.0022	roller fault

BP algorithm and in fact BP algorithm has trapped in local minimum error 0.0617. Compare Fig.2 with Fig.3, training curve in Fig.2 is steeper, this shows that chaos ant colony algorithm tends to get out of local optimum and search global optimum.

Data in table 2 are fault test samples, and served as input of trained neural network. Output of neural network and fault diagnosis results are shown in table 3. It can be seen from diagnosis results that BP neural network optimized by chaos ant colony algorithm can discern fault type very well and actual output of network is almost the same as desired output. Even when fault samples are disturbed, the diagnosis system has great discerning capability.

7 Conclusion

Study effect of conventional BP algorithm is related to initialized weights and the algorithm only can deal with differentiable object function. However chaos ant colony algorithm can overcome these shortcomings and improve local search capability so that it can avoid trapping early in local optimum. If BP neural network optimized by chaos ant colony algorithm is used for fault diagnosis, reliability and accuracy of diagnosis result will be greatly increased because this diagnosis method not only reflects fault diagnosis capability of BP neural network but also enough utilizes search optimum capability of chaos ant colony algorithm.

References

1. Cole, B.J.: Short-term activity cycles in ants: generation of periodicity by worker interaction. Am. Nat. 137, 244–259 (1991)
2. Cole Blaine, J.: Is animal behavior chaotic? Evidence from the acitivity of ants. Proceedings of The Royal Society of London Series B-Biological Sciences 244, 253–259 (1991)
3. Wilder, J.W., Vasquez, D.A., Christie, I., et al.: Wave trains in a model of gypsy moth population dynamics. Chaos 5(4), 700–706 (1995)
4. Bio, S., Couzin, I.D., Del, N.B., et al.: Coupled oscillators and activity waves in ant colonies. Proceedings of the Royal Society of London Series b-Biological Sciences 266, 371–378 (1999)
5. Logan, J.A., Allen, J.C.: Nonlinear dynamics and chaos in insect populations. Annual Review of Entomology 37(1), 455–477 (1992)
6. Frank, J.H., Mccoy, E.D.: Introduction to attack and defense behavioral ecology of defense medieval insect behavioral ecology, and chaos. Entomologist 74(1), 1–9 (1991)
7. Costantino, R.F., Desharnais, R.A., Cushing, J.M., et al.: Chaotic dynamics in an insect population. Science 275, 389–391 (1997)
8. Suddy, J.H.: An introduction to the behavior of ants. Arnold, London (1967)
9. Delgado, J., Sole, R.V.: Self-synchronization and task fulfillment in ant colonies. Jouranl of Theoretical Biology 205(3), 433–441 (2000)
10. Li, L.-X., Peng, H.-P., Yang, Y.-X., Wang, X.-D.: Parameter estimation for Lorenz chaotic systems based on chaotic ant swarm algorithm. Acta Physica Sinica (1), 51–55 (2007)

Edge Detection of Laser Range Image Based on a Fast Adaptive Ant Colony Algorithm*

Yonghua Wu[1,2,3,**], Yihua Hu[2,3], Wuhu Lei[2,3], Nanxiang Zhao[2,3], and Tao Huang[2,3]

[1] 460 Huangshan Rd. Hefei, Anhui, 230037, China
Tel.: 86-551-5767748
[2] State Key Laboratory of Pulsed Power Laser Technology (Electronic Engineering Institute), Hefei 230037, China
[3] Key Laboratory of Electronic Restriction, Anhui Province, Hefei 230037, China
wuyonghua2010@126.com

Abstract. Laser range imaging is the current priority research areas of airborne lidar. And realizing accurate edge detection of laser range image is the key of completing the subsequent three-dimensional reconstruction. Based on the characteristics of laser range image and the deficiencies of traditional edge detection methods, a new improved fast adaptive ant colony algorithm for edge detection of laser range image has been proposed in this paper. Due to the initial cluster center and the heuristic guiding function used in the algorithm, the randomness and blindness of ants walking are eliminated thoroughly, and the speed of image edge detection is also greatly increased. Meanwhile, thanks to the applied ants' selection mechanism and updating mechanism varying in contents, the error detection rate and omission factor of edge points as well as noise interference are all avoided, and the accuracy and adaptability of laser range image edge detection are greatly improved as well. Experimental results have shown that, this algorithm is more effective than other edge detection methods, and can meet the requirements of three-dimensional reconstruction.

Keywords: Ant Colony Algorithm, Edge Detection, Laser Range Image, Three-Dimensional Reconstruction, Contrast Experiment.

1 Introduction

Airborne lidar can quickly acquire large-scale, high precision, high spatial resolution three-dimensional geography information, which can also resolve thoroughly the bottlenecks of constructing three-dimensional city models based on traditional photogrammetric methods. Especially, the laser range image from the greyscale quantization imaging of laser pin-point elevation data has become the priority research areas of LIDAR, which contains plenty of necessary information for constructing three-dimensional city, such as the plane position, contour and elevation of buildings. Therefore, how to accurately detect the edge of laser range image is the key of completing the subsequent three-dimensional reconstruction (e.g. [1-2]).

* Supported by the National Natural Science Foundation of China (Grant No.60672154).
** Corresponding author.

There are a great many traditional image edge detection approaches, which mainly use the Gradient extremum property of image edge to detect the partial zero-crossing first derivative or second-order derivative as the edge points, such as the Sobel edge operator, Roberts edge operator, LoG edge operator, etc. However, owing to the large-scale data and serious noise pollution of laser range image, those traditional methods could not be well applied (e.g. [3-4]). Besides, based on the classic principle of edge detection, some new methods have also been proposed by many researchers to deal with laser range image. On the whole, they fall into three main categories: the optimal operator, multi-scale method and adaptive method (e.g. [5]). However, they still are not effective for edge detection of laser range image. In recent years, with the emergence of the ant colony algorithm with the discrete and parallel character, many domestic and abroad scholars have obtained abundant research achievements in applying ant colony algorithm to solve the problem of image segmentation.

Consequently, based on the characteristics of laser range image and the advantages of traditional edge detection methods, a new ant colony optimization algorithm for laser range image edge detection has been proposed in this paper. Section 2 states the characteristics of laser range image and the principle of the basic ant colony algorithm. Section 3 presents the proposed ant colony optimization algorithm. Section 4 gives the experimental results of laser range image edge detection.

2 Laser Range Image and Basic Ant Colony Algorithm

2.1 Characteristics of Laser Range Image

Laser range image is the range-grayscale digital image, which is mainly based on converting the distance data of scanned discrete laser pin points into the regular digital distance matrix through grid interpolating, and then executing the grayscale or pseudocolor quantization according to different gray levels corresponding to different distances. Generally, laser range image contains a wealth of necessary information for reconstructing three-dimensional scene, such as the plane position, contour and distance data of the objects in the scene.

Compared with the gray image from aerophotogrammetry, the data of laser range image measured by airborne lidar is quite huge, and the image quality is also poor. Especially, the boundary information of objects on image looks very fuzzy. This is mainly caused by interpolating the data of discrete laser pin point based on regular grid. Consequently, the fuzzy edges of objects and the boundaries region affected by noises on image have brought a great many difficulties to the edge detection and extraction of laser range image (e.g. [6]).

2.2 Principle of Edge Detection Based on Basic Ant Colony Algorithm

Ant Colony Algorithm (ACA) is an emerging probabilistic search algorithm in recent years. It completes the global optimization search process based on utilizing the ant pheromone as the basis of choosing subsequent actions, and combining with cooperation and interaction between ants. (e.g. [7-9]).

The principle of edge detection based on ant colony algorithm is regarding the original laser range image X as an undirected graph. And each pixel X_j on image is treated as an ant and expressed as a three-dimensional vector characterized by gray-scale, gradient and the neighbourhood. The edge points on image are the food sought by ants, and all the ants walk step by step according to the mechanism in path selection. Based on the probability obtained from the pheromone quantity ph_{ij} and the visibility guiding function η_{ij} between pixels, the subsequent shift location will be chosen from 3×3 pixel-neighbourhood. Then most ants will be gathered on the edge of the image through multiple circulative iterations, to achieve edge detection.

Suppose m is the number of ants, p is weighted factor (its value is set based on the degree of each component of pixel affecting the cluster). The distance between arbitrary pixel X_i and X_j is d_{ij}, and is expressed in Euclidean Distance.

$$d_{ij}=\sqrt{\sum_{k=1}^{m} p_k (x_{ik}-x_{jk})^2}\ . \tag{1}$$

If r is clustering radius, ph_{ij} is pheromone quantity on the path of pixel X_i to X_j, so

$$ph_{ij}=\begin{cases}1, & \text{if } d_{ij}\le r\\ 0, & \text{otherwise}\end{cases}\ . \tag{2}$$

Selection mechanism. If the initial pheromone quantities on various paths are the same, as $ph_{ij}(0)=C$ (constant), so for ants $j\ (j=1,2,\cdots,N)$, the probability p_{ij} of the shifting path from pixel X_i to X_j at the moment t is shown as follows:

$$p_{ij}=\begin{cases}\dfrac{ph_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum\limits_{s\in S} ph_{is}^{\alpha}(t)\eta_{is}^{\beta}(t)}, & \text{if } j\in S\\ 0, & \text{otherwise}\end{cases}\ . \tag{3}$$

Where, $S=\{X_s \mid d_{sj}\le r, s=1,2,\cdots,N\}$ is a feasible path selected by ants. α and β reflect the impact of information accumulated and heuristic factor on selecting path.

Update mechanism. Following a cycle of ants, the pheromone quantity on various paths can be adjusted according to the following formula:

$$ph_{ij}(t+1)=(1-\rho)ph_{ij}(t+1)+\Delta ph_{ij}\ . \tag{4}$$

$$\Delta ph_{ij}=\sum_{k=1}^{N}\Delta ph_{ij}^{k}\ . \tag{5}$$

Where, Δph_{ij}^{k} is the pheromone quantity left in the path (X_i, X_j) by the ant k between the moment t and $t+1$, ρ is the attenuation of pheromone quantity.

When the peset iteration times are completed by Loops, the whole algorithm will finish, and the target edge will be determined according to comparing ants' number passing each pixel on the image with set prior threshold, then the final result is out through thinning the image edge.

2.3 Shortage of Basic Ant Colony Algorithm

Through analysis, the following main deficiencies of basic ant colony algorithm can be found for the edge detection of laser range image:

First, a fairly long searching time and very large overall calculation. Due to randomness and blindness of ants' walk, for a range image with the size of $m \times n$, each pixel need calculate its distance and probability of selecting path with other pixels $m \times n - 1$, thus the algorithm must execute multiple cycles for completing the clustering process (e.g. [10]).

Second, the problem of easily emerging error detection or omission edge points. The edge detection of laser range image based on basic ant colony algorithm equally treats and deals with all the pixels in the image. In other words, both the ants' selection mechanism and the update mechanism of pheromone are identical. Thus, the accuracy of edge detection will be inevitably reduced, especially the omission detection of partial edge points or the error detection of partial target and background pixels will occur frequently.

3 Proposed Algorithm

In order to overcome the above-mentioned deficiencies of basic ant colony algorithm, and improve the edge detection accuracy of laser range image. A new ant colony optimization algorithm for laser range image edge detection has been proposed in the following paper.

3.1 Setting the Initial Cluster Centers and Guiding Function

Based on analyzing the characteristics of laser range image, general grayscale, gradient and neighborhood characteristics corresponding with the target, background, border, and noise in the image are used as the characteristics of initial cluster center. Furthermore, the similarity between ants and cluster center is also taken as a guiding function, to make ants search quickly and efficiently for a new cluster center. Suppose the initial cluster center can be expressed as $C_j(V, G, L)$ (e.g. [11]).

Calculating the grayscale of the initial cluster center-defining the characteristic vector V: Based on the histogram of laser range image, the peak points on the histogram serve as the grayscale characteristic of cluster center (e.g. Eigenvector V). Meanwhile, the number of initial cluster center is also determined by n. Thus, each pixel only needs to be compared with a few peak points, greatly improving the searching efficiency.

Calculating the gradient of the initial cluster center-defining the feature vector G: Generally, the gradient of pixels on the background and objective is relatively small, but the gradient of boundary points and noise points is larger. So the gradient of the initial cluster center of the background or objective can be set to zero, but for the rest of the cluster center, their gradient usually is the mean of the maximum gradient column on gradient image.

Calculating the neighborhood characteristics of the initial cluster center-defining the feature vector L: The neighborhood characteristics of the cluster center can be set according to the characteristics of pixel-neighborhood corresponding to different contents of laser range image. For example, those neighborhood characteristics of the cluster center owning zero gradients may be set to 8. And the cluster center of the boundary usually has a high gradient and a more pixel number corresponding to the gray. However, for the cluster center of the noise, the gradient is high but the pixel number is less.

Setting guiding function η_{ij}: After one cyclic process, the guiding function can reflect the similarity between pixels and cluster center, and is shown as follows:

$$\eta_{ij} = \frac{r}{\sqrt{\sum_{k=1}^{m} p_k (x_{ik} - c_{jk})^2}} . \tag{6}$$

Where, r is the clustering radius, and if it is greater, the clustering radius is also greater, and ants will be more likely to choose the cluster center.

3.2 Principle of Processing Different Regions

Processing principles of background and objective points: In order to make ants go away from the background and objectives as soon as possible and gather at the edge, based on the principle of ants' actual movement (e.g. [12]), the conversion rules of ants in this region is revised to random selection, to make ants move to the edge zone of image along a fixed direction having chosen. And because the pixels in this region have little effect on the edge detection, those pixels' pheromone quantity also has no influence on ants' transfer. Therefore the value of pheromone needs not to be updated.

Processing principles of noise points: The grayscale at the noise points usually mutates, and its gradient is also high. So we can distinguish them through using pixel-neighborhood. Those pixels in the 3×3 neighborhood whose grayscale value is close to these points in this region are generally considered to be noise points. Consequently, these points should be deleted during ants' transfer, and the selection mechanism should also be changed as follows:

Suppose that M is the collection of these pixels, and the transition probability:

$$p_{ij} = \begin{cases} \dfrac{ph_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{s\in S} ph_{is}^{\alpha}(t)\eta_{is}^{\beta}(t)}, & \text{if } j\in S \text{ and } j\notin M \\ 0, & \text{otherwise} \end{cases} . \tag{7}$$

Meanwhile, in order to reduce the probability of re-transferring to these noise points for other ants, the pheromone quantity of these noise points should be rapidly attenuated out after being visited by the first ant, and it no longer is released again. So the update mechanism is changed as follows:

$$ph_{ij}(t+1) = (1-\rho)ph_{ij}(t)\,. \tag{8}$$

Processing principles of edge points: Reference [13] shows that, for the purpose of improving the ants' Search capability at the edge point, the selection mechanism can be modified as following:

The grey value mutation at the boundary point is very obvious, and the gradient is also higher, the number of 3×3 neighbourhood pixels which have the similar gray level with the boundary point generally is greater than or equal to 6

$$j = \begin{cases} \underset{s\in S}{\arg\_\max}[ph_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)], & \text{if } q \le q_0 \\ S & \text{otherwise} \end{cases}\,. \tag{9}$$

Where, q is an random variable evenly distributing on $[0,1]$, q_0 is an adjustable parameter, S is the same with the former values.

If $q > q_0$, this selection mechanism is the same with the traditional method, the randomly selecting feature of probability can expand the ant's search space, and detect some minor edge effectively on the image.

If $q \le q_0$, there is no influence from the random factors of probability, so the search speed of ants can be greatly accelerated, and the efficiency of the algorithm can also be improved.

Therefore, through adjusting to affect the search, it not only can enhance the algorithm's execution efficiency, but also obtain more accurate edge.

$$p_{ij} = \begin{cases} \dfrac{ph_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum\limits_{s\in S} ph_{is}^{\alpha}(t)\eta_{is}^{\beta}(t)}, & \text{if } j\in S \\ 0, & \text{otherwise} \end{cases}\,. \tag{10}$$

4 Preliminary Experimental Result

In order to prove the practicability of this algorithm, the contrast experiment for edge detection of laser range image has been done. For simplicity a laser range image of real forest scene was selected as the experiment object, which has been photographed in Zhengzhou on December 2009. And the experiment results are shown as follows.

According to this experiment, the following conclusion can be reached: Firstly, the consuming-time of the ant colony optimization algorithm is shorter, which takes only 0.356s to detect the edge of image. However, the basic ant colony algorithm takes 1.403s and the Prewitt operator takes even longer. Secondly, the edge detection precision of this optimization algorithm is higher. Clearly, the continuity of edge detection in Fig.4 is better than that in Fig.2 and Fig.3. Thirdly, the error detection and omission factor of this algorithm are lower. In particular, the omission factor of Prewitt operator in Fig.2 is so high that the edges of trunks can not be presented. On the contrary, the profile of trunks is more clear and distinctive in Fig.4. Finally, the asperity of edge image detected by ant colony optimization algorithm is higher. Instead, the edge images in Fig.2 and Fig.3 are fuzzier and dimmer.

Generally speaking, Compared to the Prewitt operator and basic ant colony algorithm, the ant colony optimization algorithm has the advantages of short consuming-time, high detection accuracy, and low error detection and omission factor. In addition, different detection strategies can be applied based on different aspects of laser range imaging, so the adaptability of detecting edge can be enhanced and the sensitivity of noise can also be cut down.

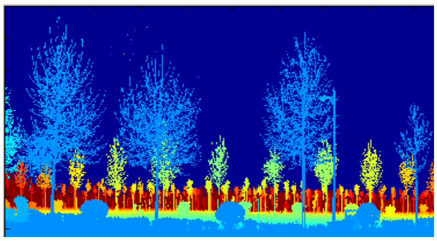


Fig. 1. Laser range image of real forest scene

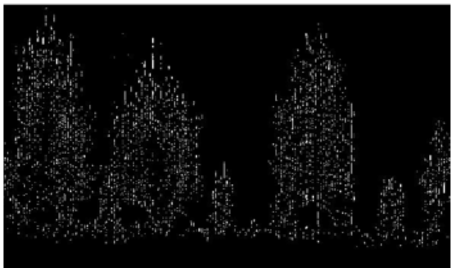


Fig. 2. Edge detection result of Prewitt operator

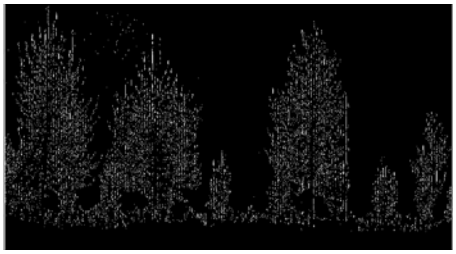


Fig. 3. Result of the basic ant colony algorithm

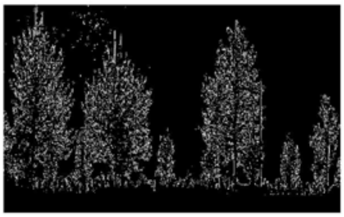


Fig. 4. Result of ant colony optimization algorithm

5 Conclusions

In this paper, according to the principles of basic ant colony algorithm, a fast adaptive ant colony algorithm for the edge detection of laser range image has been proposed, which is based on introducing the initial cluster center and heuristic guiding function, and applying ants' selection mechanism and update mechanism varying in contents. The contrast experiment has shown that, not only the algorithm's execution efficiency has been improved, but also its detection accuracy is greatly enhanced. In addition, this algorithm has good self-adaptability as well. In a word, the edge detection results of this algorithm can meet the requirements of three-dimensional reconstruction well.

References

1. Newman, P., Sibley, G., Smith, M., Cummins, M., Harrison, A., Mei, C., Posner, I., Shade, R., Schroeter, D., Murphy, L., et al.: Navigating, Recognizing and Describing Urban Spaces With Vision and Lasers. J. The International Journal of Robotics Research 28, 1406–1433 (2009)
2. Chevrier, C., Perrin, J.P.: Interactive parametric modeling: POG a tool the cultural heritage monument 3D reconstruction. In: CAADRIA conference, Chiang Mai (2008)
3. Kolomenkin, M., Shimshoni, I., Tal, A.: On edge detection on surfaces. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2767–2774 (2009)
4. Ren, C., Wu, S.-l., Jiao, L.-c.: Edge Detection Algorithm of SAR Images with Wedgelet Filter. J. Journal of Beijing Institute of Technology 3, 346–350 (2008)
5. Zhang, X.-h.: Airborne laser radar measurement theory and method, pp. 43–44. Wuhan University Press, Wuhan (2007)
6. Chen, L.C., Teo, T.A., Hsieh, C.H., Rau, J.Y.: Reconstruction of Building Models with Curvilinear Boundaries from Laser Scanner and Aerial Imagery. In: Chang, L.-W., Lie, W.-N. (eds.) PSIVT 2006. LNCS, vol. 4319, pp. 24–33. Springer, Heidelberg (2006)
7. Caldeira, J., Azevedo, R., Silva, C.A., Sousa, J.M.C.: CP with ACO. In: Supply-Chain Management Using ACO and Beam-ACO Algorithms, pp. 1–6. IEEE Press, Los Alamitos (2008)
8. Socha, K., Dorigo, M.: Ant colony optimization for continuous domains. J. European Journal of Operational Research (2006) (in press) (Corrected Proof)
9. Khichane, M., Albert, P., Solnon, C.: CP with ACO. In: Perron, L., Trick, M.A. (eds.) CPAIOR 2008. LNCS, vol. 5015, pp. 328–332. Springer, Heidelberg (2008)
10. Haibin, D.: Principle and application of ant colony algorithm, pp. 303–304. Science Press, Beijing (2005)
11. Ouadfel, S., Batouche, M.: Ant colony system to image texture classification. In: Proceedings of International Conference on Machine Learning and Cybernetics, pp. 1491–1495 (2003)
12. Leng, M., Yu, S.: An Effective Multi-level Algorithm Based on Ant Colony Optimization for Bisecting Graphs. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS (LNAI), vol. 4426, pp. 138–149. Springer, Heidelberg (2007)
13. Zhuang, X.-H.: Image feature extraction with the perceptual graph based on the ant colony system. In: Proceedings of the IEEE International Conference on Systems Man, and Cybeinetics (SMC 2004), Washington, DC, pp. 6354–6359. IEEE Computer Society, Los Alamitos (2004)

A Real-Time Moving Ant Estimator for Bearings-Only Tracking

Jihong Zhu[1], Benlian Xu[2], Fei Wang[1], and Zhiquan Wang[1]

[1] School of Automation, Nanjing University of Science and Technology, 210094 Nanjing, P.R. China
[2] Electrical and Automatic Engineering, Changshu Institute of Technology, 215500 Changshu, P.R. China

Abstract. A real-time moving ant estimator (RMAE) is developed for the bearings-only target tracking, in which ants located at their individual current state utilize the normalized weight and pheromone value to select the one-step prediction state and the dynamic moving velocity of each ant is depended directly on the normalized weights between two states. Besides this, two pheromone update strategy is implemented. Numerical simulations indicate that the RMAE could estimate adaptively the state of maneuvering or non-maneuvering target, and real-time requirement is superior to the moving ant estimator (MAE).

Keywords: Bearings-only, Ant colony optimization, Parameter estimation, Target tracking.

1 Introduction

Recursive state estimation of dynamic stochastic systems plays an important role in target tracking and many other engineering applications. As we know, Bayesian multi-target tracking propagates the multi-target posterior density recursively in time, but it involves the evaluation of multiple integrals. Fortunately, the particle filter (PF) [1-4], known as Sequential Monte Carlo (SMC) method, attracted more attentions in recent years over a wide range of applications. In PF, the target state variable distribution is approximated by a weighted set of samples. These samples are propagated and updated once new measurements are available. Using these samples, the targets state can be estimated by standard Monte Carlo integration techniques.

Ant Colony Optimization (ACO) utilizes artificial ants that cooperate to find good solutions for discrete or continuous optimization problems [5-6]. So far, there are few reports on the ACO algorithm employed into the parameter estimate field except [7-9]. The [9] proposed a moving ant estimator (MAE) to determine the parameter state. Four types of moving behaviors of ants are formulated, and the pheromone update process based on the current available measurement is also defined. However, the MAE shows computational burden due to the complex behaviors of moving ants and the strategy of pheromone update. To overcome this drawback, a real-time moving ant estimator (RMAE) based on the ACO is proposed. In this work, both pheromone value and heuristic value are considered and two case of the behavior of moving ants are formulated.

The remainder of this paper is organized as follows. Section 2 presents in detail the real-time moving ant estimator. In Section 3, numerical simulations are conducted and corresponding results are also analyzed. Finally, conclusions and future research directions are given in Section 4.

2 Real-Time Moving Ant Estimator (RMAE)

The general nonlinear non-Gaussian system model is considered and formulated as

$$\mathbf{x}(t) = f(\mathbf{x}(t-1), \mathbf{w}(t)) \tag{1}$$

$$\mathbf{z}(t) = h(\mathbf{x}(t), \mathbf{v}(t)) \tag{2}$$

where $\mathbf{x}(t)$ denotes the state of the system at time t; $\mathbf{z}(t)$ is the observation of the interested state; $\mathbf{w}(t)$ and $\mathbf{v}(t)$ are defined as the process noise and measurement noise with constant covariances $Q(t)$ and $R(t)$, respectively; and notations $f(\cdot)$ and $h(\cdot)$ represent the system and measurement models.

Similar to the PF, four estimate process phases are considered. The number of ants is fixed and equal to N. Without loss of generality, the moving behavior of a given ant k is considered, and other ants could follow the same manner. For the sake of discussion, the case of one dimensional parameter estimation such as in the x direction is considered， so the state vector $\mathbf{x}^{(k)}(t-1)$ of ant k at time $t-1$ can be represented by the position $x^{(k)}(t-1)$ and the velocity $\dot{x}^{(k)}(t-1)$ as $\mathbf{x}^{(k)}(t-1) = [x^{(k)}(t-1), \dot{x}^{(k)}(t-1)]'$, and its corresponding one-step prediction state vector $\mathbf{x}^{(k)}(t \mid t-1)$ is defined as $\mathbf{x}^{(k)}(t \mid t-1) = [x^{(k)}(t \mid t-1), \dot{x}^{(k)}(t \mid t-1)]'$.

In the first phase, assume that the state of ant k at time $t-1$ is known, the corresponding one-step prediction state is calculated according to Eq. (1), which constitutes a reference state at time t to be visited by itself or other ants at time $t-1$.

In the second phase, the weighted step is also kept. For all reference states are obtained in the first phase, the importance weights are normalized as

$$\bar{w}^{(k)}(t) = \frac{\tilde{w}^{(k)}(t)}{\sum_{j=1}^{N} \tilde{w}^{(j)}(t)} \tag{3}$$

$$\begin{aligned} \tilde{w}^{(k)}(t) &\triangleq \mathbb{N}(z(t), h(x^k(t \mid t-1)), R(t)) \\ &= \left|2\pi R(t)\right|^{\frac{1}{2}} e^{-\frac{1}{2}(z(t)-h(x^k(t|t-1)))' R^{-1}(t)(z(t)-h(x^k(t|t-1)))} \end{aligned} \tag{4}$$

where $\tilde{w}^{(k)}(t)$ is Gaussian assumption of measurement noise, and $\mathbb{N}(x, a, b)$ denotes the Gaussian distribution of x with mean a and variance b.

In the third phase, the moving velocity of each ant is determined. As mentioned above, the first and second phases construct the search space of all ants, as shown in Fig.1, the ant k will select the predicted state of ant j as its further moving direction based on the following defined probabilistic decision

$$p_{k,j} = \frac{\overline{w}^{(j)}(t)\tau_{kj}(t)}{\sum_{j\neq k}\overline{w}^{(j)}(t)\tau_{kj}(t)} \tag{5}$$

where $\tau_{kj}(t)$ is the pheromone value between state vector $\mathbf{x}^{(k)}(t-1)$ and $\mathbf{x}^{(j)}(t\mid t-1)$. The difference between Eq.(5) and that in MAE is introduction of pheromone, which will avoid the degeneracy phenomenon, i.e., the ant will not always select the predicted state with the biggest weight. It is obvious that such a selection process is a stochastic one, and for those one-step predicted states with larger normalized weights and higher pheromone value will have more chances to be selected. The state is selected by ant constitute an approximation of posterior probability density of the current interested state.

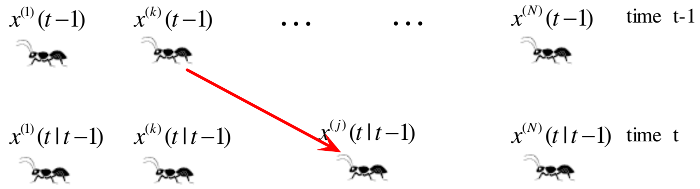


Fig. 1. Moving behavior of ant from one time to another

If the position $x^{(j)}(t\mid t-1)$ is chosen as the destination direction of ant k starting from time $t-1$, two types of moving behavior models are investigated.

- $\overline{w}^{(j)}(t) > \overline{w}^{(k)}(t)$

It means that the weight value of the state to be selected is bigger than that of current state, so ant is preferred to move toward. When ant arrives at its destination, the current state is updated based on the moving velocity of ant. In order to describe this phenomenon, the corresponding ant behavior is modeled by

$$\dot{x}^{(k)}(t) = \frac{x^{(j)}(t\mid t-1) - x^{(k)}(t-1)}{T} \tag{6}$$

$$x^{(k)}(t) = x^{(k)}(t-1) + \dot{x}^{(k)}(t)\cdot T \tag{7}$$

where T is the sampling interval. The Eq. (6) denotes that the ant k moves from state $x^{(k)}(t-1)$ to $x^{(j)}(t\mid t-1)$ with maximum velocity. Moreover $\dot{x}^{(k)}(t)$ is a vector, for example, if the state $x^{(j)}(t\mid t-1)$ is smaller than the state $x^{(k)}(t-1)$, the value of $\dot{x}^{(k)}(t)$ will be a negative, i.e., $\dot{x}^{(k)}(t) < 0$. That is to say the update state will be located at the left of the current state in x direction.

- $\overline{w}^{(j)}(t) \leq \overline{w}^{(k)}(t)$

In this model, the ant k will not move from state $x^{(k)}(t-1)$ to $x^{(j)}(t\mid t-1)$ with maximum velocity. The corresponding ant behavior is modeled by

$$\dot{x}^{(k)}(t) = \dot{x}^{(k)}(t-1) - \left(\frac{x^{(j)}(t\,|\,t-1) - x^{(k)}(t-1)}{T}\right) \cdot G_{kj}(t) \tag{8}$$

$$x^{(k)}(t) = x^{(k)}(t-1) - \dot{x}^{(k)}(t) \cdot T \tag{9}$$

where $G_{kj}(t)$ is velocity adjusting coefficient, which depends on the weight $w^{(k)}(t)$ and $w^{(j)}(t)$. The $G_{kj}(t)$ is defined as

$$G_{kj}(t) = 1 - \exp(-\lambda \cdot (w^{(j)}(t) - w^{(k)}(t))^2) \tag{10}$$

where λ is a constant value. It can be seen that if the weight $w^{(j)}(t)$ of selected state is much larger than the weight $w^{(k)}(t)$, the value of $G_{kj}(t)$ will be close to 1, i.e., the velocity of ant k is close to the maximum velocity, so the update position of ant k has a significant change according to the Eq. (8). And the update of position of ant k will change a little, and vice versa.

In the fourth phase, the pheromone update process is executed and modeled by

$$\tau_{kj}(t+1) = \tau_{kj}(t) - \frac{G_{kj}(t)}{\varepsilon} \cdot \tau_{kj}(t) \tag{11}$$

$$\tau_{kj}(t+1) = \tau_{kj}(t) + \frac{1}{\varepsilon} \cdot \tau_{kj}(t) \tag{12}$$

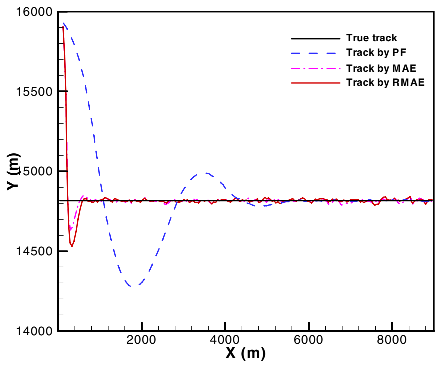
where $\varepsilon = \sum_{k \neq j} G_{kj}(t)$ denotes the total pheromone value deposited on the trail between state $x^{(k)}(t-1)$ and state $x^{(j)}(t\,|\,t-1)$. It is observed that Eq. (11) represents the indirect pheromone evaporate, which means that the pheromone value on the trail between state $x^{(k)}(t-1)$ and state $x^{(j)}(t\,|\,t-1)$ will decrease, if the state $x^{(j)}(t\,|\,t-1)$ is selected by ant k. And Eq. (12) shows the increase of pheromone value with amount of $\frac{1}{\varepsilon}$. Note that the pheromone update strategy is only depended on the velocity adjusting coefficient, which is different with that in MAE.

3 Numerical Simulations and Comparisons

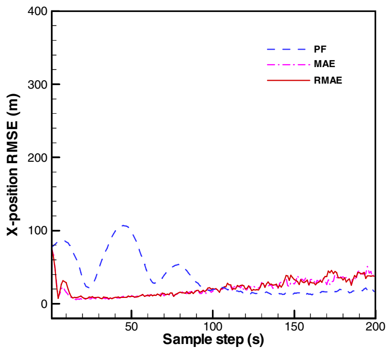
Two stationary observers are located at $(0,0)$ and $(-2000,0)$ respectively in a surveillance region. The standard deviation of the bearing measurements for each observer is taken as 0.02°, and the sampling interval is set to $T = 1$ s. Other parameters are selected as follows: $N = 100$, $\lambda = 10$, $P_0 = diag(1000,100,1000,100)$,

$$Q_1(t) = diag(0.02^2, 0.01^2, 0.02^2, 0.01^2),\ Q_2(t) = diag(0.04^2, 0.02^2, 0.04^2, 0.02^2).$$

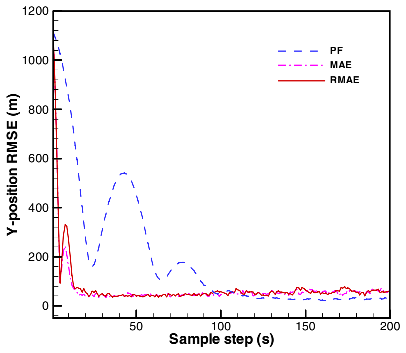
Two scenarios are presented to demonstrate the performance of the proposed RMAE. In the first we provide an example of tracking non-maneuvering target, and performance



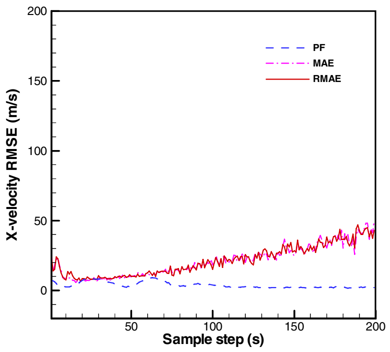
(a) Comparison of trajectories estimated by different techniques



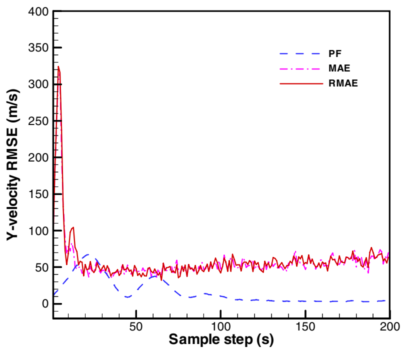
(b) Position RMSE in X axis



(c) Position RMSE in Y axis



(d) Velocity RMSE in X axis



(e) Velocity RMSE in Y axis

Fig. 2. Performance comparison with PF and MAE in scenario 1

comparisons are conducted with PF and MAE. In the second scenario we give an example of maneuvering target tracking, and present the comparative simulations with a standard interacting multiple model particle filter (IMMPF) and MAE.

Scenario 1: The target makes a uniform rectilinear motion with a given initial state $x(0) = [0m, 45m/s, 14816m, 0m/s]'$.

Fig.2 shows the performance comparison with PF and MAE after 50 Monte Carlo runs. It is observed that the trajectory estimated by the RMAE fits well close to the true one in Fig.2 (a). The velocity root mean squared errors (RMSEs) of the RMAE are closed to the MAE in Fig.2 (d) and (e), but are worse than that of the PF, and the related position RMSEs are closed to the MAE and competitive with that of PF in Fig.2 (b) and (c). Therefore, for scenario 1 we can conclude that the RMAE works well in contrast to the PF and MAE.

Table 1 shows the comparison of maximum execution time with PF and MAE under different particles or ants number. It can be seen that the biggest maximum execution time of RMAE (**0.3639** s) has a significant improvement compared with that of MAE.

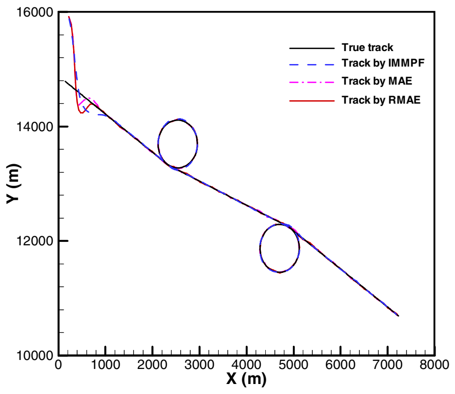
Table 1. Comparison of Maximum execution time with different techniques in scenario 1

N	PF	MAE	RMAE
50	0.0331	0.5402	**0.1667**
100	0.0554	1.4597	**0.2130**
200	0.1579	5.1901	**0.3639**

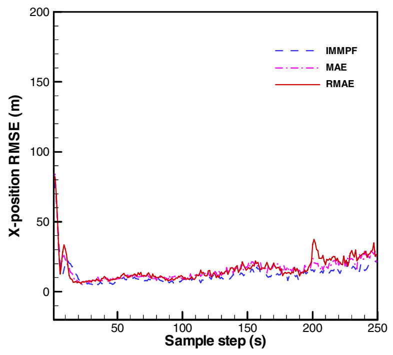
Scenario 2: Firstly, the target makes a uniform rectilinear motion with an initial state $x(0) = [100m, 45m/s, 14816m, -30m/s]'$ during the first 50 samples phase, and then the target executes a constant rate turn with $a_n = 7m/s^2$, which lasts 50 samples. After the maneuver the target maintains a constant velocity lasting about 50 samples, and then the target turns with a constant rate turn with $a_n = -7m/s^2$ for another 50 samples. Finally, the target resumes a nearly constant velocity motion, with the velocity it had attained at the end of the previous phase for another 50 samples.

The Fig.3 shows the performance comparison with the IMMPF and MAE. It is obviously that the trajectory estimated by the RMAE fits well close to the true one in Fig.3(a), the velocity RMSEs in X axis of the RMAE are closed to that of the IMMFP and MAE in Fig.3 (d) and that in Y axis is higher than IMMPF in Fig.3 (e), and the related position RMSEs are closed to the IMMPF and MAE in Fig.3 (b) and (c). Therefore, for scenario 2 we can conclude that the RMAE works well in contrast to the PF and IMMPF.

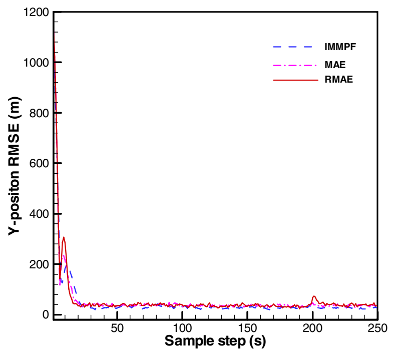
Table 2 shows the comparison of maximum execution time with IMMPF and MAE under different particles or ants number after 50 Monte Carlo runs. It is observed that the biggest maximum execution time of RMAE is smaller than the IMMPF, when the number of ants is set to be 50 and 100. But when the number of ants is set to be 200, the biggest maximum execution time of RMAE (**81.4886**s) is bigger than that of IMMPF, but it is still much less than that of MAE.



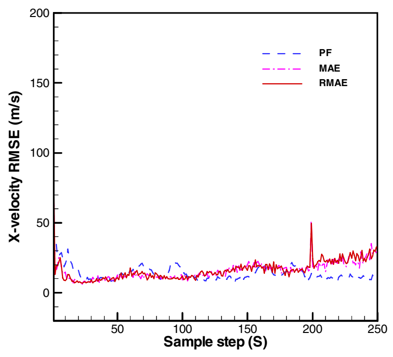
(a) Comparison of trajectories estimated by different techniques



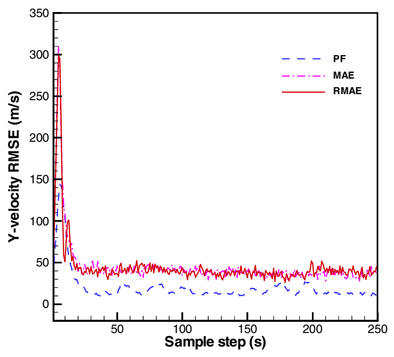
(b) Position RMSE in X axis



(c) Position RMSE in Y axis



(d) Velocity RMSE in X axis



(e) Velocity RMSE in Y axis

Fig. 3. Performance comparison with IMMPF and MAE in scenario 2

Table 2. Comparison of Maximum execution time with different techniques in scenario 2

N	IMMPF	MAE	RMAE
50	17.2922	89.7135	**11.2465**
100	33.3986	319.1860	**26.3822**
200	65.9872	1031.3842	**81.4886**

4 Conclusions

A real-time moving ant estimator is proposed in this paper, which is implemented by regulating moving behavior of a set of ants with dynamic moving velocity. Numerical simulation results show that the proposed RMAE is capable of tracking not only the non-maneuvering target but also the maneuvering one. Meanwhile, the maximum execution time is compared among different techniques, and the comparison results indicate that the proposed RMAE could satisfy the real-time requirement.

Acknowledgement. This work is supported by National Natural Science Foundation of China (No. 60804068).

References

1. Julier, S.J., Uhlmann, J.K.: A new extension of the Kalman filter to nonlinear systems. In: Proceedings of Aerosense: The 11th International Symposium on Aerospace/Defense Sensing, Simulation and Controls, vol. 3068, pp. 182–193 (1997)
2. Särkkä, S., Vehtari, A., Lampinem, J.: Rao-Blackwellized particle filter for multiple target tracking. Information Fusion 8, 2–15 (2007)
3. Yu, Y.H., Cheng, Q.S.: Particle filter for maneuvering target tracking problem. Signal Processing 86, 195–203 (2006)
4. Carpenter, J., Clifford, P., Fearnhead, P.: Improved particle filter for nonlinear problems. IEE Proceedings of Radar, Sonar and Navigation 146(1), 2–7 (1999)
5. Dorigo, M., Maniezzo, V., Colorni, A.: The ant system: optimization by a colony of cooperating agents. IEEE Transactions on System, Man and Cybernetics-Part B 26(1), 29–42 (1996)
6. Bonabeau, E., Dorigo, M., Theraulaz, G.: From nature to artificial intelligence. Oxford University Press, New York (1999)
7. Nolle, L.: On a novel ACO-estimator and its application to the target motion analysis problem. Knowledge-Based Systems 21(3), 225–231 (2008)
8. Xu, B.L., Chen, Q.L., Zhu, J.H., Wang, Z.Q.: Ant estimator and its application to target tracking. Signal Processing (2009) (in press)
9. Xu, B.L., Chen, Q.L., Wang, X.Y., Zhu, J.H.: A novel estimator with moving ants. Simulation Modelling Practice and Theory 17, 1663–1677 (2009)

Two-Stage Inter-Cell Layout Design for Cellular Manufacturing by Using Ant Colony Optimization Algorithms

Bo Xing[1,*], Wen-jing Gao[1], Fulufhelo V. Nelwamondo[2], Kimberly Battle[1], and Tshilidzi Marwala[1]

[1] Faculty of Engineering and the Built Environment, University of Johannesburg, Auckland Park, 2006, Johannesburg, South Africa
bxing2009@gmail.com
[2] Biometrics Research Group, Modelling and Digital Science Unit, Council for Science and Industrial Research (CSIR), 0001, Pretoria, South Africa

Abstract. Facility layout planning plays an important role in the manufacturing process and seriously impacts a company's profitability. A well-planned layout can significantly reduce the total material handling cost. The purpose of this paper is to develop a two-stage inter-cell layout optimization approach by using one of the popular meta-heuristics — the Ant Colony Optimization algorithm. At the first stage, the cells are formed based on the part-machine clustering results obtained through the ant system algorithm. In other words, we get the initial inter-cell layout after this stage. The work at the second stage uses a hybrid ant system algorithm to improve the solution obtained at previous stage. Different performance measures are also employed in this paper to evaluate the results.

Keywords: cellular manufacturing (CM), cell formation (CF), inter-cell layout (ICL), ant colony optimization (ACO), quadratic assignment problem (QAP), material handling cost.

1 Introduction

Cellular manufacturing (CM) has been recognized as an efficient and effective way to improve productivity in a factory. In recent years, there have been continuous research efforts to study different facets of CM systems. Among them, inter-cell layout (ICL) is one of the most frequently used representations of the CM system design. The ICL problem, in the context of CM, is the allocation of the manufacturing cells to areas within a shop-floor.

Currently, there are many methodologies available in the literature dealing with the ICL problem. Among them, ant colony optimization (ACO) is a relatively recently developed method for the solution of combinatorial optimization problems. According to Dorigo [1], ACO takes inspirations from the foraging behavior of some real ant species. These ants deposit a substance called pheromone on the ground in order to mark some favorable path that could be followed by other members of the colony. Up to date, there are a series of successful ACO variants that exist such as ant system (AS)

[2], Ant-Q [3], MAX-MIN ant system (MMAS) [4], and ant colony system (ACS) [5]. In this paper, two different ACO algorithms are investigated and employed for designing and optimizing the ICL.

The structure of this paper is as follows: Section 2 elaborates two-stage ICL design for CM. First we use AS algorithm to do the cell formation (CF) task which will generate the initial ICL; and then, a hybrid AS algorithm is used to address the inter-cell re-layout (ICRL) issues; Section 3 summaries the conclusions of this paper.

2 Description of the Proposed Two-Stage Approach

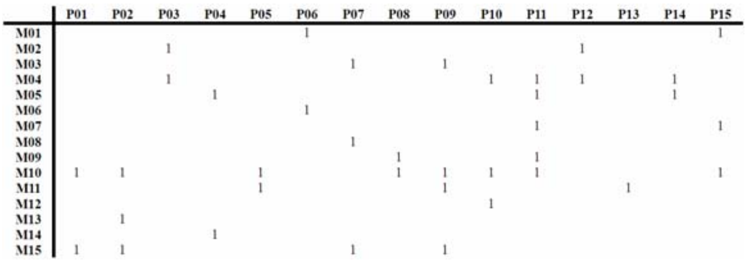
2.1 Stage I: Generating Initial Inter-Cell Layout through Cell Formation

Cellular manufacturing (CM) is the application of group technology to manufacturing. The basic idea behind group technology is to group parts which have similar processing requirements or similar design features into part-families; while the corresponding machines are assigned to machine-groups. This process is often referred to as part-machine clustering or cell formation (CF) in the literature. In this paper, we translate part-machine incidence matrix (PMIM) to distance matrix so that CF problem can be modeled as traveling salesman problem (TSP) and as a result AS is the first option at this stage.

2.1.1 Ant System (AS) Algorithm and Its Application to Cell Formation

Ant system (AS) is the first ACO algorithm and it was initially proposed in the early 1990s [2]. Its main success was in solving the classical TSP. In general, the application of AS to CF problem can be done through the following steps:

Step 1: Input the PMIM. For example we have a 15×15 input matrix (see Fig. 1).



	P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P11	P12	P13	P14	P15
M01						1									1
M02			1									1			
M03							1		1						
M04			1							1	1	1		1	
M05				1							1			1	
M06						1									
M07											1				1
M08							1								
M09								1			1				
M10	1	1			1			1	1	1	1				1
M11					1				1				1		
M12										1					
M13		1													
M14				1											
M15	1	1					1		1						

Fig. 1. PMIM (size 15×15)

Step 2: Generating the distance matrix between all the parts and machines included in the PMIM by using dissimilarity coefficient [6].

Step 3: Input the machine or part distance matrix to the AS module and start looking for the shortest Hamiltonian path for machines or parts.

In this step, during the construction of a solution, ants select the following machine or part to be visited through a stochastic mechanism. The transition probability of going from machine i to machine j (or part i to part j) for the k-th ant is calculated as follows:

$$p_{ij}^{k} = \begin{cases} \tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta} \Big/ \sum\limits_{c_{il} \in \mathbf{N}(s^{p})} \tau_{il}^{\alpha} \cdot \eta_{il}^{\beta} & \text{If } c_{ij} \in \mathbf{N}(s^{p}) \\ 0 & \text{Otherwise} \end{cases} . \tag{1}$$

where $\mathbf{N}(s^{p})$ is the set of feasible components; that is, edges (*i, l*) where *l* is a machine/part not visited yet by the ant *k*; $\eta_{ij} = 1/d_{ij}$ is a heuristic value that available a priori; and the parameters α and β control the relative importance of the pheromone versus the heuristic information, which is given by η_{ij}.

In AS algorithm, the pheromone values are updated when all the *m* ants have built solutions after their iteration. The pheromone intensity associated with the edge joining machines *i* and *j* (or parts *i* and *j*) is updated as follows:

$$\tau_{ij} \leftarrow (1-\rho) \cdot \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \ . \tag{2}$$

where

ρ is the pheromone evaporation rate;

m is the number of ants;

$\Delta\tau_{ij}^{k}$ is the quantity of pheromone laid on edge (i, j) by k-th ant;

$$\Delta\tau_{ij}^{k} = \begin{cases} Q/L_k & \text{If ant } k \text{ used connection } (i,j) \text{ in its tour} \\ 0 & \text{Otherwise} \end{cases} ;$$

Q is a constant;

L_k is the length of the tour constructed by k-th ant.

Step 4: Repeat Step 3 for obtaining the best machine or part sequence which can minimize the sum of dissimilarities between machines or parts.

Step 5: Rearrange the initial matrix according to the best machine and part sequence. In our example, the final clustering result is illustrated below.

		P07	P11	P05	P08	P09	P13	P01	P02	P12	P10	P03	P14	P04	P06	P15
Cell 1	M08	1														
	M10		1	1	1	1		1	1		1					1
	M09		1		1											
Cell 2	M11			1		1	1									
	M03	1				1		1								
Cell 3	M13								1							
	M15	1				1		1	1							
Cell 4	M12										1					
	M04		1							1	1	1	1			
	M02									1		1				
Cell 5	M14													1		
	M05		1										1	1		
Cell 6	M06														1	
	M07		1													1
	M01														1	1

Fig. 2. Final Clustering Result

2.1.2 Cell Formation Performance Measure

Although grouping efficiency [7] is the most widely used CF performance measure, it has been reported with a lower discriminating power. Even an extremely bad solution with large number of exceptional elements has a grouping efficiency value as high as 77% [8]. In order to overcome the drawbacks of grouping efficiency, in this paper we adopt another performance measure called group efficacy which was proposed by Kumar and Chandrasekharan [9]. It can be formulated as the following:

$$\text{Group Efficacy} = (1\text{-}\varphi)/(1+\phi) \ . \tag{3}$$

where φ is the ratio of the number of exceptional elements to the total number of elements; ϕ is the ratio of the number of zeros in the region of diagonal blocks to the total number of elements. In our case, the value of group efficacy is about 88.29%.

2.2 Stage II: Inter-Cell Re-Layout (ICRL) Design

The ICRL design problem with equal size of cells can be formulated as a quadratic assignment problem (QAP). The QAP, introduced by Koopmans and Beckman [10] in 1957, is always considered as one of the hardest optimization problems. Normally, the QAP can be described as follows: given $m\times m$ matrices $\mathbf{A} = [a_{ij}]$ and $\mathbf{B} = [b_{ij}]$, find a permutation π^* minimizing

$$\min_{\pi\in\prod(m)} f(\pi)=\sum_{i=1}^{m}\sum_{j=1}^{m} a_{ij}\bullet b_{\pi_i\pi_j} \ . \tag{4}$$

where $\prod(m)$ is the set of permutation of m elements. Shani and Gozelez [11] proved that the QAP problem is *NP*-hard. Even instances of relatively small size of $n \geq 25$ cannot be solved exactly [12].

Normally, the QAP-based ICRL problem can be formulated as follows:

$$\text{TC} = \sum_{i=1}^{N}\sum_{k=1}^{N}\sum_{j=1}^{N}\sum_{l=1}^{N} f_{ik} d_{jl} x_{ij} x_{kl} \ . \tag{5}$$

subject to:

$$\sum_{j=1}^{N} x_{ij} = 1, \ i = 1,...,N \ . \tag{6}$$

$$\sum_{i=1}^{N} x_{ij} = 1, \ j = 1,...,N \ . \tag{7}$$

$$x_{ij} = \{0,1\}, \ i = 1,...,N; \ j = 1,...,N \ . \tag{8}$$

where:

TC=Total Material Handling Cost

i, k=Cells in the layout;

j, l=Locations in the layout;

N=Number of cells and locations;

f_{ik} = Material flow data betwee cell i and cell k;

d_{jl} = Distance between locations j and l;

$$x_{ij} = \begin{cases} 1 & \text{If cell } i \text{ is assigned to location } j \\ 0 & \text{Otherwise} \end{cases}.$$

The Equation (5) is used to minimize the total material handling cost among cells; while constraint set (6) ensures that one location is assigned to each cell; and constraint set (7) ensures that exactly one cell is assigned to each location.

2.2.1 Hybrid Ant System and Its Application to Inter-Cell Re-layout Problem

Hybrid Ant System for the QAP (HAS-QAP for short) algorithm was developed by Gambardella *et al.* [13]. HAS-QAP algorithm departs radically from the previously described AS algorithm in Section 2.1.1. Basically, there are two main novelties of HAS-QAP algorithm: (i) Ants modify solutions as opposed to building them; (ii) Pheromone trails are used to guide the modifications of solutions, and not as an aid to direct their construction.

Since we have already got an initial ICL solution after Stage I, so at this stage, HAS-QAP algorithm is the most appropriate approach to improve this solution. In other words, the main characteristics of HAS-QAP make it suitable for solving the QAP-based ICRL problem. Overall, the implementation of HAS-QAP algorithm to ICRL design can be divided into the following steps:

Step 1: *Solution Initialization*. At this step, m (m = total number of ants) random initial permutations $\pi^1_{(1)}, \ldots, \pi^m_{(1)}$ will be generated and each one of these permutations will be associated to an ant.

In this paper ten ants (m = 10) are chosen and the initial ICL is shown in Fig. 3, where cells 1, 2, 3, 4, 5, and 6 are assigned to locations A, B, C, D, E, and F respectively. So for any k-th ant, we have the following: $\pi^k_{(1)}$ =(Cell 1, Cell 2, Cell 3, Cell 4, Cell 5, and Cell 6).

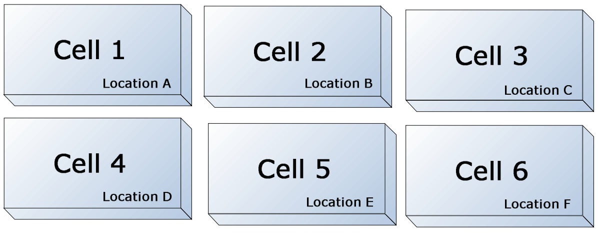


Fig. 3. Initial ICL for Six Cells

According to the obtained CF result (see Fig. 2), we can get a total material flow data matrix which is shown in Fig. 4. Meanwhile the distances between the locations are also given in Fig. 5. Based on these, the total material handling cost (distance-based) for the initial ICL can be calculated. By using equation (5), the initial total material handling cost for the initial ICL is TC_i =44.

From/To	Cell 1	Cell 2	Cell 3	Cell 4	Cell 5	Cell 6
Cell 1	0	3	3	2	1	2
Cell 2	3	0	2	0	0	0
Cell 3	3	2	0	0	0	0
Cell 4	2	0	0	0	1	0
Cell 5	1	0	0	1	0	0
Cell 6	2	0	0	0	0	0

Fig. 4. Material Flow Data Matrix

	Location A	Location B	Location C	Location D	Location E	Location F
Location A	0	1	2	1	2	3
Location B	1	0	1	2	1	2
Location C	2	1	0	3	2	1
Location D	1	2	3	0	1	2
Location E	2	1	2	1	0	1
Location F	3	2	1	2	1	0

Fig. 5. Distance Matrix

Step 2: *Improving* $\pi^{1}_{(1)},\ldots,\pi^{m}_{(1)}$ *with the Local Search Procedure*. Let π^* be the best solution. The purpose of this step is to improve the initial solution so that the best solution π^* can be used in the next step for the pheromone trail matrix initialization.

In this research, we use pair-wise exchange heuristics to improve the initial ICL plan for each ant and let π^* represent the best solution. This step needs to be repeated for 6×6=36 iterations in our case.

Step 3: *Pheromone Trail Matrix Initialization*. The pheromone trail matrix **T**=[τ_{ij}] is initialized through the following equation:

$$\tau_0 = 1/Q \bullet TC(\pi^*) \ . \tag{9}$$

where Q is the pheromone initialization parameter, and $TC(\pi^*)$ is the current best total material handling cost found. Note no information is contained in the pheromone trail matrix, so all pheromone trails τ_{ij} are set to the same value τ_0 .

Step 4: *Starting the Main Loop*. Define I_{max}, which represents the total number of iterations performed and initialize all the other parameters (i.e., Q, R, S, m, q, α_1 and α_2). See [13] for more details about the parameters' setting. After I_{max} iterations have been performed, the stopping criterion will terminate the algorithm.

Step 5: *Pheromone Trail Based Modification.* In this step, a new solution $\hat{\pi}^k$ is generated for each ant by considering R pheromone trail swaps. Swaps are performed as follows: first, a location index r is chosen randomly between 1 to n; next, the second location index s ($s \neq r$) is chosen between 1 to n as well; and then the elements π_r^k and π_s^k are swapped in each ant's current solution π^k . The second index is selected using one of the two policies presented below:

$$\text{Policy I: Max } \tau_{r\pi_s}^k + \tau_{s\pi_r}^k$$
$$\text{Policy II: } \tau_{r\pi_s}^k + \tau_{s\pi_r}^k \Big/ \sum_{j \neq r} \tau_{r\pi_j}^k + \tau_{j\pi_r}^k \ . \tag{10}$$

Note for the first iteration, elements π_r^k and π_s^k are randomly selected, since all the entries in the pheromone trail matrix are the same.

Step 6: *Improving the Current Solution using Local Search.* The solution $\hat{\pi}^k$ generated in Step 5 is improved using the pair-wise heuristics. The improved solution is denoted as $\tilde{\pi}^k$. This step will be repeated for 6×6=36 iterations in this instance.

Step 7: *Performing Intensification Mechanism.* This step is used to explore the neighborhood of good solutions more thoroughly. If the best solution has been improved, the intensification mechanism will be activated and the ant will compare the solutions between π^k and $\tilde{\pi}^k$ before it starts the next iteration; otherwise the intensification strategy will not be activated and the ant starts the next iteration with the solution $\tilde{\pi}^k$. The intensification strategy remains active while at least one ant improves its solution during an iteration.

Step 8: *Updating the Pheromone Trail Matrix.* To speed-up the convergence, the pheromone trail matrix is updated by considering only the best solution found so far. First, all the pheromone trails are evaporated by setting:

$$\tau_{ij} = (1 - \alpha_1) \bullet \tau_{ij} \ . \tag{11}$$

where $1 \leq i, j \leq n$ and $0 < \alpha_1 < 1$. Here α_1 is a parameter that controls the evaporation of pheromone trails. A value of α_1 close to 0 implies that the pheromone trails remain active for a long time; while a value close to 1 implies a high degree of evaporation and a shorter memory of the system.

Then, the pheromone trails contained in the best solution π^* are reinforced by setting:

$$\tau_{i\pi_i^*} = \tau_{i\pi_i^*} + \alpha_2 / TC(\pi^*) \ . \tag{12}$$

where α_2 is a parameter that controls the reinforcement of the pheromone trails.

Step 9: *Performing Diversification Mechanism.* The diversification mechanism is activated after S consecutive iterations with no improvement to the best solution obtained. Once this mechanism is activated, all the information such as pheromone trail matrix and solutions are deleted and the algorithm is started from the beginning, in which only the best solution obtained from an ant is used in the next iteration.

After the execution of the algorithm, we get an optimized layout solution where cells 2, 3, 5, 6, 1, and 4 have been re-assigned to locations A, B, C, D, E, and F respectively (see Fig. 6). Under this new ICL plan, the total material handling cost TC_f is further reduced to 18.

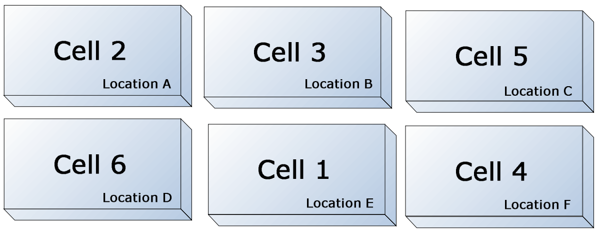


Fig. 6. Optimized ICL for Six Cells

2.2.2 ICRL Performance Measure

Often, there are some requirements or constraints that have to be satisfied during the process of layout design, and the layouts generated need to be evaluated by a preselected function which indicates their efficiency. In a layout design, material handling cost is one of the most commonly used criterion to determine the efficiency and effectiveness of ICRL. According to Shang [14], the material handling cost can comprise between 30 and 70 percent of the total manufacturing cost. So reducing the total material handling cost is the main objective of the ICRL design.

In order to provide some means of quantifying the efficiency of the final layout, the following criterion [15] is used in this paper:

$$\text{GT Efficiency}=\left(1-\frac{\mathrm{TC}_f}{\mathrm{TC}_i}\right)\bullet 100\% \ . \tag{13}$$

where TC_f is the final inter-cell material handling cost and TC_i is the initial inter-cell material handling cost.

This measure reflects the percentage of savings in material handling that resulted from the cell re-layout. The higher the value of this criterion, the better the solution is. For the instance in this paper, we have TC_f =18 and TC_i =44, so the final GT efficiency is 59.09%.

3 Conclusions

One of the major contributions of this work is to use two-stage approach for designing and optimizing ICL design. At Stage I, the PMIM is clustered and rearranged. Cells are formed and the initial ICL is generated after this stage. And then, at Stage II, the ICL is refined (i.e., ICRL design). Once we get the final optimized layout plan at the end of this stage, the total material handling cost can be further reduced.

Another contribution of this paper is the HAS-QAP algorithm has been employed to improve the performance of AS algorithm in layout design. First, by interpreting the part-machine clustering problem as a TSP, we find the shortest distance between machines and parts through AS algorithm. Although a layout is available now, the permutation of cells does not give a guarantee of the optimal total material handling cost, so we then use the HAS-QAP algorithm to modify this initial solution and to get an improved solution.

At last, group efficacy and GT efficiency are exploited as the performance measures in this research. Group efficacy evaluates the goodness of the CF result; and the effectiveness of the ICRL plan is measured by GT efficiency. According to the evaluation values, both results are good and meet our design purpose.

References

1. Dorigo, M., Socha, K.: An introduction to ant colony optimization. IRIDIA, Universite Libre de Bruxelles, Belgium (2007)
2. Dorigo, M., Maniezzo, V., Colorni, A.: The ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics-Part B 26, 1–13 (1996)
3. Gambardella, L.M., Dorigo, M.: Ant-Q: a reinforcement learning approach to the traveling salesman problem. In: Twelfth International Conference on Machine Learning, Tahoe City, CA (1995)
4. Stützle, T., Hoos, H.H.: MAX-MIN ant system. Future Generation Computer Systems 16, 889–914 (2000)
5. Dorigo, M., Gambardella, L.M.: Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation 1, 53–66 (1997)
6. Cheng, K.: Applying Ant Colony Optimization to Rearranged Matrix Partitioning Problems for cellular manufacturing Department of Information Management, Master of Science. Tatung University (2006)
7. Chandrasekharan, M.P., Rajagopalan, R.: An ideal seed non-hierarchical clustering algorithm for cellular manufacturing. International Journal of Production Research 24, 451–464 (1986)
8. Yin, Y., Yasuda, K.: Similarity coefficient methods applied to the cell formation problem: a comparative investigation. Computers & Industrial Engineering 48, 471–489 (2005)
9. Kumar, C.S., Chandasekharan, M.P.: Grouping efficacy: a quantitative criterion for goodness of block diagonal forms of binary matrices in group technology. International Journal of Production Research 28, 233–243 (1990)
10. Koopmans, T.C., Beckmann, M.J.: Assignment problems and the location of economics ativities. Econometrica 25, 53–76 (1957)
11. Shani, S., Gonzalez, T.: P-complete approximation problems. Journal of ACM 23, 555–565 (1976)
12. Engelbrecht, A.P.: Computational intelligence: an introduction. John Wiley & Sons Ltd., West Sussex (2007)
13. Gambardella, L.M., Taillard, E., Dorigo, M.: Ant colonies for the quadratic assignment problem. Journal of Operational Research Society 50, 167–176 (1999)
14. Shang, J.: Ant colony heuristics for the dynamic facility layout problem. Department of Industrial and Management Systems Engineering, Master of Science. West Virginia University, Morgantown, West Virginia (2002)
15. Nagi, R., Harhalakis, G., Proth, J.M.: Multiple routings and capacity considerations in group technology applications. International Journal of Production Research 28, 22–43 (1990)

Images Boundary Extraction Based on Curve Evolution and Ant Colony Algorithm

JinJiang Li[1,2], Da Yuan[1], Zhen Hua[1], and Hui Fan[1]

[1] School of Computer Science and Technology, Shandong Institute of Business and Technology, 264005, Yantai, China
[2] School of Computer Science and Technology, Shandong University, 250101, Jinan, China

Abstract. A new boundary contour extraction algorithm based on curve evolution model and ant colony algorithm is proposed in this paper. Firstly, ant colony algorithm is used to find the optima of snake points for rapidly converging near image edge. Then the interpolation algorithm is applied to gaining the object's rough contour that is used as the initial zero level set. The accurate contour can be obtained by the curve evolution method. Experimental results are given to demonstrate the feasibility of the proposed method in extracting contour from the blurred edge and high-noise images.

Keywords: Boundary extraction, Ant colony algorithm, Curve evolution, Mean-shift.

1 Introduction

Curve evolution is more accurate image processing method, widely used in medical image processing, computer vision and other fields. A great deal of research is made on curve evolution model and it's numerical, a number of representative methods are proposed. For example, Kass and Witkin, based least energy, give a parameters curve method (snake) [1]; Caselles proposed deformation geometry (geometric active contour) and geodesic (Geodesic Active Contours) curve evolution model [2] [3]; Osher and Sethian raised the level of set theory [4]. Mumford-Shah variation model [5] (referred to as the model for MS) is from 1989 proposed by Mumford and Shah, and the model has adaptive capacity for evolution curve topology analysis. The model, adopting the variations method, turns the problem of image contour extraction into a functional extreme value problem. Its energy function included in the border region and outline description of the image. As the MS is a model of modern mathematics in a freedom and not continuous issues, the function easily fall into the local minimum, which difficult to get numerical value in practical applications. CV model is a simplified model of MS give by Chan and Vese [6], based on the method in literature [5]. The energy evolution function does not rely on images gradient, but on the region. This method is suitable for gradient meaningful or meaningless contour extraction, also for the fuzzy or not continuous edge. The literature [6] define the energy function based on the level set, solving the evolution of partial differential equations (PDE) by Euler method, and get the image contour by repeated iterative. In order to meet the stability and convergence requirements, whether adopt explicit or implicit iterative method, the step size must be small enough. After each updated, it need to re-initialize the function (SDF) in order to maintain the stability of the calculation. Because of large calculation and the slower evolution speed, there are many restrictions in their actual application. Literature [7] improves the partial differential equations' solve of Euler-Lagrange, and constructs a rapid SDF structure.

In literature [8], a curve and layered evolution model and multi-level set equations are given. Jingfeng Han [9] gives a border matching algorithm combining contour extraction and registration. Using MS model and its automatic coupling function to express border characteristics, curve evolution is divided into two parts, linear and nonlinear, and by computer by the finite element method and the gradient descent iterative respectively. Using high-frequency filter component as the coupling function of curves evolution, Xaiojun Du improved the MS model [10], and speed up the solution of evolution function by single-variable PDE and high-frequency deconvolution filtering.

Ant algorithms have generated significant research interest within the search/ optimisation community in recent years. Ant algorithm has been successfully used to solve many NT' problems, such as TSP, assignment problem, job-shop scheduling and graph coloring. The algorithm has inherent parallelism, and we can validate its scalability. A colony of ants begins with no solutions. Each ant constructs a solution by making decisions stochastically, using existing problem constraints and heuristics combined with experience (which is analogous to a substance called pheromone). The colony then reinforces decisions in the construction process according to their successes by adding pheromone, which also decays to mitigate against poorer decisions[11][12].

In this paper, MR images' boundary is extracted by ant colony algorithm and curve evolution method. It can lower the jitter of evolution curve caused by fuzzy edge and uneven gray.

2 Ant Colony Algorithm

Ant algorithm is a method to solve combinatorial optimization problems by using principles of communicative behavior occurring in ant colonies. Ants can communicate information about the paths they found to food sources by marking these paths with pheromone. The pheromone trails can lead other ants to the food sources. Ant algorithm is an evolutionary approach where several generations of artificial ants search for good solutions. Every ant of a generation builds up a solution step by step thereby going through several decisions until a solution is found. Ants that found a good solution mark their paths through the decision space by putting some amount of pheromone on the edges of the path. The following ants are attracted by the pheromone so that they will search in the solution space near good solutions.

Let m be the number of ants; η_{ij} be the visibility of edge (i, j); τ_{ij} be trail degree of edge (i, j); $\Delta\tau_{ij}^{k}$ be pheromone of per length on edge (i, j) leaved by ant k ; P_{ij}^{k} be transition probability of ant k ; α be relative importance of trail

($\alpha \geq 0$); β be relative importance of visibility($\beta \geq 0$); ρ be the permanence of trail($0 \leq \rho < 1$), and $1-\rho$ be the attenuation degree of trail.

State transition rule the probability of selecting j as next point to visit, taking stochastic proportion, when ant k at point i. The probability can be calculated by the formula as follows.

$$P_{ij}^{k} = \begin{cases} 1, & q < q_0, j = \max_{r\in A_i}\{\tau_{ir}{}^{\alpha}\eta_{ir}^{\beta}\} \\ 0, & q < q_0, j \neq \max_{r\in A_i}\{\tau_{ir}{}^{\alpha}\eta_{ir}^{\beta}\} \\ \dfrac{\tau_{ij}{}^{\alpha}\eta_{ij}^{\beta}}{\sum_{r\in A_k}\tau_{ir}{}^{\alpha}\eta_{ir}^{\beta}}, & q \geq q_0 \end{cases} \tag{1}$$

where q is a random number($0 < q < 1$), and q_0 is a given valve value, A_k is the reachable point set of ant k at point i.

Trail updating may be done after each successive move of the ant (termed local updating) or after all ants have completed one cycle (global updating). In local updating, pheromone values are updated on edge (i, j) every time an ant moves from point i to point j. The new quantity of pheromone deposited on an edge is inversely proportional to the edge length; thus, over time, shorter edges will receive more pheromone, which leads to a positive feedback loop of increased use and further reinforcement. During global updating, only one ant is allowed to update the trail values. This elitist strategy requires that only the ant with the iteration-best tour be allowed to deposit additional pheromone. Similar to local updating, the new quantity of pheromone deposited is inversely proportional to a certain value; this time, the value in question is tour length, not edge length. Thus, edges which constitute shorter tours are reinforced more which leads to another positive feedback loop of more use and greater reinforcement. Trail updating rule can be expressed as follows.

$$\tau_{ij} = (1-\alpha)\tau_{ij} + \alpha\Delta\tau_{ij} \tag{2}$$

3 Curve Evolution

In [6] T.F Chan and L.A. Vese proposed an active contour model using an energy minimization technique. Assume that the image u_0 is formed by two regions of approximately piecewise constant intensities u_0^i and u_0^o, and the object to be detected is represented by the region with value u_0^i. If the boundary is given by c_0 ,then $u_0 \approx u_0^i$ inside c_0 and $u_0 \approx u_0^o$ outside c_0.

The energy F defined by:

$$F = F_1(C) + F_2(C) = \int_{inside(C)} |u_0 - c_1|^2 dx + \int_{outside(C)} |u_0 - c_2|^2 dx \tag{3}$$

where c is any variable curve, c_1 and c_2 are constants depending on c .Therefore, The energy F is minimized when $c = c_0$:

$$F = \inf\{F_1(C) + F_2(C)\} \approx F_1(C_0) + F_2(C_0) \approx 0 \tag{4}$$

C is represented by the zero level set of the function $\phi : R^N \to R$:

$$\begin{cases} C = \{x \in R^N : \phi(x) = 0\} \\ C_{inside} = \{x \in R^N : \phi(x) > 0\} \\ C_{outside} = \{x \in R^N : \phi(x) < 0\} \end{cases} \tag{5}$$

Using the standard definition for the Heaviside function H .

$$H(z) = \begin{cases} 1 \text{ if } z \geq 0 \\ 0 \text{ if } z < 0 \end{cases} \tag{6}$$

Area inside C= $\int_\Omega H(\phi)dx$, thus

$$F(\phi, c_1, c_2) = \int_\Omega |u_0 - c_1|^2 H(\phi)dx + \int_\Omega |u_0 - c_1|^2 \{1 - H(\phi)\}dx \tag{7}$$

Minimizing the energy function with respect to C_1 and C_2 gives

$$C_1(\phi) = \frac{\int_\Omega u_0 H(\phi)dx}{\int_\Omega H(\phi)dx} \tag{8}$$

$$C_2(\phi) = \frac{\int_\Omega u_0 \{1 - H(\phi)\} dx}{\int_\Omega \{1 - H(\phi)\} dx} \tag{9}$$

In the classical explicit snake model [1] the parametric curve is embedded into an energy minimization framework. However, the parametrization of the curve causes difficulties with respect to topological changes and numerical implementations. Thus, to prevent these difficulties, implicit active contour models have been developed. The basic ides is to represent the initial curve implicitly within a higher dimensional function, and to evolve this function under a partial differential equation.

Our model is the minimization of an energy based on segmentation. In our implicit scheme the energy minimization parameter is embedded into the diffusion equation. The evolving contour model is given by the following evolution equation:

$$\frac{\partial u}{\partial t} = g(|\nabla G_\sigma * u|^2)|\nabla u| div\left(\frac{\nabla u}{|\nabla u|}\right) + \alpha |\nabla u| F \tag{10}$$

In order to reduce smoothing at edges, the diffusivity g is chosen as a decreasing function of the edge detector $|\nabla G_\sigma * u|$. Here, $\nabla G_\sigma * u$ is the gradient of a smoothed version of u which is obtained by convolving u with a Gaussian of standard deviation σ.

4 Boundary Extraction Model

During boundary extraction, the traditional curve evolution model is easy to make a curve points converge on individual noise points, so noise has a significant influence to evolution model. In this paper, Ant colony algorithm based active contour model is used to quick search and optimize the control points, and it can make quickly converge to the edge of the image. And then using interpolation algorithm, we get the rather rough outline of goals. In next step, taking the outline as the initial object contour of zero level set curve.

The mean-shift algorithm is a nonparametric statistical method for seeking the nearest mode of a point sample distribution. Mean shift is a simple iterative procedure that shifts each data point to the average of data points in its neighborhood. Given a set of data point x_i, the mean shift vector in a simple one-dimensional case can be expressed as[13]:

$$m(x) = \frac{\sum_{i=1}^{n} x_i g((x - x_i / h)^2)}{\sum_{i=1}^{n} g((x - x_i)^2)} - x \tag{11}$$

where x is an arbitrary point in the data space (it can even be one of the x_i points), h is a positive value called the analysis bandwidth and $g(u)$ is a special function with bounded support; $g(u)$ is defined as the first derivative of another bounded-support function.

When $m(x)$ is applied to the original point, x, it results in anew position, x^1; this process can be repeated and an iterative procedure defined in this way:

$$x^{l+1} = m(x^l) + x^l \tag{12}$$

Here is the process of boundary extraction method by ant colony algorithm based active contour model:

(1) Given the initial position of ant swarm, $x_1, x_2, \ldots, x_i$;

(2) Set every parameter of ant algorithms;

(3) In general, guide function is taken as $\eta = 1/d_{ij}$. In our method, η is defined as: $\eta = \dfrac{r}{d_{ij} + |\sum_{i=1}^{k} h_i / k - \sum_{j=1}^{k} h_j / k|}$. r is clustering radius;

(4) P_{ij}^k According to equation (1), calculation the probability P_{ij}^k that integrated x_j into x_i. If $P_{ij}^k \geq q_0$, x_j class will be integrated into x_i class;

(5) Adjust the amount of permanence on the path;

(6) Recalculate the initial position x_i'. $x_i' = \frac{1}{N}\sum_{k=1}^{N} x_k \;\; x_k \in class$ of x_i;

(7) When iterative finished, the algorithm is end, and the points of initial zero level set curve is x_i';

(8) Using interpolation algorithm to get the object's rough contour;

(9) Using curve evolution method to get the accurate contour.

5 Experiments

All algorithms were coded in Microsoft Visual C++ version 6.0 and all experiments were run on a PC Pentium IV 1.8GHz with 256MB RAM running under Microsoft

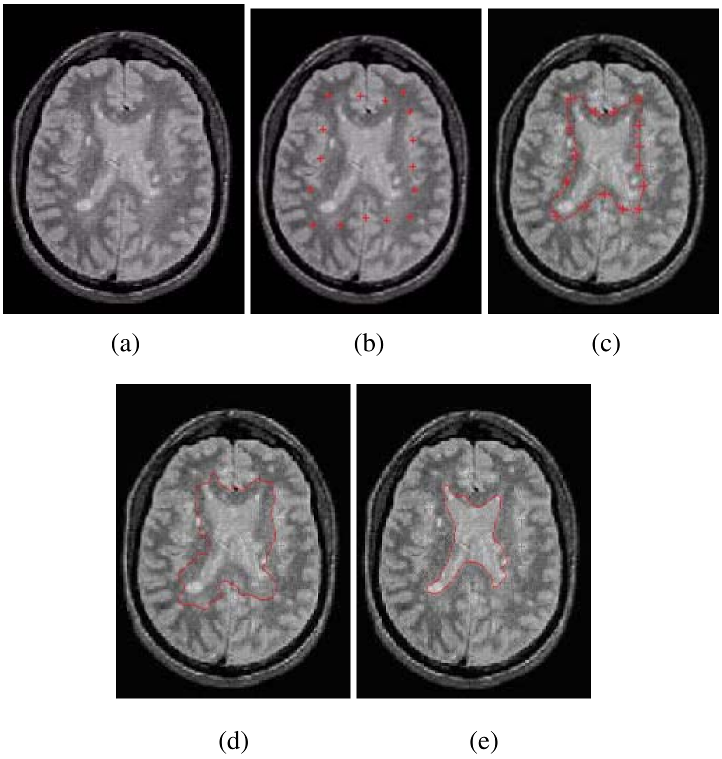


Fig. 1. (a) Original image (b) initial position (c) initial evolution curve (d) traditional curve evolution method (e) Ours

Windows 2000. The parameters of ant colony algorithm are: $\alpha = 3, \beta = 1, \rho = 0.3$. In order to validate the actual results, in this paper, we select brain MR image. MR images are well classified into grey matter, white matter, cerebrospinal fluid, scalpbone, and background. Here, we present numerical results using our model. Figure 1(a) is the original image. Figure 1(b) is the initial position of ant swarm. Figure 1(c) is the initial evolution curve. Figure 1(d) is extraction result by traditional curve evolution method; with the initial curve that interpolation by figure 1(b), which takes 50th iteration. Figure 1(e) is used ours method, which takes 50th iteration. According to the results, our proposed model can be effectively extracted the border of brain tissue, outlining the objectives boundary for the fuzzy part and the gradual change.

6 Conclusion

According to the curve evolution theory and ant colony algorithm, a new boundary contour extraction algorithm is proposed. There are many problem in MR images, such as the border region fuzzy, gray uneven, difficult positing the border. Using ant colony algorithm, the initial evolution is get, and then the boundary is extraction by curve evolution. The approach automatically detect smooth boundaries, and change of topology. Experiments show that ours method can effectively lays out the fuzzy image and discontinuous marginal.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (NSFC) under Grant No. 60773053 and 60970105; Natural Science Foundation of Shandong Provincial under Grant No. Y2007G22 and ZR2009GQ005; Science and technology projects of Shandong Province Education Department No. J08LJ06.

References

1. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: active contour models. International Journal of Computer Vision 1(4), 321–331 (1987)
2. Caselles, V., Catte, F., Coll, T., Dibos, F.: A geometric model for active contours in image processing. Num. Mathematik. 66, 1–31 (1993)
3. Caselles, V., Kimmel, R., Sapiro, G.: Geodesic Active Contours. International of Journal of Computer Vision 22(1), 61–79 (1997)
4. Osher, S., Sethian, J.: A.Fronts propagation with curvature dependent Speed:Algorithm based on Hamilton-Jacobi formulations. Journal of Computational Physics 79(1), 12–49 (1998)
5. Mumford, D., Shah, J.: Optimal approximation by piecewise smooth functions and associated variational problems. Communications on Pure and Applied Mathematics 42(5), 577–685 (1989)
6. Chan, T. F., Vese, L.A.: Active contours without edges. IEEE Transactions on Image Processing 10(2), 266–277 (2001)

7. Li, J., Yang, X., Shi, P.-F.: A Fast Level Set Approach to Image Segmentation Based on Mumford-Shah Model. Chinese Journal of Computers 25(11), 1175–1183 (2002)
8. Gao, S., Bui, T.D.: Image segmentation and selective smoothing by using Mumford-Shah model. IEEE Transactions on Image Processing 14(10), 1537–1549 (2005)
9. Han, J., Berkels, B., Droske, M., et al.: Mumford–Shah Model for One-to-One Edge Matching. IEEE Transactions on Image Processing 11(16), 2720–2732 (2007)
10. Du, X., Bui, T.D.: A New Model for Image Segmentation. IEEE Signal Processing Letters 15, 182–185 (2008)
11. Wein Christopher, J., Blake Jan, F.: On the performance of Fractal Compression with Clustering. IEEE Trans. Image Process. 5(3), 522–526 (1996)
12. Dorigo, M., Gambardella, L.M.: Ant colony system: a cooperative learning approach to the traveling saleman problem. IEEE Trans. on Evolutionary Computation 1(1), 53–56 (1997)
13. Cheng, Y.: Mean Shift, Mode Seeking, and Clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence 17(8), 790–799 (1995)

ACO Based Energy-Balance Routing Algorithm for WSNs

Xuepeng Jiang[1] and Bei Hong[2]

[1] Department of Strategic Missile Engineering, Naval Aeronautical and Astronautical University, 264001 Yantai, Shandong, China
[2] Xi'an Institute of Hi-Tech, HongQing Town, Xi'an, China
{Xue-pengJIANG,BeiHONG,cati2k}@yahoo.com.cn

Abstract. Ant Colony Optimization (ACO) is a heuristic bionic evolutive algorithm. In ACO algorithm, every ant has simple function, works with simple principle, which suits the characteristic of Wireless Sensor Networks (WSNs) and the request of its routing design. An ACO based Energy-Balance Routing Algorithm(ABEBR) was presented to balance the energy consumption in WSNs. Furthermore, a new pheromone update operator was designed to integrate energy consumption and hops into routing choice. This paper compares ABEBR with some classic routing algorithms (LEACH, DD and Flooding). Simulation results show that the presented algorithm can avoid energy working out too early on the less hops path, obviously balance the energy consumption and prolong the lifetime of WSNs.

Keywords: Ant Colony Optimization, WSNs routing algorithm, Energy Balance, Pheromone.

1 Introduction

Due to the fast development of the microprocessor, sensor and transceiver, there is great applications foreground abort WSNs. IN WSNs, once the sensors were disposed to the area of sense, every sensor only depends on the cell to provide energy. The limit energy is the key issue influencing WSNs performance. So, how to use the limit energy of WSNs to maximize the life of WSNs becomes the all-important problem of routing design.

In order to optimize of the routing quality and the energy consumption, it is need to achieve the tradeoff between route hops and the energy consumption. ZWang[1] proved that it is a NP-complete problem when there are up two more routing measurements in routing design. It is hard for traditional algorithm to solve NP-complete problem. The Ant Colony Optimization is a heuristic bionic evolutive algorithm, which has good positive feedback and parallel computing. So, it shows good performance on solving NP-complete problem. In this paper, we present an ACO based Energy-Balance Routing Algorithm (ABEBR), which optimizes the consumption of WSNs and prolongs the lifetime of nets obviously.

The simulation results show that this algorithm has better energy consumption radio and longer lifetime compared with some classic routing algorithms (such as DD, LEACH).

This paper is organized as follows. In Section II, we introduce recent routing researches about ACO. In Section III, we briefly recall the relevant foundation of ACO. In Section IV, we present an ACO based Energy-Balance Routing Algorithm(ABEBR)to balance the energy consumption in WSNs; And a new pheromone update operator is designed to integrate energy consumption and hops into routing choice. In Section V, we compare the presented method with other methods, such as LEACH, DD, Flooding by numerical examples. We conclude in Section VI.

2 Related Work

ACO routing algorithms are first designed for wired networks as in AntNet [4], ABC (Ant Based Control) [5]. The main concept of both AntNet and ABC is to deploy small ant packets to discover forwarding paths from source to destination. These algorithms exhibit a number of interesting properties such as working in a fully distributed way, being highly adaptive to network and traffic changes, and automatically taking care of data load spreading. But using periodic unicast ants to discover routers would incur a large delay and the adaptability to topology changes would be unacceptably slow. There are some ACO routing schemes for Mobile Ad-hoc Networks (MANETs) such as GPS/Ant-like routing algorithm for mobile ad-hoc networks [6], Ant-AODV [7], and ARA (Ant-Colony-Based Routing Algorithm) [8]. However, none of these algorithms is satisfied for special characteristics of WSNs.

3 Basic Principle of Ant Colony Optimization

The Ant Colony Optimization was presented by M. Dorig during the early 1990s. It's a heuristic bionic evolutive algorithm which simulates the behavior of the ants' searching road in nature. This algorithm has been applied successfully with many combinatorial optimization problems. Compared with classic intelligent algorithms (such as GA, GP, ES), the ant colony optimization has such advantages: positive feedback which can induct good solution quickly; parallel computing that can prevent the early convergence; strong method of elicitation which can find suitable solution during the early stage of optimization.

First of all, the main procedures problem solving by basic ant colony algorithm[2] is briefly described in the following.

According to the characteristics of the problem to be solved, the solution space of the problem is transformed into a searching graph $G=(V,E)$, where $V=\{v_1,v_2,...,v_m\}$ is the set of nodes in the graph G, and $E=\{e_1,e_2,...,e_r\}$ is the set of oriented arcs among those nodes. A path, connecting an initiative node and a terminal node through a series of interim nodes by oriented arcs, is denoted by ω which corresponds to a feasible solution candidate. A colony of n ants are denoted by $A=\{a_1,a_2,...,a_n\}$. In each searching period, an ant a_k chooses a path ω_k in the graph G randomly according to a predetermined path selection possibility. A searching period ends up in the algorithm when all the n ants finish the path seeking respectively. The path selection possibility $p_{i,j}(t)$ for a searching from v_i to v_j the searching period t is defined by

$$p_{ij}^k(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}(t)]^{\beta}}{\sum\limits_{s\notin tabu_k}[\tau_{is}(t)]^{\alpha}[\eta_{is}(t)]^{\beta}}, & j\notin tabu_k \\ 0, & j\in tabu_k\end{cases} . \tag{1}$$

where $tabu_k$ is the partial path that has been searched by ants in the period t; $\tau_{i,j}(t)$ is the density of pheromone accumulated on the path segment (v_i,v_j) by ants in the period t; $\eta_{i,j}(t)$ is the information of searching for that path segment, and $\eta_{i,j}(t)$ is defined by $\eta_{i,j}(t)=1/d_{i,j}$, $d_{i,j}$ is the distance of the arc from v_i to v_j; and $\alpha,\beta>0$ are called the pheromone index and cost index, respectively. The pheromone $\tau_{i,j}(t)$ will be update at the end of each searching period in the way of

$$\tau_{ij}(t+n)=(1-\rho)\tau_{ij}(t)+\rho\Delta\tau_{ij}(t) . \tag{2}$$

Where $\rho\in(0,1)$ is called the evaporation factor; $\Delta\tau_{i,j}$ is a pheromone increment as a node increasing function of objective values, e.g. ,

$$\Delta\tau_{ij}(t)=\sum_{k=1}^{m}\Delta\tau_{ij}^k(t) . \tag{3}$$

In the model of Ant-Cycle, $\Delta\tau_{ij}^k(t)$ is defined by

$$\Delta\tau_{ij}^k(t)=\begin{cases}\dfrac{Q}{L_k} & if \quad a_k \quad chooses \quad (v_i,v_j) \\ 0 & \qquad otherwise\end{cases} . \tag{4}$$

where Q is the density of pheromone, and Q influences the computing time in some degree; L_k is the accumulated length by ant a_k in the period t.

4 ACO Based Energy-Balance Routing Algorithm

When designing the routing algorithm in wired networks, we only need to search the least hops path in networks. However, in WSNs, where the energy is limit, there is special request on designing routing algorithm. Low energy consumption and long lifetime are the most basic conditions for good performance of WSNs. Therefore, it is very necessary to put the factor of energy consumption into routing design. We design the ACO based Energy-Balance Routing Algorithm, through which, the energy consumption and hops are integrated into routing choice. Through this algorithm, the energy consumption in WSNs is balanced distinctly.

4.1 WSNs Model for Algorithm

Our algorithm is suitable to the model as follows:

(1) The bi-direction link is used in networks. That is to say, if sensor A can communicate with sensor B, then sensor B can communicate with sensor A.
(2) The initial energy in sensors is equal. During the early period of networks life, there is enough energy in sensor to communicate with border sensors.
(3) The sensor has all-direction antenna.

4.2 ACO Based Energy-Balance Routing Algorithm Description

In ABEBR, two kinds of ant are inducted to optimize the routing, that is forward ant and backward ant. The forward ant is launched by the source in order to find multiple paths to the sink. And the backward ant returns to the source to set up the paths. According to the common practice in ACO algorithms, the paths are set up in the form of pheromone tables indicating their respective quality.

At the route setup phase, source node doesn't have routing information for sink available, it broadcasts a forward ant. If the forward ant arrives to the sink, a path between source and sink is set up. The routing information of source is represented in its pheromone table. The neighbors of source, which are on the path from source to sink, are recorded in source node's pheromone table. The entry $\tau_k(i,j)$ of pheromone table is the pheromone value indicating the estimated goodness of forward ant k going from i over neighbor j to reach sink. Through this strategy, source node obtains multiple paths to the sink. Then the source nodes launch forward ants respectively at each iteration. The forward ant chooses node j in pheromone table N_i as next hop stochastically with the transfer probability $P_k(i,j)$ which is described as follows:

$$P_k(i,j)=\begin{cases}\dfrac{[\tau_k(i,j)]^{\alpha}[\eta(j)]^{\beta}}{\sum\limits_{\substack{s\in N_i\\ s\notin tabu_k}}[\tau_k(i,s)]^{\alpha}[\eta(s)]^{\beta}}, & j\notin tabu_k\\ 0, & j\in tabu_k\end{cases} . \qquad (5)$$

In this formula, α is pheromone heuristic factor which is used to figure the weight of pheromone; β is expectation heuristic factor that is used to figure the weight of heuristic information; $\eta(j)$ is heuristic information which can be evaluated by formula(6).

$$\eta(j) = \frac{1}{c - e_{residual}(j)} . \tag{6}$$

$e_{residual}(j)$ is the residual energy of node j; c is the initial energy of nodes which is a constant. $tabu_k$ is a list of forward ant k which records the node ID traversed by forward ant.

After one iteration, ant will update the pheromone on formed path. Every ant records the average energy level of the sensor ($eavg_k$) and the minimum energy level ($emin_k$) on its path. $eavg_k$ and $emin_k$ are figured out by sensors on path and record in ant at last. At the end of searching path, ant arrives at the aim node (sink), and then the sink figures out the pheromone of every path which is used to update through the data in ant. The more details about pheromone update as follows:

$$\tau_k(i,j) = (1-\rho)\tau_k(i,j) + \Delta\tau_k . \tag{7}$$

$\Delta\tau_k$ in formula (7) is the key to pheromone update in our algorithm. In order to balance the energy consumption, the update of $\Delta\tau_k$ is figured considering both the hops of path and energy level. A few hops path with low energy level may lose its function quickly, consequently reducing the WSNs performance. In the same, a high energy level path with many hops may increase the energy consumption and shorten the lifetime of WSNs. In our model, $eavg_k$ and $emin_k$ are used to compare the energy level among paths and hop_k describes the path's length. We bring these three factors into the evaluation of $\Delta\tau_k$. The details are as follows:

$$\Delta\tau_k = Q\left(\frac{\varphi}{hop_k} + \frac{\gamma}{e_{avgk}} + \frac{\mu}{e_{\min k}}\right) . \tag{8}$$

φ , γ , μ are weight coefficients which are set according to the importance of hops and energy level in WSNs.

4.3 Algorithm Flow Chart

The algorithm is started to optimize when sensors are disposed into monitor area and self-organize to be networks. Considering the characteristic of dynamic about WSNs, the algorithm should be started at regular intervals to adapt to the topology change of WSNs. After N_{max} iterations, optimized paths between source node and sink are set up. The flow chart is described in Fig. 1.:

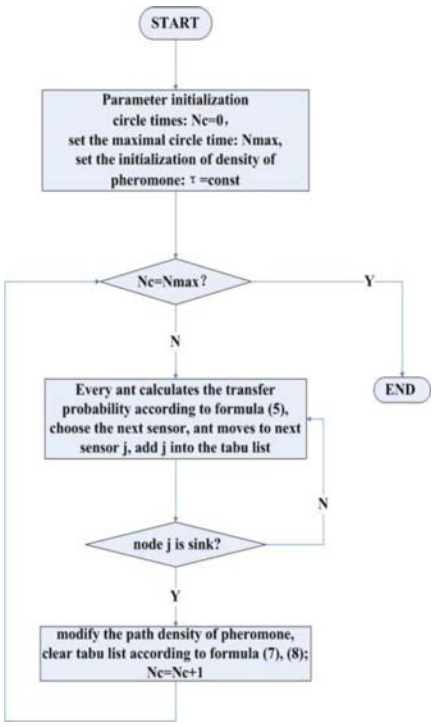

START
Parameter initialization
circle times: Nc=0,
set the maximal circle time: Nmax,
set the initialization of density of
pheromone: τ =const
Nc=Nmax?
Y
N
END
Every ant calculates the transfer
probability according to formula (5),
choose the next sensor, ant moves to next
sensor j, add j into the tabu list
N
node j is sink?
Y
modify the path density of pheromone,
clear tabu list according to formula (7), (8);
Nc=Nc+1

Fig. 1. Flow Chart

5 Simulations and Analysis

To test performance of algorithm, this paper simulates and compares ABEBR with LEACH, DD, Flooding based on NS2. On the assumption that, m sensor nodes are randomly laid in the monitor area of the simulation model and the value of m is among 50 to 250. The initialization energy of every sensor node is the same, but the initialization energy of sink node is higher than normal nodes. The simulation compares ABEBR with LEACH, DD, Flooding according to two energy efficiency variables: energy consumed rate and lifetime of network. The energy consumed rate of network means that the average consumed energy of whole network when sink node receives a data package. It can well estimate the influence to consumed energy of whole network by different algorithm. This paper compares the energy consumed rate of 4 algorithms in different network scales, the details are shown in Fig. 2.

As shown in Fig.2, the energy consumed rate is higher as the scale of network is bigger. The energy consumed rate of Flooding is distinctly higher than others, because there are a lot of broadcast transmissions. The line of LEACH is flat, that is the result of cluster. The ABEBR algorithm can find optimization fast. It gives the optimal path from every node to sink in a short time. So it reduces the energy consumed rate than other algorithms.

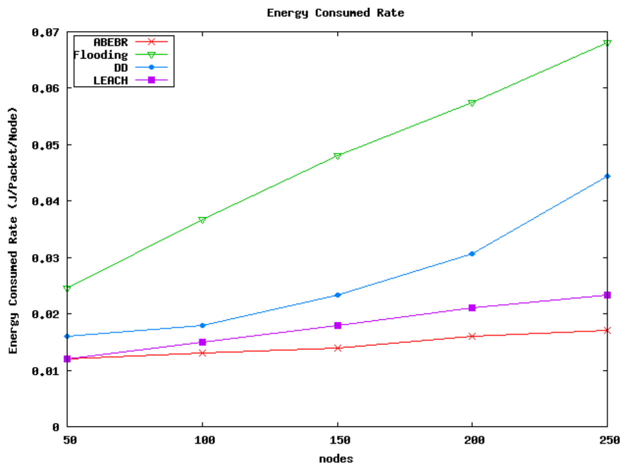


Fig. 2. Energy Consumed Rate in Different Network Scales

Fig. 3. shows the influence to lifetime of whole network by different algorithm in different scales. This paper defines the time that sink can't receive data package from source nodes as lifetime of network. When sink can't receive any data package, the network can't work. That is to say, the network can't transmit any data. The trend of natural life is shown in Figure 3. The trend of ABEBR is flatter. So the algorithm ABEBR can be applied in different scale networks.

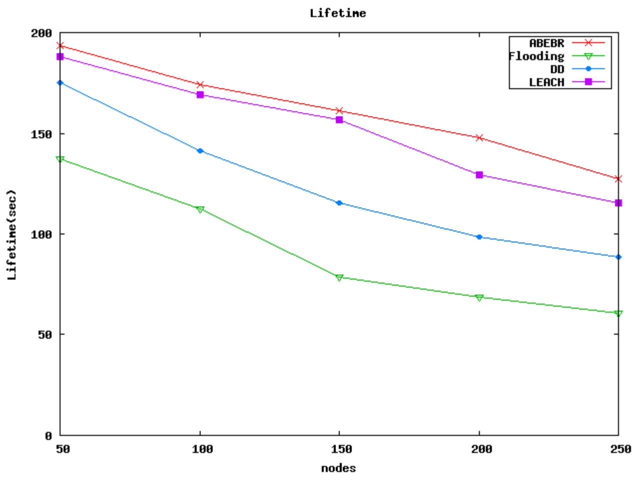


Fig. 3. Lifetime in Different Network Scales

6 Conclusion

In this paper, we present a new WSNs routing algorithm (ABEBR) which is based on modified ant colony optimization. In this algorithm, a new pheromone update operator is designed, through which the energy state of routings are considered during routing choosing. As the simulations results shown, the ABEBR algorithm has good performance on energy balance and prolongs the lifetime distinctly. So, the ABEBR provides a good method for balancing energy consumption during routing choosing, which can keep the connectivity of WSNs considerably.

References

1. Wang, Z., Crowcroft, J.: Quality of Service Routing for Supporting Multimedia Applications. IEEE Journal on Selected Areas in Communications 14(7), 1228–1234 (1996)
2. Duan, H.B.: Ant Colony Algorithm: Theory and Applications, pp. 24–26 (2005)
3. Zhang, Y., Kuhn, L., Fromherz, M.: Improvements on Ant Routing for Sensor Networks. In: Dorigo, M., Birattari, M., Blum, C., Gambardella, L.M., Mondada, F., Stützle, T. (eds.) ANTS 2004. LNCS, vol. 3172, pp. 154–165. Springer, Heidelberg (2004)
4. Singh, G., Das, S., Gosavi, S., Pujar, S.: Ant Colony Algorithms for Steiner Trees: An Application to Routing in Sensor Networks. In: Recent Developments in Biologically Inspired Computing, pp. 181–206. Idea Group Publishing (2004)
5. Alonso, J., Dunkels, A., Voigt, T.: Bounds on the energy consumption of routings in wireless sensor nodes. In: WiOpt 2004: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, Cambridge, UK (2004)
6. Al-Karaki, J.N., Kamal, A.E.: Routing Techniques in Wireless Sensor Networks: A Survey. In: Wireless Communications, pp. 6–28. IEEE, Los Alamitos (2004)
7. Madiraju, S., Mallanda, C.: EBRP: Energy Band based Routing Protocol for Wireless Sensor Networks. In: Intelligent Sensors, Sensor Networks and Information Processing Conference (2004)
8. Min, R., Bhardwaj, M., Cho, S.-H., Shih, E., Sinha, A., Wang, A., Chandrakasan, A.: Low-Power Wireless Sensor Networks. In: The 14th International Conference on VLSI Design, p. 205 (2001)

Swarm Intelligence Algorithms for Portfolio Optimization[*]

Hanhong Zhu[1,2], Yun Chen[1], and Kesheng Wang[2]

[1] School of Public Economics & Administration, Shanghai University of Finance and Economics (SUFE), 200433 Shanghai, China
zhh@mail.shufe.edu.cn, yun.chen@shufe.edu.cn
[2] Department of Production and Quality Engineering,
Norwegain University of Science and Technology (NTNU)
kesheng.wang@ntnu.no

Abstract. Swarm Intelligence (SI) is a relatively new technology that takes its inspiration from the behavior of social insects and flocking animals. In this paper, we focus on two main SI algorithms: Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO). An extension of ACO algorithm and a PSO algorithm has been implemented to solve the portfolio optimization problem, which is a continuous multi-objective optimization problem.. The portfolio optimization model considered in this paper is based on the classical Markowitz mean-variance theory. The results show ACO performs better than PSO in the case of small-scale and large-scale portfolio, but in the case of medium-scale portfolio, PSO performs a better than ACO technique.

Keywords: Swarm Intelligence (SI), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Portfolio Optimization (PO), Sharpe Ratio (SR).

1 Introduction

Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO) are two major techniques within Swarm Intelligence (SI) [1, 2]. There are much studies on ACO and PSO in the literatures, but none of them deals with comparison of ACO and PSO in their performance of portfolio optimization. This paper presents an empirical study to compare the performance of both algorithms to the portfolio optimization.

Portfolio Optimization (PO) is one of the most studied topics in finance field. The major problem in PO is concerned with managing the portfolio of assets that minimizes the risk objectives subjected to the constraint for guaranteeing a given level of returns [3]. In this paper, we deal with the so-called Mean-Variance portfolio selection, which is formulated in the similar way done by Markowitz [4]. His theory has revolutionized the common understanding about portfolio of assets, and has gained widespread acceptance as a practical tool for portfolio optimization. But in some cases, researchers face other problems, such as its size, real-world requirements [5], very limited computation time, and limited precision in estimating instance parameters, which will make analytical

[*] This research is supported by the Shanghai key scientific and technological project (Grant No. 08DZ1120500).

methods not particularly suitable for tackling large instances of the constrained Mean-Variance Model. Therefore researchers and practitioners have to resort to other enhancements such as SI optimization techniques. In the last decade or so, there is a growing trend to apply swarm intelligence to portfolio optimization. There are some reports of solving the portfolio optimization problem using swarm intelligence optimization techniques. Ahmed and Sajjad [6] have compared AIS with PSO for constrained portfolio optimization. Kendall and Su [7] compared PSO with Excel Solver. Their results showed that PSO performed better in constrained portfolio optimization. The results of this study are compared between ACO and PSO.

We arrange the rest part of the paper as the following, in section 2, a general principle of portfolio optimization problems are described. An extension of ACO algorithm and a PSO algorithm are presented in sections 3. The results and discussions are mentioned in section 4. The conclusion is drawn in Section 5.

2 The Portfolio Optimization Problem

Portfolio management is a trade-off between the return and risk of an investment. We want to maximize the return while minimizing the risk. We will use the Mean-Variance portfolio selection model [4]. The Mean-Variance model is described as:

$$\text{Min}\sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_{ij} \tag{1}$$

$$\text{Subject to} \sum_{i=1}^{N} w_i r_i = \text{R}^*, \tag{2}$$

$$\sum_{i=1}^{N} w_i = 1, \tag{3}$$

$$0 \le w_i \le 1 \quad i = 1, \cdots, N. \tag{4}$$

where N is the number of different assets, σ_{ij} is the covariance between returns of assets i and j, w_i is the weight of each stock in the portfolio, r_i is the mean return of stock i and R^* is the desired mean return of the portfolio. We also use the second method to model portfolio optimization problem as the following:

2.1 Efficient Frontier

We can find the different objective function values by varying desired mean return R^*, so a new named risk aversion parameter $\lambda \in [0,1]$ has been introduced, the sensitivity of the investor to the risk increase as λ increasing from zero to unity. With the λ, the model can be described as:

$$\text{Min}\lambda\left[\sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_{ij}\right] - (1-\lambda)\left[\sum_{i=1}^{N} w_i r_i\right] \tag{5}$$

$$\text{Subject to}\sum_{i=1}^{N} w_i = 1, \tag{6}$$

$$0 \le w_i \le 1 \quad i = 1, \cdots, N. \tag{7}$$

In the model included parameter λ, we can draw a continuous curve that is called an efficient frontier according the Markowitz theory[4], the curve composed of mean return and variance according different λ, and every point on an efficient frontier curve indicates an optimum.

2.2 Sharpe Ratio Model

Instead of focusing on the mean variance efficient frontier, we seek to optimize the portfolio Sharpe Ratio (SR) [8]. The Sharpe ratio is quite simple and it is a risk-adjusted measure of return that is often used to evaluate the performance of a portfolio. It is described as the following equation:

$$SR = \frac{R_p - R_f}{StdDev(p)} \tag{8}$$

Where p is the portfolio, R_p is the mean return of the portfolio p, R_f is the test available rate of return of a risk-free security. *StdDev(p)* is the standard deviation of R_p. Adjusting the portfolio weights w_i, we can maximize the portfolio SR in effect balancing the trade-off between maximizing the expected return and at the same time minimizing the risk.

3 Swarm Intelligence for Portfolio Optimization

Swarm Intelligence is the emergent collective intelligence of groups of simple agents acting almost independently. Algorithms following this paradigm have many desirable properties: flexibility, decentralized control, robustness and fault tolerance. The list of SI includes ant colony optimization (ACO) and particle swarm optimization (PSO). The optimization algorithms have been used in several financial applications ranging from the prediction of stock markets, foreign exchange rates, and risk management.[9] This paper will study the comparison of the two SI optimization algorithms, namely ACO and PSO, for portfolio optimization. Both concepts are briefly described in the following subsections.

3.1 ACO

The ACO algorithm is based on the foraging behavior of real ants. The idea was exploited for the first time by Dorigo[1]. The ACO algorithm starts with setting up some parameters and initializing the pheromone trails. Then follows the main body of the algorithm, which consists of two important steps: solution construction and pheromone update. This part of the algorithm is repeated until some termination condition is met. Solutions are built incrementally based on the pheromone and possibly based on some knowledge of the problem. In each step of the solution construction the next component *j* from the set of all components that are still possible, *N*, with the following probability when we are in component i:

$$p_{ij} = \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{l \in N} \tau_{ij}^{\alpha}\eta_{ij}^{\beta}} \tag{9}$$

where τ is the pheromone, η is problem specific information and α and β are parameters of the algorithm. α and β are generally chosen $\alpha = 1$ and β between 2 and 5. the pheromone update depends on the specific ACO algorithm you would choose but in general the existing pheromone is decreased and then some extra pheromone is added based on the solutions constructed in that iteration. So it looks like

$$\tau_{ij} = (1-\rho)\tau_{ij} + \Delta\tau_{ij} \tag{10}$$

where $\Delta\tau_{ij}$ is the pheromone added if the path *ij* is part of a solution used for the update. The standard ACO algorithm uses each constructed solution and the amount of pheromone dropped on the solutions components is higher for good solutions.

Unfortunately, we can't use the normal ACO algorithm for the portfolio selection problem since it only works for discrete problems. We will follow the extended ACO algorithm as it was described by Socha[10]. The main problem in the original ACO algorithm is its continuous nature. We can't use a table to store the pheromone, since we would need an infinite amount of values inside the table. These problems are both in the pheromone update and in the construction of solutions. Instead of the pheromone table we will keep the solution in a list. Each solution will have some pheromone value associated with it, so the pheromone update stays the some. The solutions will still be build incrementally but there are some new part to assign a value to each component of a solution. The idea of the algorithm in general is still the same, as you can see in the reference[10].

3.2 PSO

Particle Swarm Optimization (PSO) [2] is a population swarm intelligence optimization technique inspired by the natural behavior of bird flocking ad fish schooling. The algorithm is initialized with a population of potential solutions, called particles, which "fly" through the search space by following the current optimum particles. All particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which directed the flying of the particles. Each particle keeps track of its coordinates in the problem space which are associated with the best solution (fitness) it has achieved so far. In this paper, we set the Sharpe Ratio (equation 8) as the fitness function. This value is called *pbest*. Another "best" value that is tacked by the particle swarm optimizer is the best value, obtained so far by any particle in the neighbors of the particle. This location is called *lbest* when a particle takes all the population as its topological neighbors, the best value is a global best and is called *gbest*. At each time stamp, each particle updates its velocity and position with the following equations.

$$\vec{v}_i(t+1) = w\vec{v}_i(t) + c_1 r_1[\vec{p}_i(t) - \vec{x}_i(t)] + c_2 r_2[\vec{p}_g(t) - \vec{x}_i(t)] \tag{12}$$

where *t* is the iteration sequence of the particle *i*, c_1 and c_2 are positive constant parameters called acceleration coefficients which are responsible for controlling the maximum step size, r_1 and r_2 are random numbers between (0, 1), w is a constant. and $\vec{v}_i(t+1)$ is particle *i*'s velocity at iteration *t* + 1. $\vec{v}_i(t)$ is particle *i*'s velocity at iteration t. $\vec{x}_i(t)$ is particle *i*'s position at iteration *t*. $\vec{p}_i(t)$ is the historical individual best position of the swarm. Finally, the new position of particle *i*, $\vec{x}_i(t+1)$, is calculated as shown in (13). The detail algorithm is as below algorithm1.

$$\vec{x}_i(t+1) = \vec{x}_i(t) + \vec{v}_i(t+1) \tag{13}$$

```
Randomly initialize particles (solutions)
Do until a stopping criterion is met
  For each particle
    Calculate fitness value
    If the fitness value is better than its personal best
      Set current value as the new pBest
  End
  Choose the particle with the best fitness value in the neighbourhood as gBest
  For each particle
    Calculate particle velocity according equation (12)
    Update particle position according equation (13)
 End
```

Algorithm 1. The PSO algorithm

4 Experiment and Discussion

For the experiments, real data from companies listed on the Shanghai stock exchange 50 Index (SSE50 Index) is used. The corresponding rate of return is computed from the daily values during the period from January to April 2009. After computing the average return and variance/covariance matrix, ACO, and PSO algorithms are applied to optimize the expected return (ER), stand variance (StdDev) and Sharpe Ratio (SR) of the different portfolios.

During the experiments, the population size of PSO and ACO is set to 100 while the number of generations is set to 500. Each algorithm is run 10 times (with different starting points) to computed the average and maximums value of the ER, StdDev and SR. The results are shown in the table1.

In Table 1, in terms of average Sharpe Ratio, when the number of the stocks is 5, 15, 20 and 35, ACO and PSO produced the same results.When the number is 8 and 30, ACO produced a little better results, and when the number is 10, 25 and 40, PSO produced a little better results. In terms of Maximum Sharpe Ratio, when the stocks number is 5, 30, 35 and 40, ACO produce a better result than PSO, and when the number is 8 and 20, ACO and PSO produce the same results, and when the number is 10, 15 and 25, PSO produced a little better results.

Table 1. Results by ACO and PSO when Risk-free is 0

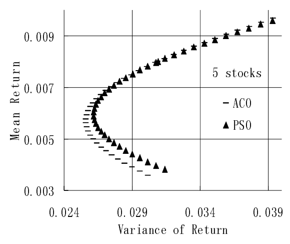
Stocks	**Avg SharpeRatio**		**Max SharpeRatio**	
	ACO	**PSO**	**ACO**	**PSO**
5	25.945%	25.945%	25.947%	25.945%
8	32.132%	32.123%	32.132%	32.132%
10	32.986%	32.991%	32.986%	33.033%
15	32.597%	34.597%	34.597%	34.598%
20	32.692%	34.692%	34.692%	34.692%
25	32.936%	34.942%	34.936%	34.991%
30	34.947%	34.946%	34.948%	34.947%
35	34.947%	34.947%	34.949%	34.947%
40	34.948%	34.949%	34.955%	34.950%

Table 2. Random Results gotten by ACO and PSO when Risk-free is 0 and 0.03%

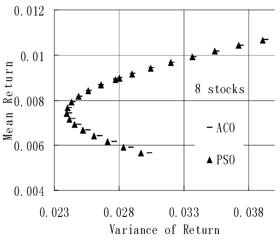
Stocks	R_f	Mean		StdVar		SharptRatio	
		ACO	PSO	ACO	PSO	ACO	PSO
5	0	0.772%	0.775%	2.976%	2.986%	25.945%	25.944%
	0.03%	0.811%	0.812%	3.137%	3.141%	16.296%	16.290%
8	0	0.855%	0.855%	2.611%	2.608%	32.754%	32.769%
	0.03%	0.918%	0.917%	2.905%	2.901%	21.273%	21.271%
10	0	0.828%	0.828%	2.510%	2.511%	32.986%	32.984%
	0.03%	0.883%	0.882%	2.722%	2.721%	21.416%	21.405%
15	0	0.871%	0.874%	2.517%	2.528%	34.597%	34.577%
	0.03%	0.938%	0.938%	2.760%	2.769%	23.107%	23.055%
20	0	0.890%	0.883%	2.565%	2.554%	34.692%	34.567%
	0.03%	1.003%	1.005%	2.971%	2.994%	23.649%	23.562%
25	0	0.930%	0.907%	2.661%	2.610%	34.936%	34.749%
	0.03%	1.079%	1.087%	3.179%	3.222%	24.515%	24.418%
30	0	0.934%	0.920%	2.672%	2.649%	34.947%	34.711%
	0.03%	1.083%	1.079%	3.185%	3.206%	24.577%	24.303%
35	0	0.934%	0.935%	2.672%	2.696%	34.947%	34.679%
	0.03%	1.082%	1.060%	3.183%	3.153%	24.577%	24.091%
40	0	0.934%	0.911%	2.672%	2.652%	34.947%	34.330%
	0.03%	1.083%	1.046%	3.188%	3.245%	24.571%	22.990%

In the Table 2, there are random results generated by PSO and ACO techniques when the riskfree is 0 and 0.3%. Taking the sets of optimal portfolios in the Table 2, we trace out their efficient frontiers in Fig.1. Fig.1(a), (b), (c) ,(d), (e), (f) ,(g) ,(h) , and (i) are the efficient frontier curves for the portfolio of 5, 8, 10, 15, 20, 25, 30, 35, and 40 stocks. When the number of the stocks is few (5 and 8) or very many (30, 35 and 40), the upper part of the efficient frontier by ACO technique is above the efficient frontier by PSO. And when the portfolio is medium-sized (10, 15, 20 and 25), the two efficient frontier curves obtained by ACO and PSO almost overlap.

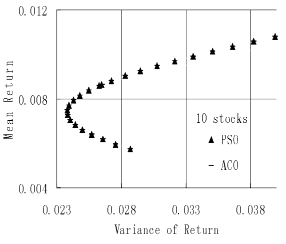
To sum up, we conclude that when the number is very fewer (≤ 10) or the number is very much (≥ 30), ACO performs better than PSO for the portfolio optimization, and when the portfolio is medium-sized, PSO performs a little better than ACO in the portfolio optimization problem.



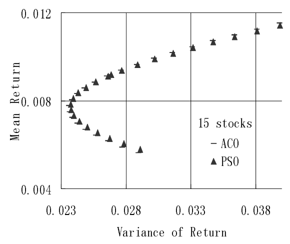
(a) 5 stocks portfolio



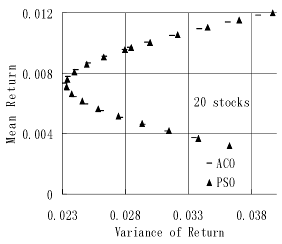
(b) 8 stocks portfolio



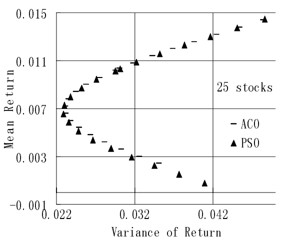
(c) 10 stocks portfolio



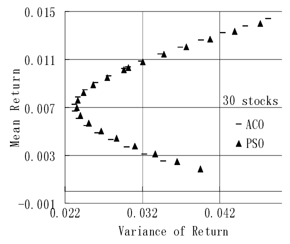
(d) 15 stocks portfolio



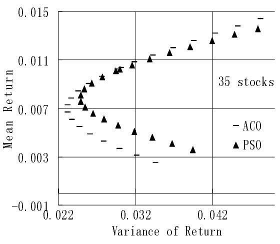
(e) 20 stocks portfolio



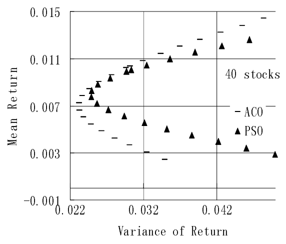
(f) 25 stocks portfolio



(g) 30 stocks portfolio



(h) 35 stocks portfolio



(i) 40 stocks portfolio

Fig. 1. The Efficient frontier of the portfolios gotten from the ACO and PSO algorithms

5 Conclusion

This paper presents an empirical study to compare the performance of two swarm intelligence optimization techniques, named ACO and PSO, for constrained portfolio optimization. They are all drawn from a swarm metaphor, and exhibit flexibility, robustness, self-organization and communication between individual members of the population. In many cases, however, ACO ends up with slightly better results than PSO especially in the case of small-scale and large-scale portfolio. But in the case of medium-scale portfolio, the PSO performs a little better than the ACO techniques. The future work would focus on comparing these and other optimization techniques.

Acknowledgments. The authors gratefully acknowledge the valuable comments from Yi Wang. (Nottingham Business School, NTU, UK.) and Simon Crevals (Norwegian University of Science and Technology, NTNU).

References

1. Dorigo, M.: Optimization, Learning and Natural Algorithms. Politecnico di Milano, Ph.D (1992)
2. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: International Conference on Neural Networks, ICNN 1995, p. 1942 (1995)
3. Black, F., Litterman, R.: Global Portfolio Optimization. Financial Analysts Journal 48, 28–43 (1992)
4. Markowitz, H.: Portfolio Selection. The Journal of Finance 7, 77–91 (1952)
5. Zhou, X.Y., Li, D.: Continuous-Time Mean-Variance Portfolio Selection: A Stochastic LQ Framework. Applied Mathematics and Optimization 42, 19–33 (2000)
6. Ahmed, A., Sajjad, H.: Comparison of AIS and PSO for Constrained Portfolio Optimization. In: International conference on information and financial engineering, Singapore, pp. 50–54 (2009)
7. Kendall, G., Su, Y.: A Particle Swarm Optimization Approach in The Construction of Optimal Risky Portfolios. In: The 23rd IASTED international multi-Conference, Artificial intelligence and applications, Innsbruck, Austria, pp. 14–16 (2005)
8. Sharpe, W.F.: Mutual Fund Performance. The Journal of Business 39, 119–138 (1966)
9. Brabazon, A., O'Neill, M.: Biologically Inspired Algorithms for Financial Modelling. Springer, Heidelberg (2006)
10. Socha, K.: ACO for Continuous and Mixed-Variable Optimization. In: Dorigo, M., Birattari, M., Blum, C., Gambardella, L.M., Mondada, F., Stützle, T. (eds.) ANTS 2004. LNCS, vol. 3172, pp. 25–36. Springer, Heidelberg (2004)

Document Classification with Multi-layered Immune Principle

Chunlin Liang[1], Yindie Hong[1], Yuefeng Chen[1,*], and Lingxi Peng[2,*]

[1] Software School, Guangdong Ocean Univ., Zhanjiang 524025, China
yes2384735@126.com
[2] School of Computer Science, Guangzhou University, Guangzhou, 510006, China
scu.peng@gmail.com

Abstract. Automatic document classification is helpful in both organizing and finding information on huge resources. A novel multi-layered immune based document classification algorithm is presented. First, we represent the definition of the immune cells, antibody, antigen, and discuss the architecture of multi-layered immune system. Second, we evolve the dynamic models of immune response, immune regulation and immune memory, and establish the corresponding equations. Finally, we implement the simulation experiments, and compare the results with those obtained using the best methods for this application. Experiments show that the algorithm has higher classification accuracy than other document classification methods, and the attractive features such as diversity, self-learning, adaptive and robust etc. It provides a novel solution for document classification.

Keywords: document classification, immune, artificial intelligence.

1 Introduction

Artificial immune is a new research focus following the neural networks, fuzzy systems, and evolutionary computation etc. Especially, the artificial immune system has the attractive features such as diversity, adaptive and robust etc. In recent years, these inviting features caused researchers' widespread concern and heating discuss.

In related research, P. Wang & C. Domeniconi [1] presented an algorithm of building semantic kernels for text classification using Wikipedia. The algorithm overcomes the shortages of the word-based vector approach by embedding background knowledge derived from Wikipedia into a semantic kernel, which is then used to enrich the representation of documents. Ifrim G. et al. [2] presented an algorithm for both classification and retrieval of natural language text documents. The algorithm maps every word onto a concept, and adopts the WordNet thesaurus as a background knowledge base to classify and search documents. Wang Z. & Qian X. [3] presented a novel document classification algorithm based on kernel discriminant analysis (KDA) and SVM. The algorithm firstly reduces the high dimensional Web document space in the training sets to the lower dimensional space with KDA algorithm, then the classification and predication in

* Corresponding author.

the lower dimensional feature space are implemented with the multiplicative update-based optimal SVM. Overall, theses algorithms that are mentioned above all have a long learning time. The implement of algorithms are complicated fairly. The accuracy of document classification still has possibility for improvement.

In order to improve the accuracy of document classification and the convergence time of algorithm, a multi-layered immune based document classification algorithm, referred to as *MIDC*, is presented. Experiments show that the algorithm has higher classification accuracy than other document classification methods, and the attractive features such as diversity, self-learning, adaptive, and robust, etc.

2 Proposed Algorithm Model

MIDC references the part mechanism that the adaptive biological immune system responses antigens. When the antigen enters the immune system, it first contacts with free antibodies that the *B-cells* secret on the surface. Then, some of free antibodies attempt to recognize the antigen. When the amount reaches a pre-set threshold, which free antibodies can recognize the antigen, the corresponding *B-cells* are activated. The activated *B-cells* are cloned and mutated. The process produces many new *B-cells* that have higher affinity with the antigen. Finally, the new *B-cells* will clear the antigens, and the highest affinity *B-cell* will evolve into memory cell by competing, which can enhance the speed of secondary immune response.

Before the presenting of *MIDC*, we represent firstly the terminologies, symbols, as well as formulas.

Let the antigen represent an unlabeled document. The antigenic determinant is from extracting the features of document. The process is as same as antigen presenting cells in immune system.

Ag is defined as antigens set. $Ag \subset D$, $D=S^l \subseteq R^l$, where *S* is the shape space of antigens, *l* is the dimensionality of shape space of antigen, and *R* is the real space. Let *ag* represents an antigen, such that $ag \in Ag$.

Ab is defined as *B-cells* set. $Ab= \{<d, t, c>|d \in D, t, c \in N\}$, where *t* is the life cycle of *B-cell*, *c* is the class of *B-cell*, and *N* is the set of natural numbers. Let *ab* represents a *B-cell*, such that $ab \in Ab$.

Mc is defined as memory cells set, $Mc \subseteq Ab$. Let *mc* represents a memory cell, such that $mc \in Mc$.

Let *f*(*x*,*y*) represents the affinity function of *x* with *y*, where *x* or *y* may be an antigen, *B-cell*, memory cell etc. *f*(*x*,*y*) is defined as Eq.(1).Where d_k is the *k* attribute value of *d*.

$$f(x, y) = 1/(1 + \sqrt{\sum_{k=1}^{l} (x.d_k - y.d_k)^2 / l}) \tag{1}$$

Let *sAb* represents a subset of *B-cells*. $sAb \subset Ab$, and $|sAb|=\eta^*|Ab|$, where η is a coefficient, such that $0<\eta<1$.

2.1 The Initialization

In order to simplify the calculation and improve the efficiency, *MIDC* is divided into three layers, which are named as the freedom antibodies layer, the *B-cells* layer, and the memory cells layer, respectively. The cell competes internally with each other in layers. The layer has a feedback with interrelationship with each other.

The algorithm initialization is to establish the freedom antibodies layer, the *B-cells* layer, and the memory cells layer of immune system, respectively. First, *MIDC* selects randomly *n* learning samples to build the *B-cells* layer from the antigens set. *Ab* represents the antibodies set of the *B-cells* layer. The initial life cycle of *B-cell* is zero. Then, *MIDC* selects *m* randomly *B-cells* to secrete freedom antibodies from the *B-cells* layer. The freedom antibodies compose the freedom antibodies layer. *Fa* represents the freedom antibodies set, and is defined as Eq.(2), such that $m=|Fa|$.

$$Fa=\{x \mid x\in Ab, \exists y\in Ab(Dist(x,y)>\beta)\wedge x\neq y\} \tag{2}$$

Where $Dist(x,y)$ is the function of distance between x and y, β is the mean distance of *Ab*. $Dist(x,y)$ and β are defined as Eq.(3) and Eq.(4), respectively.

$$Dist(x,y)=\sum_{k=1}^{l} |x.d_k - y.d_k| / l \tag{3}$$

$$\beta=\frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} Dist(ab_i, ab_j)}{\frac{n(n-1)}{2}} \tag{4}$$

Finally, *MIDC* selects antibodies to build the memory cells layer from *Ab*. *Mc* represents the antibodies set of the memory cells layer, which is defined as Eq.(5).

$$Mc=\{x \mid x\in Ab, \exists y\in Ab(f(y,x)>\frac{1}{1+\beta}\wedge x.c=y.c\wedge x\neq y)\} \tag{5}$$

After the multi-layered immune system established, *MIDC* begins to learn each one of the sample antigens for immunization. This process is as following.

2.2 The Immune Learning

In this process, the sample antigens contact firstly with the freedom antibodies, and then enter the *B-cells* layer. Finally, the activated *B-cells* are cloned and mutated. Some of the new *B-cells* become memory cells into the memory cell layer.

2.2.1 The Immune Response

The response can be divided into primary response and secondary response. Firstly, *MIDC* selects randomly a subset to contact with the antigen from *Fa*. φFa is defined to represent the subset. *Rec* is defined as the result of the freedom antibody contacting with the antigen. If the affinity of the freedom antibody with the antigen is more than a computing threshold δ, the freedom antibody recognizes the antigen, and the value of *Rec* is 1. Otherwise the value of *Rec* is 0. δ and *Rec* is defined as Eq.(6) and Eq.(7).

$$\delta = \frac{\sum_{i=1}^{n-1} f(ab, ab_i)}{(n-1)} \quad iff\ ab \in Fa \quad and \quad n = |Fa| \tag{6}$$

$$Rec(ab, ag) = \begin{cases} 1 & iff \quad f(ab, ag) \geq \delta \\ 0 & \end{cases} \quad ab \in \varphi Fa \tag{7}$$

MIDC needs to sum up the number of freedom antibodies that can identify antigen. If the sum exceeds a pre-set threshold, the activated *B-cells* respond the antigen secondarily. Otherwise, they do primary one. Sum is defined as Eq.(8).

$$Sum = \sum_{i=1}^{n} Rec(ab_i, ag) \quad ab_i \in \varphi Fa \tag{8}$$

At the same time, if none of freedom antibodies can recognize the antigen, it is a new antigen. *MIDC* executes the similar primary response, and clone the antigen to the *B-cells* layer and the freedom antibodies layer. However, if few of freedom antibodies can recognize the antigen, it will not trigger the immune response; the corresponding *B-cells* will secrete some antibodies to the freedom antibodies layer. The process is repeated continuously. The immune response will eventually trigger.

2.2.2 The Immune Regulation

MIDC first selects the activated *B-cells* from *Ab* in accordance with Eq.(9). Where the activated *B-cells* can recognize the antigen, and have the same class with the antigen. The activated *B-cell* is referred as ab_{stim}, such that $ab_{stim} \in Ab_{stim}$.

$$Ab_{stim} = \{ab \mid ab \in \varphi Fa, ab \in Ab \wedge f(ab, ag) \geq \delta \wedge ab.c = ag.c\} \tag{9}$$

Then, ab_{stim} will be cloned and mutated. The clone amount of ab_{stim}, referred to as *Count*, is based on the affinity of ab_{stim} with the antigen. The higher the affinity, the greater stimulation ab_{stim} has, and the more the amount. The mutation probability of ab_{stim}, referred to as *Rate*, is based on the affinity of ab_{stim} with the antigen. The higher the affinity, the fewer stimulation ab_{stim} has, and the lower the probability. *Count* and *Rate* are calculated by Eq.(10) and Eq.(11), respectively.

$$Count = \lambda \times Sum \times f(ab_{stim}, ag) \tag{10}$$

$$Rate = \zeta \frac{1}{Sum + f(ab_{stim}, ag)} \tag{11}$$

λ is the cloning constant, which is used to ensure that the new *B-cells* can be generated enough. ζ is the mutating constant, which is used to preserver the antibody diversity in the *B-cells* layer.

After each ab_{stim} is cloned and mutated, the scale of the *B-cells* expands rapidly. In order to suppress the concentration of certain class of *B-cells*, *MIDC* needs to eliminate some of *B-cells*. If the *B-cells* have not been activated during the threshold of the life cycle, *MIDC* would eliminate the *B-cells* from *Ab*, and the corresponding freedom antibodies from the freedom antibodies layer.

Finally, Ab_{stim} secrete freedom antibodies to the freedom antibodies layer. The number of freedom antibodies secreted is defined as Eq.(13), referred to as *Num*.

$$S_{Max} = Max(Dist(ab_i, ab_j)) - Min(Dist(ab_k, ab_l)) \tag{12}$$

$$Num = \gamma(S_{Max} - Dist(ab_{stim}, ag)) \tag{13}$$

S_{Max} is the farthest distance difference between *B-cell* antibodies in *Ab*. ab_i, ab_j, ab_k, and ab_l. represent different antibodies, respectively. It shows that the more specific of a *B-cell*, the greater freedom antibodies that it need to ensure to be triggered during the next response. γ is a constant to ensure the concentration of freedom antibodies.

2.2.3 The Immune Memory

After immune regulation, *MIDC* selects a *B-cell* as candidate memory cell from *Ab*, referred to as ab_{cand}. ab_{cand} has the highest affinity and the same class with the antigen. Before ab_{cand} enters the memory cell layer, it must compete with *mc*. First, *MIDC* selects a memory cell as candidate cell from *Mc*, referred to as mc_{cand}. mc_{cand} has the highest affinity and the same class with the ab_{cand}. If the affinity of ab_{cand} with mc_{cand} is more than a computing threshold, referred to as θ, and the affinity of ab_{cand} with the antigen is the highest, ab_{cand} enters the memory cell layer to replace mc_{cand}. Otherwise, ab_{cand} enters the memory cells layer. θ is calculated by the Eq.(14).

$$\theta = \frac{\sum_{i=1}^{n} f(ab_{cand}, mc_i)}{n} \quad iff \;\; mc_i.c = ab_{cand}.c \tag{14}$$

2.3 The Document Classification

After the immune learning, the memory cell layer's evolvement is finished. *MIDC* classifies unlabeled documents according to *Mc*. The class of unlabeled document is determined by *mc*, which has the highest the affinity with it. First of all, *MIDC* calculates the affinity of the unlabeled document with each *mc* according to Eq.(1). Then, *MIDC* selects the highest affinity *mc*. The class of the unlabeled document is the same as selected *mc*.

3 The Simulation Experiments

3.1 The Experimental Dataset

The experiment uses the document classification dataset, referred to as *OHSCAL*, which is a subset of *OHSUMED* [4]. *OHSUMED* is from the medical information database as *MEDLINE10*, contains 270 pharmaceutical journal title and / or summary from 1987 to 1991, and includes 348,566 documents. *OHSCAL* is from ten classes of *OHSUMED*, and includes 11,162 documents. In experiment, *MIDC* remove the common words of documents using Porter's suffix-stripping algorithm [5].

The experiment selects randomly 9,000 learning item as the training set from the data set, the other 2162 learning items as a test set. The experimental parameters λ=1.80, ζ=96, and γ=3, respectively.

3.2 The Results and Discussion

The results of experiment are shown in Fig. 1 and Fig. 2. Such as Fig. 1, *MIDC* achieves 14.2% documents classification error rate, with the increase of the size of memory cells layer, the error rate of documents classification is decreased. However, when the size of memory cells layer is over a certain value, the error rate is stable to 14.2%, because the memory cells layer is convergent.

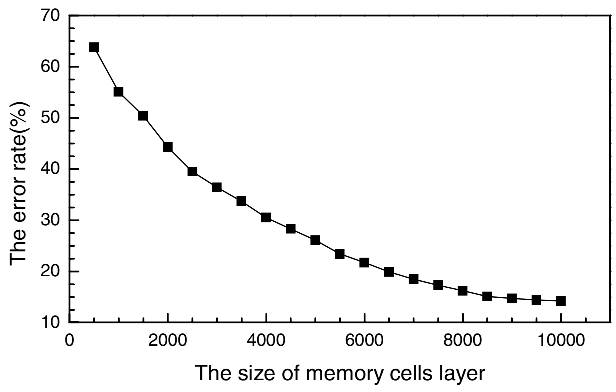


Fig. 1. The error rate of documents classification

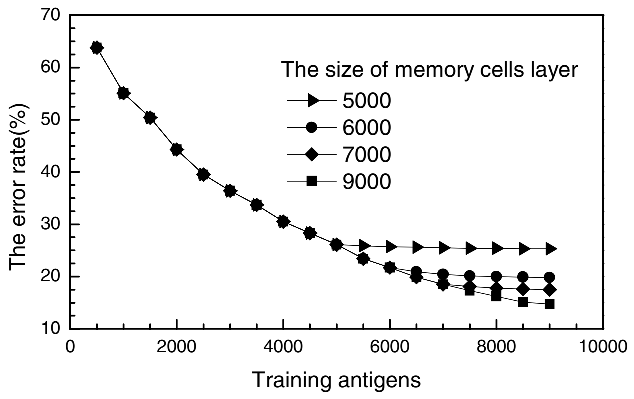


Fig. 2. The error rate achieved by the different size of memory cells layer

In the Fig. 2, the experiments shows the different, when the size of memory cells layer is changed to 5000, 6000, 7000, and 9000, respectively. With the size of memory cells layer increased, the error rate of documents classification is decreased. However, when the size of memory cells layer is a certain value about 9000, with the increase of training antigens, the error rate of documents classification results is stable, because the memory cells layer is already convergent.

In order to prove that *MIDC* decreases the error rate of documents classification, Table 2. shows the comparison of classification accuracy, compared with the different document classification algorithms. The accuracy of other document classification algorithms is from the literature [6]. *MIDC* shows that the classification accuracy is the highest on *OHSCAL* dataset.

Table 1. The error rate of documents classification on OHSCAL dataset

Algorithm	Classification Accuracy (%)
NB	74.6
C4.5	71.5
kNN	62.5
Cntr	75.4
MIDC	85.8

4 Conclusion

In this paper, a novel document classification algorithm based on multi-layered immune, referred as *MIDC*, is presented. Experiments show that *MIDC* has higher classification accuracy than other traditional document classification algorithms, which provides a new solution for document classification. Finally, if the immune cell regard as a given pattern, and the antigens set is reasonable adjusted, *MIDC* can also be applied to areas such as virus detection, pattern recognition, and etc.

References

1. Wang, P., Domeniconi, C.: Building semantic kernels for text classification using wikipedia. In: International Conference on Knowledge Discovery and Data Mining (KDD 2008), New York, USA (2008)
2. Ifrim, G., Theobald, M., Weikum, G.: Learning word-to-concept mappings for automatic text classification. In: Learning in Web Search Workshop, ICML (2005)
3. Wang, Z.Q., Qian, X.: Document classification algarithm based on KDA and SVM. Journal of Computer Application 29(2), 416–418 (2009)
4. Hersh, W., Buckley, C., Leone, T.J., Hickam, D.: OHSUMED: An interactive retrieval evaluation and new large test collection for research. In: SIGIR 1994, pp. 192–201 (1994)
5. Porter, M.F.: An algorithm for suffix stripping. Program 14(3), 130–137 (1980)
6. Eui-Hong, S.H., George, K.: Centroid-Based Document Classification: Analysis and Experimental Results. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 424–431. Springer, Heidelberg (2000)

A Quantum Immune Algorithm for Multiobjective Parallel Machine Scheduling

Zhiming Fang

Zhijiang College, Zhejiang University of Technology, Hangzhou 310024, China
fangzm_55@163.com

Abstract. The study presents a novel quantum immune algorithm (QIA) for solving the parallel machine scheduling in the textile manufacturing industry. In this proposed algorithm, there are distinct characteristics as follows. First, the encoding method is based on Q-bit representation. Second, a novel mutation operator with a chaos-based rotation gate is proposed. Most importantly, two diversity schemes, suppression algorithm and similarity-based truncation algorithm, are employed to preserve the diversity of the population, and a new selection scheme is proposed to create the new population. Simulation results show that QIA is better than two quantum-inspired evolutionary algorithms.

Keywords: immune algorithm, multiobjective optimization, quantum computing, knapsack problem.

1 Introduction

For the multiobjective scheduling problems in the textile manufacturing industry, there are the following characteristics: (1) parallel machines are nonidentical, and (2) the type of jobs processed on each machine can be restricted. In 2008, Gao et al. [1] studied them and presented a multiobjective scheduling model subjected to special process constraint on parallel machines with the following objectives: makespan, total earliness, and total tardiness, and proposed a vector genetic algorithm (VGA) to solve it effectively. Furthermore, based on VGA and immune theory, Gao et al. presented an immune vector genetic algorithm [2]. An immune operator was adopted to guarantee the diversity of the population, and a local search algorithm was applied to improve the quality of the population. However, all approaches suggested by Gao et al. transform the multiobjective problem into a single objective one through a linear combination of objectives and weights. Here, we study the scheduling of the textile manufacturing industry with the following two objectives: total completion time and total tardiness. However, different from the literature [2], we will not find a single optimal solution but Pareto-optimal solutions.

Quantum computing is a research area that includes concepts like quantum-mechanical computers and quantum algorithms, and research on merging evolutionary algorithms (EAs) with quantum computing has been started since late 1990s [3,4,5]. Recently, quantum computing was combined with other intelligent algorithms such as neural network [6], ant algorithm [7] et al, and some people proposed some quantum-inspired EAs [4,5]. However, there has been relatively little work done in merging immune algorithm and quantum computing.

In this paper, the multiobjective scheduling problems in the textile manufacturing industry will continue to be studied and a novel quantum-inspired immune algorithm (QIA) based on immune system and quantum computing is presented. In this proposed algorithm, the encoding method is based on Q-bit representation, and a novel mutation operator with a chaos-based rotation gate is proposed. Especially, two diversity schemes, suppression algorithm and similarity-based truncation algorithm, are employed to preserve the diversity of the population, and a new selection scheme is proposed to create the new population.

2 The Proposed Algorithm

In this section, we will present a novel quantum-inspired immune algorithm (QIA) based on immune system and quantum computing. The detailed procedures are listed as follows.

(1) A random initial population P_0 of size N_{pop} is created and set the memory $N_{\text{mem}} = \emptyset$. Set a counter $t = 0$.
(2) For each cell in the population, reproduce the cell N_{clones} times and mutate each clone by a chaos-based rotation gate.
(3) Combine the population P_t and the offspring as R_t.
(4) Evaluate R_t, and perform trimming operation for R_t. Compute all nondominated cells in R_t, and store them to N_{mem}, and then delete all dominated cells in N_{mem}.
(5) Select new population P_{t+1} from R_t by using the selection scheme based on a similarity-based truncation algorithm.
(6) Randomly select a cell from N_{mem}, and use it to update the new population P_{t+1} by a rotation gate.
(7) if $t > T$ (the maximum number of generations) or another termination condition is satisfied, then terminate the algorithm and output the nondominated cells in N_{mem}. Otherwise, let $t = t + 1$ and return to (2).

In this proposed algorithm, we only introduce the encoding and decoding technique, the method of initiating the population, a mutation operator with a chaos-based rotation gate, and the update scheme with a rotation gate. For the other aspects, please refer to the literature[8,9].

2.1 Encoding and Decoding Technique

For QIA, the encoding technique based on Q-bit is utilized. Each bit for a cell with size n is represented as a string of m Q-bits in the following:

$$\left[\begin{array}{c|c|c|c} \alpha_1 & \alpha_2 & \cdots & \alpha_m \\ \beta_1 & \beta_2 & \cdots & \beta_m \end{array}\right]$$

where $|\alpha_i|^2 + |\beta_i|^2 = 1$, $i = 1, 2, \ldots, n$. For the Q-bit string, it can be obviously seen that it can not be directly used to calculate the objective values. Thus, a random key decoding technique is applied. According to [4], a Q-bit string is first converted to a binary string. In particular, for the jth ($j = 1, 2, \cdots, m$) bit of Q-bit string, a random number τ is generated at interval [0,1]. If $|\beta_j|^2 > \tau$, then let the corresponding bit ν_j of the binary string be 1; otherwise let ν_j be 0. For the parallel scheduling problems, such a binary string should be further converted to a random-key representation. For example, considering a seven-job three-machine problem (general,we let m = number of machines, and n = number of jobs). Suppose the resultant binary string is [0 1 1| 1 0 0| 1 1 1| 1 0 1| 1 1 0| 1 1 1| 0 1 0]; thus its random-key representation is [2 1 3 2 2 3 1]. Obviously, a scheduling scheme can be easily obtained by such a random-key representation.

2.2 Initializing the Population

Here a chaos-based method initializing the population is used in order to obtain the population with a better characteristic of diversity. Chaos is an universal natural phenomenon, and can be generated by the following mathematic model (Logistic mapping):

$$x_{k+1} = \mu x_k(1 - x_k), \tag{1}$$

where $\mu = 4$. From (1), the chaos variable x_k ($k = 1, 2, \cdots$) will be obtained, and $x_k \in [0, 1]$. Compared to randomicity, chaos has better universality of search space.

In the following, we list the method initializing the population on the basis of (1).

(1) $i = 1$.
(2) Generate $n \times m$ different random numbers x_j^0 ($j = 1, \cdots, n \times m$) at interval [0,1].
(3) Using the following equation to calculate x_j^i.

$$x_j^i = \mu x_j^{i-1}(1 - x_j^{i-1}), \tag{2}$$

(4) Let $\alpha_j^i = \cos(2x_j^i\pi)$ and $\beta_j^i = \sin(2x_j^i\pi)$, and then α_j^i and β_j^i form a Q-bit $\begin{bmatrix} \alpha_j^i \\ \beta_j^i \end{bmatrix}$, $j = 1, 2, \cdots, n \times m$,
(5) A cell is made of $n \times m$ Q-bits and can be expressed as $\left[\begin{array}{c|c|c|c} \alpha_1^i & \alpha_2^i & \cdots & \alpha_{n\times m}^i \\ \beta_1^i & \beta_2^i & \cdots & \beta_{n\times m}^i \end{array}\right]$.
(6) $i = i + 1$.
(7) If $i > N_{\text{pop}}$, then N_{pop} cells are obtained and the procedure is terminated. Otherwise, return to (3).

2.3 Mutation Operator

For QIA, a chaos rotation Q-gate is utilized to perform mutation for each cell. Every Q-bit is mutated with a predefined probability p_{m}. Assume that a parent cell is $\left[\begin{array}{c|c|c|c}\alpha_1 & \alpha_2 & \cdots & \alpha_{n\times m} \\ \beta_1 & \beta_2 & \cdots & \beta_{n\times m}\end{array}\right]$, its mutation procedure is listed as follows.

(1) $j = 1$
(2) Generate a random number η at interval $[0, 1]$.
(3) If $\eta < p_{\mathrm{m}}$, (α_j,β_j) of the jth Q-bit is updated as follows.

$$\begin{bmatrix}\alpha_j^{'} \\ \beta_j^{'}\end{bmatrix} = \begin{bmatrix}\cos(\Delta\theta_j) & -\sin(\Delta\theta_j) \\ \sin(\Delta\theta_j) & \cos(\Delta\theta_j)\end{bmatrix}\begin{bmatrix}\alpha_j \\ \beta_j\end{bmatrix} \tag{3}$$

where $\Delta\theta_j$ is the angle parameter of rotation Q-gate, and $\Delta\theta_j = \lambda x_k^j$. x_k^j is a chaos variable, and be calculated by the following equation:

$$x_k^j = 8x_{k-1}^j(1 - x_{k-1}^j) - 1. \tag{4}$$

From (4), it can be observed that $\Delta\theta_j$ is within the range $[-\lambda, \lambda]$.
(4) $j = j + 1$.
(5) If $j > n \times m$, then terminate the procedure. Otherwise, return to (2).

In the above procedure, λ is an amplitude, and its value depends on the rank of the parent cell in the population. Assume that the number of nondominated fronts in the population is K, for a parent cell, if it is on the ith front ($i = 1, 2, \cdots, K$),

$$\lambda = \lambda_0 \exp((i - |K|)/|K|), \tag{5}$$

where λ_0 is a control parameter. From(3)-(5), we can see that for a parent cell, the better its rank is, the smaller the chaos disturb is. And vice versa.

2.4 Rotation Gate for Q-Bit in QIA

The rotation gate is only used in QIA to adjust the probability amplitudes of each Q-bit. According to the rotation gate in (3), a quantum-gate $U(\theta_i)$ is a function of $\theta_i = s(\alpha_i, \beta_i) \cdot \Delta\theta_i$, where $s(\alpha_i, \beta_i)$ is the sign of θ_i that determines the direction and $\Delta\theta_i$ is the magnitude of rotation angle. Like the literature [3,4], the angle parameters used for the rotation gate are selected from the lookup table (see Table 1). In Table 1, r_i and b_i are the ith bits of a binary solution r and a nondominated solution b randomly selected from the memory , respectively. Because b comes from the nondominated set in QIA, the use of Q-gate is to emphasize the searching direction toward b, which will be helpful for enhancing the quality of the population.

Table 1. Lookup table of rotation angle

r_i	b_i	$f(r) < f(b)$	$\Delta\theta_i$	$s(\alpha_i, \beta_i)$			
				$\alpha_i\beta_i > 0$	$\alpha_i\beta_i < 0$	$\alpha_i = 0$	$\beta_i = 0$
0	0	true	0	0	0	0	0
0	0	false	0	0	0	0	0
0	1	true	0.05π	+1	-1	0	±1
0	1	false	0	0	0	0	0
1	0	true	0.01π	-1	+1	±1	0
1	0	false	0.025π	+1	-1	0	±1
1	1	true	0.005π	+1	-1	0	±1
1	1	false	0.025π	+1	-1	0	±1

3 Simulation Experiments

In this section, we apply QIA to parallel machine scheduling in the textile manufacturing industry, and compare the performance of QIA with and HQGA [3] and WBMOIGA [10]. All experiments are conducted on an IBM computer, which is equipped with a Pentium IV 2.8G processor and 2 GB of internal memory. The operating system is Windows XP and the programming language is C++. The compiler is Borland C++ 6.0.

3.1 Test Problems and Performance Measure

We use the following two objective scheduling problem subjected to special process constraint on parallel machines with fifteen jobs and four machines as our test problem.

Test Problem:
min $f_1(\mathbf{x})= 2 \times$ (total completion time)
min $f_2(\mathbf{x})= 8 \times$ (total tardiness)

In this experiment, We use two test data (denoted by I and II), which come from the literature [10], and the two metrics (Coverage and Ratio)[11] to investigate the performance of QIA.

3.2 Experimental Results

For a given test problem, each algorithm is randomly run 20 times. We show the average of the ratio metrics of the 20 trials for each algorithm for Test Problems I and II in Tables 2 and 3, respectively. The average of the coverage metrics of the 20 trials for each algorithm for Test Problems I and II is shown in Tables 4 and 5, respectively. From Tables 2 and 3, we can observe that QIA obtains more solutions than WBMOIGA and HQGA, and the ratio is also bigger than them. Moreover, From Tables 4 and 5, it can be seen that the set coverage metrics C(QIA,WBMOIGA) and C(QIA,HQGA) are bigger than C(WBMOIGA,QIA)

Table 2. Average ration for Test Problem I

Algorithm	QIA	WBMOIGA	HQGA
$\lvert B_i \rvert$	8.55	6.45	5.65
$\lvert A_i \rvert$	8.15	4.65	2.85
Ratio ($\text{Ratio}_i = \frac{\lvert A_i \rvert}{\lvert B_i \rvert}$)	95.3%	72.0%	50.4%

Table 3. Average ration for Test Problem II

Algorithm	QIA	WBMOIGA	HQGA
$\lvert B_i \rvert$	9.57	6.95	5.75
$\lvert A_i \rvert$	9.25	3.20	2.05
Ratio ($\text{Ratio}_i = \frac{\lvert A_i \rvert}{\lvert B_i \rvert}$)	96.6%	46.0%	35.6%

Table 4. Average coverage for Test Problem I

Algorithm	QIA	WBMOIGA	HQGA
QIA	—	0.89513317	0.94626893
WBMOIGA	0.51436313	—	0.64650325
HQGA	0.31475521	0.44727468	—

Table 5. Average coverage for Test Problem II

Algorithm	QIA	WBMOIGA	HQGA
QIA	—	0.88122923	0.92123281
WBMOIGA	0.41764625	—	0.66332017
HQGA	0.30311923	0.41729368	—

and C(HQGA,QIA), respectively. According to the definitions of metrics, we can draw a conclusion that QIA has higher performance than WBMOIGA and HQGA.

4 Conclusion

In this study, we present a quantum immune algorithm. By applying to the parallel machine scheduling, we show that QIA has higher performance than WBMOIGA and HQGA. However, how to select the parameters of QIA can not be considered in this paper. It will be researched in the future work.

References

1. Gao, J.Q., He, G.X., Wang, Y.S.: Multi-objective scheduling problems subjected to special process constraint. In: Proc. IEEE Congr. Evolutionary Computation, Hongkong, China, pp. 105–110. IEEE Press, Los Alamitos (2008)

2. Gao, J.Q., He, G.X., Wang, Y.S.: A new parallel genetic algorithm for solving multiobjective scheduling problems subjected to special process constraint. Int. J. Adv. Manu. Tech. 43(1-2), 151–160 (2009)
3. Li, B.B., Wang, L.: A hybrid quantum-inspired genetic algorithm for multiobjective flow shop scheduling. IEEE Trans. Evol. Comput. 37(3), 576–591 (2007)
4. Han, K.H., Kim, J.H.: Quantum-inspired evolutionary algorithms with a new termination criterion, h_ε gate, and two-phase scheme. IEEE Trans. Evol. Compt. 8(2), 156–169 (2004)
5. Kim, Y., Kim, J.H., Han, K.H.: Quantum-inspired multiobjective evolutionary algorithm for multiobjective 0/1 knapsack problems. In: Proc. 2006 IEEE Congress on Evolutionary Computation, Vancouver, Canada, pp. 2601–2606. IEEE Press, Los Alamitos (2006)
6. Li, P.C., Li, S.Y.: Learning algorithm and application of quantum BP neural networks based on universial quantum gates. J. Sys. Eng. Electron. 19(1), 167–174 (2008)
7. Li, P.C., Li, S.Y.: Quantum ant colony algorithm for continuous space optimization. Chinese J. Contr. Theory Appl. 25(2), 237–241 (2008)
8. Gao, J.Q., Wang, J.: WBMOAIS: A novel artificial immune system for multiobjective optimization. Comput. Oper. Res. 37(1), 50–61 (2010)
9. Gao, J.Q., Fang, L., Wang, J.: WBMOAIS: A weight-based multiobjective immune algorithm: WBMOIA. Eng. Optimiz., doi:10.1080/03052150903406563 (to appear)
10. Gao, J.Q.: WBMOIGA: Weight-based multiobjective immune genetic algorithm and its application. In: Proc. IEEE International Conference on Information and Automation (ICIA 2009), Hai Zhu, China, pp. 1–6. IEEE Press, Los Alamitos (2009)
11. Deb, K.: Multi-objective Optimization Using Evolutionary Algorithms. John Wiley & Sons, Ltd., New York (2001)

A Resource Limited Immune Approach for Evolving Architecture and Weights of Multilayer Neural Network

Xiaoyang Fu[1], Shuqing Zhang[2], and Zhenping Pang[3]

[1] College of Computer Science and Technology, Jilin University, Zhuhai 519041, China
dvndavidfu@vip.163.com
[2] Northeast Institute of Geography and Agroecology, Chinese Academy of Sciences, Changchun 130012, China
[3] College of Computer Science and Technology, Jilin University, Zhuhai 519041, China

Abstract. A resource limited immune approach (RLIA) was developed to evolve architecture and initial connection weights of multilayer neural networks. Then, with Back-Propagation (BP) algorithm, the appropriate connection weights can be found. The RLIA-BP classifier, which is derived from hybrid algorithm mentioned above, is demonstrated on SPOT multi-spectral image data, vowel data and Iris data effectively. The simulation results demonstrate that RLIA-BP classifier possesses better performance comparing with Bayes maximum-likelihood classifier, k-nearest neighbor classifier (k-NN), BP neural network (BP-MLP) classifier and Resource limited artificial immune classifier (AIRS) in pattern classification.

Keywords: resource limited immune approach (RLIA), evolutionary artificial neural network (EANN), pattern classification.

1 Introduction

Pattern classification is an important problem in the multi-spectral remote sensing image research, phonetics, handwriting, etc. So far, besides conventional classifiers such as Bayes classifier, k-NN classifier[1], several pattern recognition classifiers has been adopted, for example, Genetic classifier[2], neural network classifier[3], evolutionary neural network classifier[4],AIRS classifier[5].

The most widely used neural network model is the multilayer perceptron (MLP), in which the connection weights training is normally completed by a BP learning algorithm. The essential character of the BP algorithm is gradient descent. Because the gradient descent algorithm is strictly dependent on the shape of the error surface, the error surface may have some local minimum and multimodal. This results in falling into some local minimum and appearing premature convergence [7].

Genetic Algorithms (GAs) [8], the optimization techniques guided by the principles of evolution and natural genetics, are efficient, adaptive and robust searching processes, producing near-optimal solution and handling large, highly complex and multimode spaces. Of course, it is not surprise to evolve the neural network with GA, and to study pattern classification using GA evolving neural networks [4, 6].

However, crossover and mutation operators, the two main genetic operators of GA, are searched randomly and iteratively in a certain probability so that premature may be appeared and community diversity is reduced while an individual in population is being evolved [9].

Artificial immune systems (AIS) [9-10] are an intelligent algorithm inspired by human immune system. They possess several properties such as noise tolerance, unsupervised learning, self-organization and immune memory etc. Resource limited artificial immune classifier (AIRS) [5] based on clonal selection theory adopts the conception of artificial recognition balls (ARBs). The training of an antigen (Ag) and the processing of each ARB are conducted through a resource allocated mechanism according to the simulation level between the ARB and Ag. If the total resource allocated across the ARB population exceeds the allowed limit, the weakest ARBs are removed until the totality of allocated resources is under the limit. The ARBs further produce offspring through clonal mutation until the average stimulation value for all the existing ARBs reaches the pre-set threshold. Once the training process for current Ag is fulfilled, the best matching ARB will become a long-lived memory cell if it matches the Ag better than the mc_{match} does. The algorithm continues until all training Ag have been presented and trained. Finally, the evolved memory cells will be used for classification through k-nearest neighbor approach. AIRS has got better result for classification of multi-spectral remote sensing image [11].

In this paper, a resource limited immune approach (RLIA) was developed to evolve an appropriate architecture and initial connection weights of multilayer neural networks. The optimal connection weights can be further trained by BP algorithm.

2 Method

2.1 The Three-Layer Neural Network

Assuming a three-layer neural network with ***m*** inputs (features), ***q*** outputs (categories), and ***l*** hidden nodes, see Fig 1. The relations of input and output can be formulated as follows:

$$net_j = \sum_{i=1}^{m} W_{ji} x_i + w_{j0} \tag{1}$$

$$y_j = f(net_j) \tag{2}$$

$$net_k = \sum_{j=1}^{l} V_{kj} y_j + v_{k0} \tag{3}$$

$$O_k = f(net_k) \tag{4}$$

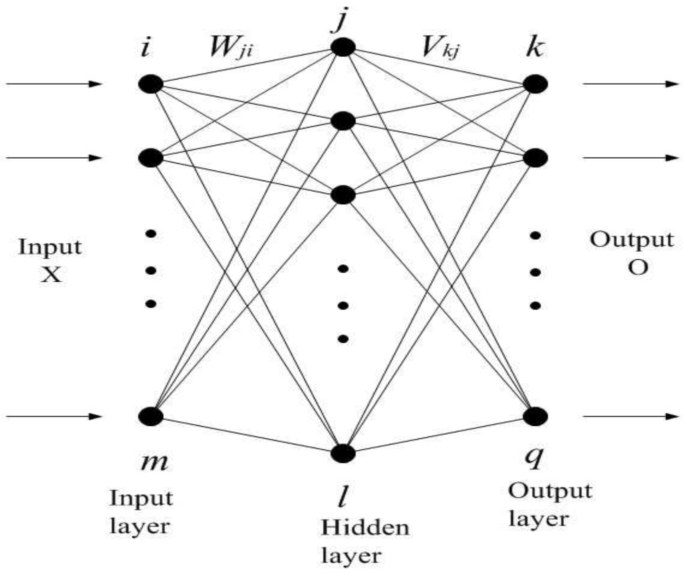


Fig. 1. Three layers neural network structure

where W_{ji} is the connection weight between the j^{th} hidden node and the i^{th} input node ,v_{j0} is its bias; the net_j is the input of the j^{th} hidden node; V_{kj} is the connection weight between the j^{th} hidden node and the k^{th} output node, v_{k0} is its bias. net_k is the input of the k output node. $f(net)$ is a sigmoid activation function, which is defined as:

$$f(net) = \frac{1}{1 + e^{(-net)}} \tag{5}$$

where $net \in [-\infty, +\infty]$.

Suppose we have a set of training patterns $X=\{X_1, X_2, \ldots, X_n\}$, where n is the number of training patterns, each training pattern X_i in set X is an m-dimensional feature vector. Let $T= \{T_1, T_2, \ldots, T_n\}$ as set X's corresponding output classes, $T_i=\{t_1, t_2, \ldots, t_q\}$ is a q-dimensional class vector. If the target class for a specific pattern is k $(1 \leq k \leq q)$, then $t_k=1$; otherwise, $t_k=0$. Let's denote O_{ik} as the i^{th} actual neuron output for input training pattern x_i at the output node k while t_{ik} as its desired response. The mean square error function (MSE) for this neural network is defined as:

$$MSE(W) = \frac{1}{nq} \sum_{i=1}^{n} \sum_{k=1}^{q} (t_{ik} - o_{ik})^2 \tag{6}$$

where W represents all the weights in the network. Thus, this error is some scalar function of the weights and is minimized when the network output matches the desired outputs.

2.2 RLIA-BP Classifier

RLIA-BP algorithm includes the following five steps:

1. Normalization of the training samples and initialization of ARBs and memory cells.
2. ARB generation
3. Competition for resources and selection of candidate memory cells.
4. Promoting candidate cells to memory cell pool. Do the step 2 to 4 repeatedly until some stopping criterion is met.
5. Continue BP training and classification.

These steps are detailed as followings:

2.2.1 Normalization and Initialization

1. Normalization all samples

A simple method can be adopted so that the average over the training set of each feature is zero and the variance is 1.0. Naturally, the testing data must be subjected to the same transformation.

2. Calculating affinity and stimulation

In RLIA algorithm, an antibody cell (ARB) represents a neural network with a certain architecture and connection weight values, let *AB* represent the set of ARBs and ab is single ARB, $ab \in AB$.

$$AB = (ab_1, ab_2, \cdots, ab_i, \cdots ab_p) \tag{7}$$

$$ab_i = (l_i, W^i, V^i) \tag{8}$$

where l_i is the number of hidden nodes of the i^{th} ARB. W^i and V^i represent input and output connection weights respectively. The affinity between a set of training samples (*ags*) and an antibody (*ab*) can be described by the mean square error function (*MSE*) for this neural network.

Define: affinity (*ags,ab*)=MSE (9)
Stimulation (ags,ab) =1-affinity(*ags,ab*)
=1-*MSE* (10)

Thus, the stimulation of an ARB represents the fitness of the neural network for the training set.

3. Initialization

In initialization, the connection weights (*W* and *V*) values are randomly chosen from range[-1,+1] and the number of hidden nodes(*l*) are randomly chosen from range[$l_{\min}, l_{\max}$]. We can obtain a set of ARBs for initial set. The highest stimulation ARB with the same architecture from *AB* sets is selected as a memory cell. These memory cells with different architectures constitute the initial set of memory cells *(MC)*.

By convenient rule of thumbs [12], the number of hidden nodes is chosen such that the total number of weights in network is roughly *n*/10, or more than this number, but it should not be more than the total number of samples, n. For the three-layer neural network, the total number of weights (exception bias) is $l/(m+q)$, so the maximum and minimum number of hidden nodes will be $l_{\max} \le n/(m+q)$, $l_{\min} = l_{\max}/10$.

2.2.2 ARBs Generation

First, find the memory cell, mc_{match}, which has highest stimulation in *MC*.

$$mc_{match} = \text{argmax}_{mc \in MC}\ \text{stimulation}(ags, mc) \tag{11}$$

Once the memory cell with highest stimulation, mc_{match}, is identified, it generates new ARBs to place into the population of pre-existing ARBs through mc_{match} clone. The clones are defined as:

$$\text{NClones}=\text{hyper\_clonal\_rate}*\text{clonal\_rate}*\text{stimulation}(ags,mc_{match}) \tag{12}$$

where hyper_clonal_rate and clonal_rate are pre-set values by user. The hyper_clonal_rate is a multiplier which ensures that a hyper-mutating memory cell produce more new cells than a standard ARB. It will increase the AB population diversity.

2.2.3 Competition for Resources and Selection of Candidate Memory Cells

The detailed training procedure is described as followings:

1. Normalizing ARBs stimulation level and calculating the resources

   For each $ab \in AB$, normalize its stimulation according to equation (13)

$$ab.stim = \frac{ab.stim - \min stim}{\max stim - \min stim} \tag{13}$$

$$ab.resources = ab.stim * clonal\_rate \tag{14}$$

   The *resAlloc*, total resources allocated across AB population, is defined as:

$$resAlloc = \sum_{i=1}^{|AB|} ab_i \cdot resources \tag{15}$$

   If the *resAlloc* exceeds the allowed values (TotalNumRes), the weakest ARBs will be removed until the totality of allocated resources is under the limit.

2. Stopping criterion for the training procedure

   Calculate the average stimulation level. If the average stimulation value of all ARBs is less than a given stimulation threshold (*ST*), the process moves to step 3; otherwise, jump to step 4.

3. Clonal reproduction and ARBs mutation

   For each $ab \in AB$, it has the opportunity to produce mutated offspring. The number of clones is in proportion to its stimulation level, and is defined as:

$$\text{NClones}=\text{clonal\_rate} *\text{stimulation}(ags,ab) \tag{16}$$

   Each ab represents a neural network so that ARBs mutation refers to their architecture *l*, connection weights *W* and *V*. The mutation procedure and the corresponding function mutate (*ab*) are defined in figure 2.

In Fig 2, the function random() returns a random value within the range[0,1], random1() returns a random value within the range[-1,+1], random2() returns a random value within the range[$l_{\min}$,$l_{\max}$], and random3(n) returns a random value within the range[1,n]. The mutation operation uses the non-uniform operator, which is able to achieve fine tuning [13]. The mutation function $\Delta(t, y)$ is defined as followings:

$$\Delta(t, y) = y(1 - r^{(1-t/T)^{\lambda}} \tag{17}$$

where t is the iteration number, T is the maximum of iteration number, r is a random value in range[0,1], λ is a parameter to decide the nonconforming degree.

```
mutate(ab)
{
    for each(ab.w, ab.v)
    do
        a= $\min W_{ji}$    b= $\max W_{ji}$
        c= $\min V_{kj}$    d= $\max V_{kj}$
        e=flase    f=random()    g=random1()
        j=random2()    i=random3($m$)    k=random3($q$)
        if (f<mutation_rate)
            e=true
            if(g $\geq$ 0)
                $W_{ji} = W_{ji} + \Delta(t, b - W_{ji})$
                $V_{kj} = V_{kj} + \Delta(t, d - V_{kj})$
            else

                $W_{ji} = W_{ji} - \Delta(t, W_{ji} - a)$
                $V_{kj} = V_{kj} - \Delta(t, V_{kj} - c)$
    Done
    If(e $\equiv$ true)
      f=random()
      If(f<mutation_rate)
      ab.$l$=random2()
    return ab
}
```

Fig. 2. Mutation function

After mutation procedure, the architecture l may be changed. Suppose the architecture of ab_i, l_i changed to l_j, the three phases are described as following:

1) $l_i = l_j$, the architecture of ab_i not be changed;
2) $l_i > l_j$, the weights W and V items connect with the nodes of $l_i > l_j$ must be deleted.
3) $l_i < l_j$,then some weights W and V items connect with the nodes of $l_j > l_i$ must be joined, their values can be randomly chosen from range[-1,+1].

4. Selection of candidate memory cells

The highest stimulation ARB with the same architecture (l) from last AB set is selected as a candidate memory cell of the architecture ($mc_{candidate}$). All kinds of $mc_{candidate}$ constitute a set of candidate memory cells.

2.2.4 Promoting Candidate Cells to Memory Cell Pool

Comparing $mc_{candidate}$ with mc of the same architecture in MC set, if the stimulation of $mc_{candidate}$ is more than the stimulation of mc, the $mc_{candidate}$ will replace the mc, into memory cell pool.

Once the candidate memory cells have been evaluated and added into the set of memory cells, one training cycle is completed. The next training process proceeds from step 2.2.2 to step 2.2.4. This process continues until the average stimulation value of all ARBs is higher than a given stimulation threshold ST or reaches the maximum of iteration number T.

2.2.5 Continue BP Training and Classification

Continue BP training based on the mc_{match} in order to find the optimal connection weights. Then, the testing data can be classified by the optimal neural networks.

3 Experiments and Discussion

3.1 Dataset

Vowel data are come from MIT Lincoln Laboratory [14]. The data has two dimensional features (frequencies). The total vowel samples 383 are divided into the 10 English classes.

The SPOT image of Calcutta in India used for classification is come from literature [2]. The set has three bands, i.e. green band (0.50—0.59μm), red band (0.61—0.68μm) and infrared band (0.79—0.89μm), and comprises 932 points belonging to seven classes: turbid water (TW), pond water (PW), concrete (concr), vegetation (Veg), habitation (Hab), open space (OS), and roads (including bridges) (B/R).

Iris data are come from personal website of A. Watkins. The set, in which total 150 samples are divided into three classes, has four dimensional features,.

There are five types of supervised classifiers: Bayes classifier, k-NN classifier, AIRS classifier, BP-MLP classifier and RLIA-BP classifier.

In experiments, the 30% data are randomly chosen from total samples for training, and 70% for testing. For any given run, the results are an average of three runs on each of the 30% training–70% test set combinations.

3.2 Implementation Parameters of Classifiers

In Bayes algorithm, assuming the conditional densities are the normal densities, and the prior probabilities can be got from training patterns. In k-NN algorithm, *k*=3~7 for different data. In AIRS algorithm, hyper_colnal_rate=2.0, clonal_rate=10, mutation_rate=0.1, *ST*=0.9, TotalNumRes=200, ATS=0.1~0.3 and *k*=3~7 (depended on classifying data). In BP-MLP algorithm, the learning rate is 0.02, the learning rate increment is set to 1.001 and the momentum rate is 0.9. The algorithm ends if performance (*MSE*) is less than 0.012. Otherwise, the algorithm is executed for 2000 epochs.

In the RLIA-BP algorithm, hyper_clonal_rate=4.0, clonal_rate=10, mutation_rate=0.1, *ST*=0.90~0.95(depended on classifying data), TotalNumRes=100, the maximum of iteration number T=1200. BP parameters are set to the same as BP-MLP.

3.3 Performance Comparison of Classifiers

Table 1, 2 and 3 are the performance comparison of classifiers for vowel, SPOT and Iris data respectively.

As seen from Table 1, 2 and 3, for vowel and SPOT data, the overall classification accuracy of the RLIA-BP classifier is better than other classifiers, about 1.5%~ 7%. For Iris dataset, the performance of RLIA-BP classifier is the same as BP-MLP and Bayes, but better than k-NN and AIRS.

Table 1. The performance comparison of classifiers for vowel data

Accuracy	Classifier				
	Bayes	k-NN	AIRS	BP-MLP	RLIA-BP
Overall accuracy (%)	68.49	63.43	65.55	71.01	73.53
Kappa coefficient	0.634	0.598	0.619	0.678	0.708

Table 2. The performance comparison of classifiers for SPOT data

Accuracy	Classifier				
	Bayes	k-NN	AIRS	BP-MLP	RLIA-BP
Overall accuracy (%)	80.37	83.74	82.67	86.04	87.58
Kappa coefficient	0.656	0.734	0.730	0.755	0.785

Table 3. The performance comparison of classifiers for Iris data

Accuracy	Classifier				
	Bayes	k-NN	AIRS	BP-MLP	RLIA-BP
Overall accuracy (%)	97.14	91.43	93.33	97.14	97.14
Kappa coefficient	0.957	0.888	0.900	0.958	0.958

4 Conclusions

In this paper, we have described an evolving neural network classifier based on resource limited immune approach (RLIA) in detail. RLIA made it feasible to automatically evolve the appropriate architecture of neural networks and to globally find near-optimal connection weights. Then, with BP algorithm, the optimal connection weights can be found. The RLIA-BP classifier, which is derived from hybrid algorithm mentioned above, is demonstrated on vowel data, SPOT multi-spectral image data and Iris data effectively. The simulation results show that RLIA-BP classifier processes better performance than other classifiers, such as Bayes classifier, k-NN classifier, AIRS classifier, and BP-MLP classifier.

The feature of the proposed algorithm is that the neural network structure is evolved automatically while connection weights are being evolved.

RLIA , which does not use crossover operator that would destroy the structure of neural network in EANN based on GA [4], has a chance to increase the population diversity, expand the searching space and promote the probability of getting optimal individual through clone, mutation, resources competition and immune memory mechanism.

Acknowledgments. The authors acknowledge the help of Ph.D. Bandyopahyay (Machine Intelligence Unit, Indian Statistical Institute, India) for providing the SPOT image data, and also acknowledge Prof. Zengqi Sun (Tsinghua National Laboratory for Information Science and Technology, Beijing, China) for reviewing the paper and making many helpful suggestions.

References

1. Tou, T.T., Gonzalez, R.C.: Pattern Recognition Principles. Addison-Wesley, New York (1974)
2. Pal, S.K., Bandyopadhyay, S., Murthy, C.A.: Genetic Classifiers for Remotely Sensed Images: Comparison with Standard Methods. International Journal of Remote Sensing 2(13), 2545–2569 (2001)
3. Lippmann, P.P.: Pattern Classification Using Neural networks. IEEE Communications Magazine, 47–63 (November 1989)
4. Yao, X.: Evolving artificial neural networks. Proceeding of the IEEE 87(9), 1423–1447 (1999)
5. Watkins, A., Boggess, L.: A resource limited artificial immune classifier. In: Proceeding of the 2002 Congress on Evolutionary Computation (CEC 2002), Special Session on Artificial Immune System, vol. 1, pp. 926–931. IEEE Press, CA (2002)
6. Fu, X., Dale, P.E.R., Zhang, S.: Evolving Neural Network Using Variable String Genetic Algorithm for Color Infrared Aerial Image Classification. Chin. Geogra. Sci. 2008 18(2), 162–170 (2008)
7. Hertz, J., Krogh, A., Palmer, R.: An Introduction to the Theory of Neural Computation. Addison Wesley Publ. Comp., Redwood City (1991)
8. Holland, J.H.: Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor (1975)

9. Jiao, L.C., Du, H.F., Liu, F., Gong, M.G.: Immune Optimal Computation, Learning and Recognition. Science Press, Peking (2006) ISBN 978-7-03-017006-4
10. Timmis, J., Neal, M., Hunt, J.: An Artificial Immune System for Data Analysis. Biosystem 55(1/3), 143–150 (2000)
11. 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)
12. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn., p. 311. John Wiley & Sons, New York (2001)
13. Michalewicz, Z.: Genetic Algorithms +Data Structure=Evolution programs. Springer, New York (1992)
14. Kukolich, L., Lippmann, R.: LNKnet User's Guide. Revision 4, MIT Lincoln Laboratory (February 2004), `http://www.ll.mit.edu/IST/lnknet`

Cryptanalysis of Four-Rounded DES Using Binary Artificial Immune System

Syed Ali Abbas Hamdani, Sarah Shafiq, and Farrukh Aslam Khan[*]

Department of Computer Science, FAST National University of Computer and Emerging Sciences, A.K. Brohi Road, H-11/4, Islamabad, Pakistan
ali_hamdani_1214@yahoo.com, iz_sarah@yahoo.com,
farrukh.aslam@nu.edu.pk

Abstract. In this paper, we present a new approach for the cryptanalysis of four-rounded Data Encryption Standard (DES) based on Artificial Immune System (AIS). The proposed algorithm is a combination of exploitation and exploration of fitness landscape where it performs local as well as global search. The algorithm has the property of automatically determining the population size and maintaining the local solutions in generations to generate results close to the global results. It is actually a known plaintext attack that aims at deducing optimum keys depending upon their fitness values. The set of deduced or optimum keys is scanned to extract the valuable bits out by counting all bits from the deduced key set. These valuable extracted bits produce a major divergence from other observed bits. This results in a 56-bit key deduction without probing the whole search space. To the best of our knowledge, the proposed algorithm is the first attempt to perform cryptanalysis of four-rounded DES using Artificial Immune System.

Keywords: Cryptanalysis, Four-rounded DES, Artificial Immune System (AIS), Fitness measure.

1 Introduction

Data Encryption Standard (DES) is an encryption algorithm which differs from the conventional cryptographic algorithms that deploy single encryption or decryption technique, whereas DES utilizes various techniques in series to make the information secure. DES is therefore identified as a product cipher among the cryptographic techniques. The major property of DES algorithm is that encryption and decryption are basically identical processes that are comprised of substitution and permutation functions. Only the order of application of these functions to encrypt data is reversed for data decryption.

Recent research done illustrates that most of the conventional ciphers can be broken by applying evolutionary techniques, but for the modern block ciphers, high-nonlinearity and low autocorrelation is often used in the cryptographic design which makes the cryptanalysis of modern block ciphers hard using these techniques. This

[*] Corresponding author.

paper presents a new approach for the cryptanalysis of four-rounded DES based on Artificial Immune System (AIS). First, we adopt a chosen plaintext attack for which an efficient fitness measure is deployed for the cryptanalysis. Secondly, a set of optimum keys is generated from a variety of plaintext/ciphertext pairs depending upon the fitness measure and then optimal keys with higher fitness values are extracted from them. Thirdly, some valuable bits in these optimum key set, which produce an evident divergence from the observed bits are deduced. Finally, these valuable bits are applied to find other bits without probing the whole key space. The fitness measure is flexible and can be deployed to break other Feistel block ciphers. The experimental results indicate that artificial immune system algorithm is successful at breaking the four-rounded DES.

The rest of the paper is organized as follows: The related work is presented in section 2. Section 3 explains the artificial immune system. Section 4 discusses the DES algorithm and problem formulation. Section 5 explains the proposed technique. Section 6 discusses the experiments and results, and section 7 concludes the paper.

2 Related Work

A lot of research has been done in the field of cryptanalysis. In 1993, Spillman et al. [6] presented a technique using genetic algorithms to attack substitution ciphers. Clark [7] investigated optimization algorithms for the purpose of cryptanalysis. Clark and Dawson [8] carried research on cipher cryptanalysis to the next level by using simulated annealing, tabu search and genetic algorithm [9]. Laskari et al. [10] in 2005, used particle swarm optimization for the cryptanalysis of SDES. A lot of research further investigated the use of evolutionary techniques to cryptanalyze various algorithms including DES [2, 3, 11, 12, 13, 14, 15, 16]. Matsui [17] introduced linear cryptanalysis methods to cryptanalyze DES. Shahzad et al. [18] used binary Particle Swarm Optimization (PSO) for the cryptanalysis of four-round DES. The features of DES that make its cryptanalysis cumbersome include autocorrelation and nonlinearity. These features make the relation between plaintext and ciphertext ambiguous [5]. From all the work done so far in this area, it is obvious that many evolutionary techniques have been deployed for the cryptanalysis of DES, but, to the best of our knowledge, artificial immune system algorithm has not been used by anyone for the cryptanalysis of four-rounded DES.

Optimized aiNet algorithm [1] was initially designed for data clustering and was then optimized for multimodal function optimization. In this paper, we use the optimized aiNet algorithm for the cryptanalysis of four-rounded DES block cipher. It basically uses a set of cells that represent the initial population. Each cell represents a 56-bit key. The fitness of each cell depends on known ciphertext and the ciphertext generated using the candidate key. A set of optimum keys is generated after several iterations and these optimum keys are then used to extract the original key.

3 The Artificial Immune System

Artificial immune systems have cells and use the generation mechanism by creating clones of the parent cell population [4]. Selection of the clones from the clone population and affinity maturation principles are used to describe the reaction of the immune system to pathogens and the enhancement in its capacity of detecting and eliminating pathogens. The clone selection process activates whenever a pathogen attacks the organism; many immune cells that identify these pathogens reproduce and some of these immune cells become effector cells and some become memory cells.

The effector cells emit many antibodies whereas the memory cells remain alive for a long time to act rapidly and more effectively in future attacks by similar pathogens. During the reproduction of the cells, they experience high rate somatic mutations and with a selection process, the cells with promising affinity in relation to the attacking pathogen distinguish into memory cells. This procedure of somatic mutation and selection is known as affinity maturation.

Both these processes; selection process of clones and maturation of affinity, are very much similar to each other with a few differences between these immune processes and the evolution of species. Within the immune system, somatic cells replicate without crossover of genetic material during mitosis, the mutation experienced by an immune cell is relative to its affinity with the selective pathogen. This means that with high affinity, the mutation rate gets smaller, and the number of offspring of each cell is also relative to its affinity with the selective pathogen i.e. with higher affinity, the number of offsprings also increases. Evolution in the immune system occurs within the organism and thus it can be viewed as a micro-evolutionary process.

4 Data Encryption Standard (DES) Algorithm

4.1 The Feistel Ciphers

A Feistel cipher is a block cipher but with a specific structure. In feistel ciphers, encrypt and decrypt operations are used which combine multiple rounds of iterative functions. Each round function normally includes bit-shuffling, non-linear functions, and XOR operation. The data encryption standard (DES) is also a feistel iterated block cipher. The DES algorithm takes a known plaintext *P* block and the output is the ciphertext *C* block, each of 64 bits. Through a 16-round process, the current 64-bit word is divided into two 32-bit parts, the left part L and the right part R and 16 rounds *i*. L and R can be computed as follows:

$$Li = Ri\text{-}1,$$

$$Ri = Li \oplus \mathrm{f}\,(\mathrm{R}i\text{-}1,\ Ki)$$

Each R block of 32-bit is then expanded to 48-bit block using a table and the expanded R_{n-1} is XORed with the key $K_{i.}$ This 48-bit XOR block with eight blocks of each 6 bits is then passed through S blocks to produce an output of eight blocks of 4 bits each. Where K_i is derived from the cipher key *K* through the key arrangement algorithm, K_i is the 48-bit subkey used in *i*th round, and f is a round function. The round function f is the major module of DES and consists of three operations: substitution, permutation and XOR operations. The S-boxes are nonlinear mappings and the only nonlinear part of DES. There are eight S-boxes each of which is a substitution mapping of 6 to 4 bits.

4.2 Formulating the Problem

The cryptanalysis using artificial immune system is basically the search of the optimum key in the search space. The difficulty of searching an optimum key depends upon the length of the key which is 56 bits long. A cell is used for the representation of a key, which consists of a binary bits string. A cell also includes the key fitness and its normalized fitness. The cell evolution process is same as the process of a key optimization. A bit difference taking place in the key should cause a drastic change in the cipher text. Accordingly, initial keys close to the real key, by applying genetic mutation operator, must not generate a better solution than the previous value.

The selection of a suitable fitness function is a main problem that requires careful decision. In general, it is feasible for classical ciphers that the design of the fitness measure is modeled on the linear estimated expression or the distributional properties of the keys. It is for this reason that the cryptanalysis done based on heuristic methods is difficult for block ciphers. However, selection of a suitable fitness function does not guarantee that the algorithm will result in deducing the real key exactly.

5 Cryptanalysis of Four-Rounded DES Using Artificial Immune System

5.1 The Initialization Process

In the initialization process, we define the parent population which is initialized on the basis of changes made in the key that we are deducing. The cells in the population are initialized by maintaining the bits that are fixed in the deduced key. It can speed up the evolution process so as to find the better optimum key as quickly as possible. The initialization process needs to maintain the diversity of the clone population.

5.2 Fitness Function

As a solution, we define the fitness function in a way that the function should measure the connection between initial plaintext/ciphertext and target ciphertext/plaintext pairs. We define the plaintext *M,* original ciphertext C_S using real key *K,* the encrypted ciphertext C_t using trial key *K',* and the fitness function *f:*

$$f = (S) / 64 \qquad (1)$$

Where S denotes the number of the same bits in identical positions between C_S and C_t. It is an approximate expression for cryptanalysis using artificial immune system. Many better keys, if adequate plaintext and ciphertext pairs are used, can be successfully obtained based on this fitness function in the experiments. Such search solutions have better situations as a whole. In some situations, the divergence is comparatively evident between the key optimization and the real key. This divergence can be established using other plaintext/ciphertext pairs or the fitness function. The parent population formation and the fitness of keys are shown in Fig. 1 and Fig. 2 respectively.

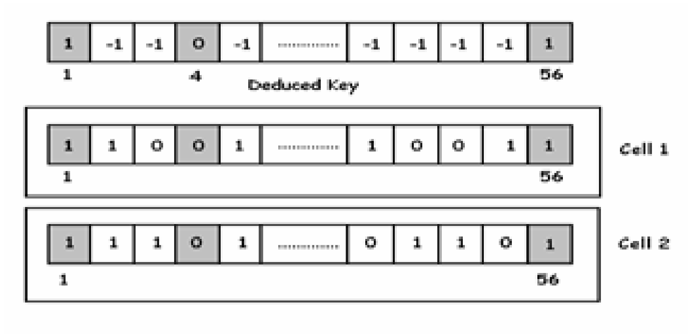


Fig. 1. Parent population formation using deduced key

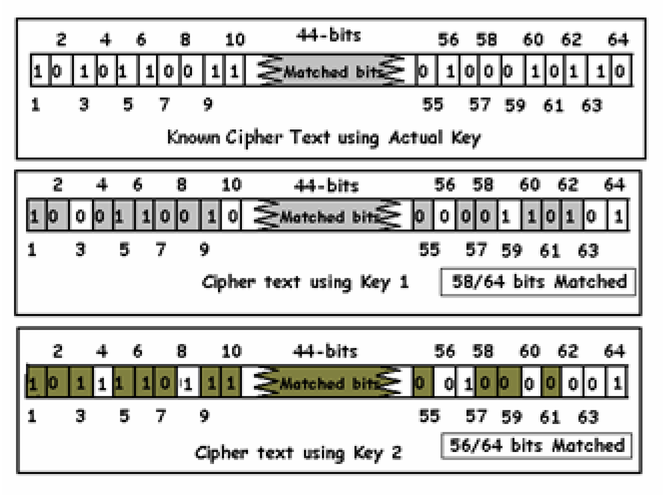


Fig. 2. Fitness of keys

5.3 The Evolution Process

Given an initial population of parent cells, each one with a fitness value, the algorithm progresses by selecting a cell from clone population having the highest fitness value. Average error will be calculated for the fitness of parents and clones unless the two populations are quite similar, the whole process continues. The two populations are then merged and similar cells are removed to overcome the redundancy problem. The next step is of search space exploration, which is done by adding a specific amount of new cells to the parent population. The parent population and the clone population are shown in Fig. 3 and Fig. 4 respectively.

5.4 Breeding Strategy

The proposed algorithm is shown below. The algorithm can be elaborated in simple steps including steps 1 to 5, where at each iteration; a population of cells is optimized locally through affinity comparative mutation. During steps 6 to 8, when this population reaches a steady state depending upon the average fitness, the cells interact with

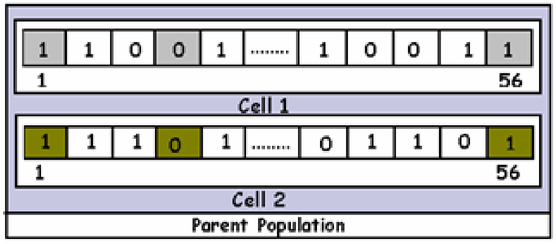


Fig. 3. Parent Population

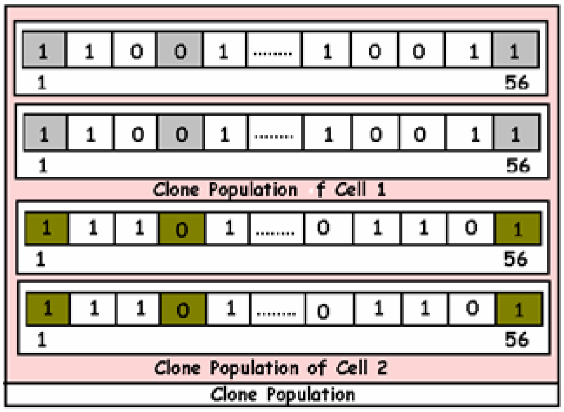


Fig. 4. Clone Population

Proposed Algorithm

```
A population of cells is randomly initialized.

While stopping criterion is not met
Do
    1. Determine the fitness of each cell and normalize the fitnesses.
    2. Generate a number of clones for each cell.
    3. Mutate each clone relative to the fitness of its parent cell, while keeping the original
       parent cell.
    4. Determine the fitness for all individuals of the population including parent as well as
       clone population.
    5. For each clone, select the cell with highest fitness and calculate the average fitness
       of the selected population.
    6. If the average error of the population is not drastically different from the previous
       iteration, then continue. Else, return to step 1.
    7. Determine the affinity of all cells in the population. Suppress all but the highest fitness
       of those cells whose affinities are less than the suppression threshold and determine the
       number of memory cells after suppression.
    8. Introduce a percentage of randomly generated cells and return to step 2.
End While
```

each other and some of the similar cells are eliminated from the population to avoid redundancy. Also, a number of randomly generated cells are added to the current population and the process of local optimization restarts. Whenever the algorithm is run, a key is found and passed to the next run of the algorithm and the new population is then initialized by maintaining the fixed bits of the deduced key of the last run. Optimum keys are selected from final solutions whose fitness value is the largest. For all optimum keys, we count the amount of 1 or 0 for each bit position separately, where each count is divided by total *Runs*. The sum of some bits with 1 or 0 divided by *Runs* has higher value than the threshold *alpha*. These bits can be deduced as 1 or 0. The process of mutation is shown in Fig. 5.

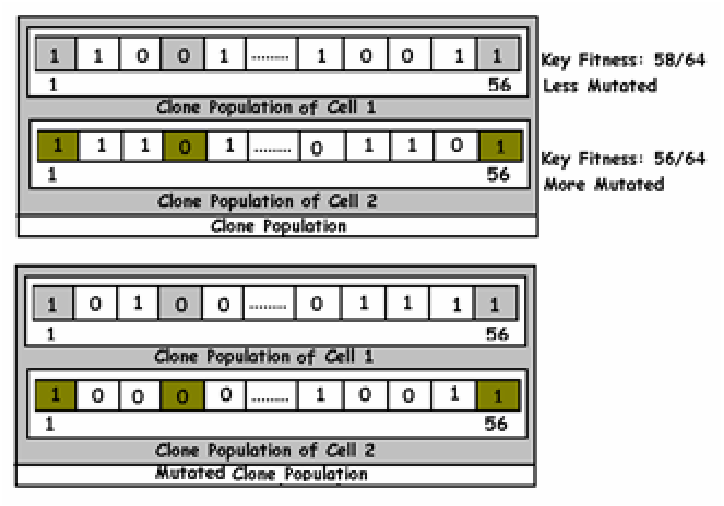


Fig. 5. The mutation process

5.5 Key Deduction Strategy

The key bits are deduced by comparing the optimum keys obtained from different runs. The simple method for deduction is, for example, K_1 = 1001011, K_2 = 1110010 and K_3 = 1101001 are the three keys from the set of optimum keys; *Runs* = 3. The sum of 1 for each bit position is 3, 2, 1, 2, 0, 2, 2 and the sum of 0 is 0, 1, 2, 1, 3, 1, 1. Accordingly, the sum divided by *Runs* is 1.0, 0.67, 0.33, 0.67, 0, 0.67, 0.67 and 0.0, 0.33, 0.67, 0.33, 1.0, 0.33, 0.33 respectively. If threshold is 0.8, the first value 1 and the fifth value 0 are higher than the threshold. Thus, the first bit is 1 and the fifth bit is 0 in the real key.

6 Experimental Setup and Results

The given table summarizes the performance results and parameters of cryptanalysis using artificial immune system for four-rounded DES. The fixed number of generations N is 1000, and the runs R is 500 times in our experiments. This algorithm is implemented in Visual C++ and run on Intel P-M processor.

Table 1. Experimental results based on the proposed technique

Round	**λ**	**α**	***R***	**Optimum rate**	**Success bits**
Four	0.75	0.60	500	1.0	19
Four	0.78	0.60	500	1.0	6
Four	0.80	0.70	500	1.0	11
Four	0.80	0.75	500	1.0	11

The parameters defined in our results include:
λ: Fitness threshold, that is, the set of candidate keys. Include only that key whose fitness value is greater than or equal to the specified threshold value. In the experiments, the threshold values are 0.75, 0.78 and 0.80 respectively.
α: The threshold used to set the values of the deduced key.
R: It is the number of runs after which the deduced key is obtained.
Optimum rate: It is the number of keys found during the run having fitness value greater than or equal to the defined fitness threshold.
Success bits: These are the number of bits matched in the deduced key and the original known key.

As Table 1 indicates, the algorithm would find four valuable bits at least for the four-rounded DES using the above algorithm. Subsequently, we could fix these four bits in the key and repeat the above steps until the whole key is found. In the above experiments, the algorithm runs four times and can find the 56-bit key.

7 Conclusion

We have presented a new approach for the cryptanalysis of four-rounded Data Encryption Standard (DES) based on Artificial Immune System (AIS). The algorithm has the property of automatically determining the population size and maintaining the local solutions in generations to generate results close to the global results. A known plaintext attack is used that aims at deducing the optimum keys depending upon their fitness values. The fitness function is an approximate expression for using AIS to cryptanalyze the DES algorithm. We can produce adequate optimum keys for all plaintext/ciphertext pairs based on this fitness function. However, we also observed that the design of actual fitness function should consider the following factors: Firstly, fitness function should have a better characteristic of global performance and large numbers of optimum keys can be found. Secondly, it is hard to create a high fitness value for each search. In some cases, the dissimilarity is comparatively evident between the current cell and the actual key. This can be solved by using various plaintext/ciphertext pairs in the experiments. Thirdly, the supporting fitness function can be used to improve the evaluation measure. This fitness function does not depend upon the cipher's underlying structure.

References

1. de Castro, L.N., Timmis, J.: An Artificial Immune Network for Multimodal Function Optimization. In: Proc. of Congress on Evolutionary Computation, CEC 2002 (2002)
2. Song, J., Zhang, H., Meng, Q., Wang, Z.: Cryptanalysis of Four-Round DES Based on Genetic Algorithm. In: International Conference on Wireless Communications, Networking and Mobile Computing (WiCom 2007), Shanghai, China, pp. 2326–2329 (2007)
3. Song, J., Zhang, H., Meng, Q., Wang, Z.: Cryptanalysis of Two-Round DES Using Genetic Algorithms. In: Kang, L., Liu, Y., Zeng, S. (eds.) ISICA 2007. LNCS, vol. 4683, pp. 583–590. Springer, Heidelberg (2007)
4. Dasgupta, D.: Artificial Immune Systems and Their Applications. Springer, Heidelberg (1999)
5. Coppersmith, D.: The data encryption standard (DES) and its strength against attacks. IBM Journal of Research and Development 38(3), 243–250 (1994)
6. Spillman, R., Janssen, M., Nelson, B., Kepner, M.: Use of A Genetic Algorithm in the Cryptanalysis of simple substitution Ciphers. Cryptologia XVII(1), 187–201 (1993)
7. Clark, A.: Modern Optimisation Algorithms for Cryptanalysis, pp. 258–262. IEEE, Los Alamitos (1994)
8. Clark, A., Dawson, E.: Optimisation Heuristics for the Automated Cryptanalysis of Classical Ciphers. J. Combinatorial Mathematics and Combinatorial Computing 28, 63–86 (1998)
9. Clark, A.J.: Optimization Heuristics for Cryptology, PhD thesis, Queensland University of Technology (1998)
10. Laskari, E.C., Meletiouc, G.C., Stamatioud, Y.C., Vrahatis, M.N.: Evolutionary computation based cryptanalysis: A first study, pp. 823–830. Elsevier, Amsterdam (2005)
11. Hernández, J.C., et al.: Easing collision finding in cryptographic primitives with genetic algorithms. In: Proc. of CEC 2002, Honolulu, HI, USA, vol. 1, pp. 535–539 (2002)
12. Russell, M., Clark, J.A., Stepney, S.: Using Ants to Attack a Classical Cipher. 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. 146–147. Springer, Heidelberg (2003)
13. Morelli, R., Walde, R., Servos, W.: A Study of Heuristic Approaches for Breaking short Cryptograms. International Journal on Artificial Intelligence Tools 13(1), 45–64 (2004)
14. Bafghi, A.G., Sadeghiyan, B.: Finding Suitable Differential Characteristics for Block Ciphers with Ant Colony Technique. In: Proc. of Ninth International Symposium on Computers and Communications (ISCC 2004), Washington, DC, USA, vol. 2, pp. 418–423 (2004)
15. Clark, J.A., Jacob, J.L., Stepney, S.: The Design of S-Boxes by Simulated Annealing. New Generation Computing 23(3), 219–231 (2005)
16. Castro, J.C.H., Sierra, J.M., Isasi, P., Ribagorda, A.: Genetic Cryptoanalysis of Two Rounds TEA. In: Sloot, P.M.A., Tan, C.J.K., Dongarra, J., Hoekstra, A.G. (eds.) ICCS-ComputSci 2002. LNCS, vol. 2331, pp. 1024–1031. Springer, Heidelberg (2002)
17. Matsui, M.: The First Experimental Cryptanalysis of the Data Encryption Standard. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 1–11. Springer, Heidelberg (1994)
18. Shahzad, W., Siddiqui, A.B., Khan, F.A.: Cryptanalysis of Four-Rounded DES using Binary Particle Swarm Optimization. In: ACM GECCO 2009, Montréal, Québec, Canada, pp. 2161–2166 (2009)

An Immune Concentration Based Virus Detection Approach Using Particle Swarm Optimization

Wei Wang[1,2], Pengtao Zhang[1,2], and Ying Tan[1,2]

[1] Key Laboratory of Machine Perception, Ministry of Eduction, Peking University
[2] Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, 100871, P.R. China
ytan@pku.edu.cn

Abstract. This paper proposes an immune concentration based virus detection approach which utilizes a two-element concentration vector to construct the feature. In this approach, 'self' and 'nonself' concentrations are extracted through 'self' and 'nonself' detector libraries, respectively, to form a vector with two elements of concentrations for characterizing the program efficiently and fast. Several classifiers including k-nearest neighbor (KNN), RBF neural network and support vector machine (SVM) with this vector as input are then employed to classify the programs. The selection of detector library determinant and parameters associated with a certain classifier is here considered as an optimization problem aiming at maximizing the accuracy of classification. A clonal particle swarm optimization (CPSO) algorithm is used for this purpose. Experimental results demonstrate that the proposed approach not only has a very much fast speed but also gives around 98% of accuracy under optimum conditions.

Keywords: Immune Concentration, Clonal Particle Swarm Optimization, Virus Detection.

1 Introduction

Computer virus has been considered as an increasingly serious problem while evolving with the rapid development of computer environments, such as operating system, network, etc. There are two main virus detection methods: signature-based method and malicious activity detection[1]. With novel advanced technologies being widely used in manufacturing new viruses, polymorphic viruses can change their signatures while spreading. Heuristic methods which is more sophisticated, like malicious activity detection, can be used to identity unknown viruses. Among those heuristic methods, AIS as a dynamic, adaptive and distributed learning system, protects benign files against virus invasion by distinguishing 'nonself' from 'self'[2]. It is often combined with traditional classification algorithms to construct a virus detection system, including Naïve Bayes, Support Vector Machine (SVM), Artificial Neural Network (ANN), k-nearest neighbor (KNN), and hybrid approaches[3][4].

In this paper, a AIS method is used to generate a two-element concentration vector as the feature vector for virus detection. 'Self' and 'nonself' detector libraries contain the bit strings which are utmost representative of benign and virus programs, respectively. Document frequency and information gain of a fixed length fragment are used to decide whether the fragment can be taken as a detector. ‘Self’ and ‘nonself’ concentrations are constructed by using ‘self’ and ‘nonself’ detectors in the libraries to traverse a program, respectively. Unlike the traditional anti-virus methods, our proposed model helps reduce the feature dimensionality. The feature with these two elements is the input of a classifier, and one binary value is the output. As a result, the detector library determinant and parameters associated with the classifier become the vector that we need to optimize by using a clonal particle swarm optimization (CPSO) algorithm[5]. The optimal vector is the one whose cost function associated with classification is minimum, namely the one make the accuracy of classification maximum.

Comprehensive experiments are conducted on a public virus data set in the pervious works[6][7]. 10-fold cross validation is used to measure the performance. Comparisons on performance are also made among different classifiers including k-nearest neighbor (KNN), RBF neural network and support vector machine (SVM). Experimental results shows that the proposed approach achieves more than 97% detection rate by just using a two-element vector, so outperforms current approach.

The paper is organized as follows. In Section II, algorithmic implementations of our proposed approach are elaborated in detail with CPSO-based optimization process for detector library determinant and parameters associated with a certain classifier. Several experimental results are reported in Section III.

2 Immune Concentration Based Virus Detection Approach

2.1 Overview of Our Proposed Approach

Our proposed approach is mainly divided into three parts. (1)Generate ‘self’ and ‘nonself’ detector libraries. (2)Extract the two-element concentration vector of each training sample as the input of a classifier and use CPSO to optimize detector library and parameters associated with the classifier. (3)Several trained classifiers including KNN, RBF neural network and SVM are used to detect the testing samples characterized by optimized concentration vectors. The outline of our proposed approach is described in algorithm 1 and each step in detail is discussed in the following sections.

2.2 Generation of Detector Libraries

The operating principle of generating ‘self’ detector library and ‘nonself’ detector library is shown in algorithm 2.

One meaningful computer instruction is 8 or 16 bits normally. Here a fixed length L-bit fragment of binary data which is considered containing appropriate information of functional behaviors is taken as the detector to discriminate virus from benign program. The length L is set not too short to discriminate ‘self’ and ‘nonself’ and not too long to make virus-special data hidden in the binary data of files. Intuitively, the fragment that appears most in virus programs while seldom in benign programs is a good representative of a virus. Consequently, a sliding window (size: L bits, the overlap: [L/2] bits)

Algorithm 1. Algorithm for Virus Detection

```
Generate 'self' and 'nonself' detector libraries from training set
The sizes of the libraries are decided by parameter $m$ which corresponds to proportional selection of the candidate detectors

for each the sample in training set do
    Extract the two-element concentration vector of each training sample through the two detector libraries as the input of a classifier
end for

Use these feature vectors to train a certain classifier
Use CPSO to optimize detector library determinant $m$ and classifier-related parameters

while Algorithm is running do
    if a program is detected then
        Characterize the sample by concentration vector through trained 'self' and 'nonself' detector libraries
        Use trained classifier to predict the label of the program
    end if
end while
```

is used to count the fragment's document frequency in the virus and benign programs, which can reflect its tendency to be a virus or a benign file. In this paper, L is set to 32.

After counting the document frequency of each fragment, the tendency $T(X)$ of fragment X is defined in formula 1.

$$T(X) = P(X = 1|C_v) - P(X = 1|C_s) \tag{1}$$

$P(X = 1|C_v)$means document frequency of fragment X appears in virus samples of training set;
$P(X = 1|C_s)$means document frequency of fragment X appears in benign samples of training set.

If each fragment is extracted to form a dictionary, the size of this dictionary would be very large. The features that appears in most of files are not relevant to separate these files because all the classes have instances that contain these features. So with the number growth of features, the cost of computing would be increasing but the accuracy may not be improved and even worse. We reduces the number of fragments to generate 'self' and 'nonself' detector libraries based on information gain ratio of each detectors to make the detectors more representative. The process is described in Algorithm 2, in which $m$is a parameters to be adjusted, means proportional selection of all the fragments. The information gain is defined in formula 2.

$$IG(X,C) = \sum_{x\in\{0,1\},c\in\{C_v,C_s\}} P(X = x \wedge C = c) \cdot \log_2 \frac{P(X = x \wedge C = c)}{P(X = x) \cdot P(C = c)} \tag{2}$$

$P(X = 0|C_v)$means document frequency of fragment X which doesn't appear in virus samples of training set;

$P(X = 0|C_s)$means document frequency of fragment X which doesn't appear in benign samples of training set.

Algorithm 2. Algorithm for Generation of Detector Libraries

```
Initialize *'self'* and *'nonself' detector libraries* as $\varnothing$

**while** Algorithm is running **do**
    **for** each *fragment X* in the sample of training set **do**
        Calculate the *tendency* of *fragment X* by formula 1
        Calculate the *information gain* of *fragment X* by formula 2
    **end for**

    **for** each *fragment X* in the sample of training set **do**
        **if** $IG(X) > m$ **then**
            **if** $T(X) < 0$ **then**
                add *fragment X* into *'self' detector library*
            **else**
                add *fragment X* into *'nonself' detector library*
            **end if**
        **end if**
    **end for**
**end while**

Extract $P_s\%$ of fragments to form *'self' detector library* and $P_n\%$ of fragments to form *'non-
self' detector library*, $P_s, P_n$ are decided by parameter $m$
```

2.3 Construction of Feature Vector

For constructing a feature vector, a sliding window with [L/2] bits overlap is used to get the 'self' concentration and 'nonself' concentration of a program i, which are defined in formula 3,4, respectively.

$$BC_i = \frac{BN_i}{N_i} \tag{3}$$

$$VC_i = \frac{VN_i}{N_i} \tag{4}$$

Where BC_i and VC_i denotes the 'self' and 'nonself' concentration, respectively; BN_i is the number of detectors appearing in both program i and 'self' detector library; VN_i is the number of detectors appearing in both program i and 'nonself' detector library; N_i denotes the number of different L-bit fragments in the program.

2.4 Classification Parameters Selection

The feature constructed by two-element concentrations is the input of a classifier, and one binary value is the output. The generation of 'self' and 'nonself' detector libraries, which in turn determine the two-element concentration vector uniquely, here is an optimization problem. The optimal vector is the one make the accuracy of classification maximum.

Algorithm 3. Algorithm for Feature Construction

```
For a program to be detected, calculate the number $N_i$ of different L-bit fragments using a
sliding window with [L/2] bits overlap

Initialize $BN_i = 0, VN_i = 0$

for each different L-bit fragment in the program do
    if it appears in 'self' detector library then
        $BN_i$++;
    else if it appears in 'nonself' detector library then
        $VN_i$++;
    end if
end for

self-concentration=$BN_i/N_i$
nonself-concentration=$VN_i/N_i$
```

The vector that we need to optimize $P^* = \{P_s^*, P_n^*, P_1^*, P_2^*, \cdots, P_m^*\}$ is composed of detector library determinant m, and parameters $P_1^*, P_2^*, \cdots, P_m^*$ associated with a certain classifier. When m is set to different values, P_s^*, P_n^* would take different values, different detector libraries are obtained. So for a program to be characterized, a 'self' concentration which represents its similarity to benign program and a 'nonself' concentration which represents its similarity to virus can be constructed. $P_1^*, P_2^*, \cdots, P_m^*$ are classifier-related parameters which influence the performance of a certain classifier. Different classifiers hold different parameters and lead to different performance. For examples, parameters associated with KNN include number of nearest neighbors and the ways of distance measures. SVM-related parameters that determine the position of optimal hyperplane in feature space, include cost parameter C and kernel parameters. Finally, the trained classifiers are used to classify detecting samples.

The optimal vector is the one whose cost function associated with classification is minimum, namely the one make the accuracy of classification maximum. The cost function $CF(P)$ can be defined as

$$CF(P) = Err(P) \tag{5}$$

where $Err(P)$ is the classification error measured by 10-fold cross validation on the training set.

Input vector include two parts: detector library determinant m, and $P_1^*, P_2^*, \cdots, P_m^*$ these classifier-related parameters. Output is to find a P^*, so that

$$CF(P^*) = Err(P^*) = \min_{\{m, P_1^*, P_2^*, \cdots, P_m^*\}} Err(P) \tag{6}$$

3 Experimental Results

3.1 Data Set

Experiments are conducted on a public virus data set in the pervious works[6][7]. "Cilpku08" data set, which can get from http://www.cil.pku.edu.cn/malware, includes

3547 viruses classified to 685 families based on their properties. we select 915 benign files and 900 virus to test our method. 1815 programs are divided into ten partitions with approximately equal numbers of virus and benign programs. 10-fold cross validation is used to measure the performance.

3.2 Experiments on Different Concentrations

Here, m is a parameters to be adjusted, which means proportional selection of all the fragments. Different 'self' and 'nonself' concentrations correspond to $P_s\%$ of fragments extracted to form 'self' library and $P_n\%$ of fragments extracted to form 'nonself' library. P_s, P_n are decided by parameter m. A linear SVM with default parameters is used as the classifier. m ranges from 10% to 100% with a step 10%. Figure 1 shows the accuracy with different m value.

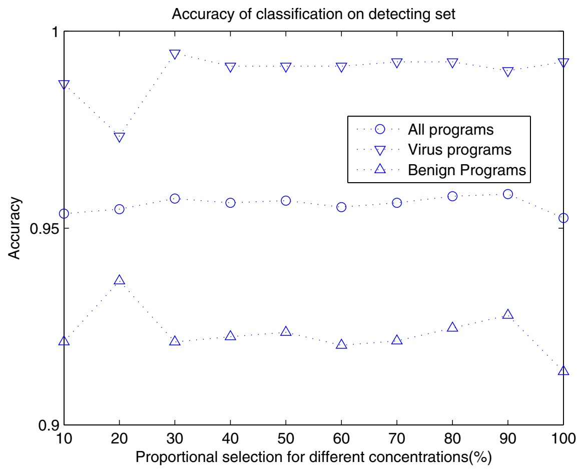


Fig. 1. Accuracy with different concentrations on detecting set by SVM

It is easy to see that different *m* gets almost the same result. In order to get a small detector library, we set *m* as 10%.

3.3 Classification Parameter Optimization

The selection of detector library determinant m and the classifier-related parameters, $P_1^*, P_2^*, \cdots, P_m^*$, is a dynamic optimization process.

Parameters associated with KNN include number of nearest neighbors K and the ways of distance measures, K is optimized in the integer number interval [1, 20], the ways of distance measures are chosen among $euclidean$, $cityblock$, $cosine$, $correlation$. For RBF neural network, the spread is optimized in real number interval [1, 5]. For SVM with linear kernel the cost parameter C is optimized in real number interval [1, 200]. For SVM with RBF kernel the cost parameter C and γ are optimized in real number interval

Table 1. Average detection rates on the detecting set under optimum conditions

	Empirical classification designs			Optimized classification designs		
Methods	$All(\%)$	$Virus(\%)$	$Benign(\%)$	$All(\%)$	$Virus(\%)$	$Benign(\%)$
KNN	95.76	94.00	97.50	98.90	98.78	99.02
RBF NN	97.57	97.56	97.59	97.96	97.78	98.14
SVM(Linear)	95.86	99.00	92.79	96.92	99.22	94.66
SVM(RBF)	95.81	99.44	92.22	98.62	98.00	99.24

$[1, 200]$ and $[1, 20]$, respectively. m is optimized in the integer number interval [5, 100]. The maximum number of generations is set to be 200 as the stop criterion, the number of particles in a swarm is 20.

The randomness of CPSO leads to the performance and obtained parameters vary slightly, therefore the average results of ten independent classes of experiments are used to evaluate tests, which is more reasonable. The average performances of empirical and optimized classification designs are reported in Table 1.

The results show that the optimized classification design resulted in a 2% increase in detection rate compared with the empirical classification design in average. The CPSO-based method has got an obvious advantage.

4 Conclusions

The immune concentration based virus detection approach proposed in this paper took a two-element concentration vector as the virus feature and employed several classical classifiers to detect virus. When parameters were optimized, the accuracy on detecting set reached 98%.

Different from traditional binary data mining methods, our method established an uniform framework for a general and systematical approach of feature construction and reduced the dimensionality to make the training and detecting faster. Also, the proposed feature extraction optimization approach attained better comparable results than the approach with empirical tentative parameters setting. The new method is easier without sacrificing performance.

Acknowledgement

This work was supported by the National High Technology Research and Development Program of China (863 Program), with grant number 2007AAOIZ453, and partially supported by National Natural Science Foundation of China (NSFC), under grant number 60673020 and 60875080. This work was also in part financially supported by the Research Fund for the Doctoral Program of Higher Education (RFDP) in China.

References

1. Henchiri, O., Japkowicz, N., Nathalie, J.: A feature selection and evaluation scheme for computer virus detection. In: Sixth International Conference on Data Mining, pp. 891–895 (2006)

2. Kephart, J.O.: A biologically inspired immune system for computers. In: Artificial Life IV, Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems, pp. 130–139 (1994)
3. Schultz, M.G., Eskin, E., Zadok, F., Stolfo, S.J.: Data mining methods for detection of new malicious executables. Security and Privacy, 38–49 (2001)
4. Wang, J., Deng, P.S., Fan, Y., et al.: Virus detection using data mining techniques. In: IEEE 37th Annual 2003 International Carnahan Conference on Security Technology, pp. 71–76 (2003)
5. Tan, Y., Xiao, Z.: Clonal particle swarm optimization and its applications. In: IEEE Congress on Evolutionary Computation, pp. 2303–2309 (2007)
6. Wang, W., Zhang, P.T., Tan, Y., He, X.G.: A hierarchical artificial immune model for virus detection. In: 2009 International Conference on Computational Intelligence and Security, pp. 1–5 (2009)
7. Chao, R., Tan, Y.: A virus detection system based on artificial immune system. In: 2009 International Conference on Computational Intelligence and Security, pp. 6–10 (2009)
8. Kerchen, P., Lo, R., Crossley, J., et al.: Static analysis virus detection tools for unix systems. In: 13th National Computer Security, pp. 4–9 (1990)
9. Hofmeyr, S.A., Forrest, S., Somayaji, A.: Intrusion detection using sequences of system calls. Journal of Computer Security, 151–180 (1998)
10. Tan, Y., Deng, C., Ruan, G.C.: Concentration based feature construction approach for spam detection. In: International Joint Conference on Neural Networks, pp. 3088–3093 (2009)
11. Deng, P.S., Wang, J., Shieh, W., et al.: Intelligent automatic malicious code signatures extraction. In: IEEE 37th Annual 2003 International Carnahan Conference on Security Technology, pp. 600–603 (2003)
12. Preda, M.D., Christodorescu, M., Jha, S., et al.: A semantics-based approach to malware detection. In: 34th Annual Symposium on Principles of Programming Languages, vol. 42(1), pp. 377–388 (2007)
13. Ruan, G.C., Tan, Y.: A three-layer back-propagation neural network for spam detection using artificial immune concentration. Soft Computing 14, 139–150 (2010)
14. Moskovitch, R., Stopel, D., Feher, C., et al.: Unknown malcode detection via text categorization and the imbalance problem. In: IEEE International Conference on Intelligence and Security Informatics, pp. 156–161 (2008)
15. Drucker, H., Wu, D., Vapnik, V.N.: Support vector machines for spam categorization. IEEE Transactions on Neural Networks 10, 1048–1054 (1999)

Fireworks Algorithm for Optimization

Ying Tan and Yuanchun Zhu

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,ychzhu}@pku.edu.cn

Abstract. Inspired by observing fireworks explosion, a novel swarm intelligence algorithm, called Fireworks Algorithm (FA), is proposed for global optimization of complex functions. In the proposed FA, two types of explosion (search) processes are employed, and the mechanisms for keeping diversity of sparks are also well designed. In order to demonstrate the validation of the FA, a number of experiments were conducted on nine benchmark test functions to compare the FA with two variants of particle swarm optimization (PSO) algorithms, namely Standard PSO and Clonal PSO. It turns out from the results that the proposed FA clearly outperforms the two variants of the PSOs in both convergence speed and global solution accuracy.

Keywords: natural computing, swarm intelligence, fireworks algorithm, particle swarm optimization, function optimization.

1 Introduction

In recent years, Swarm Intelligence (SI) has become popular among researchers working on optimization problems all over the world [1,2]. SI algorithms, e.g. Particle Swarm Optimization (PSO) [3], Ant System [4], Clonal Selection Algorithm [5], and Swarm Robots [6], etc., have advantages in solving many optimization problems. Among all the SI algorithms, PSO is one of the most popular algorithm for searching optimal locations in a D-dimensional space. In 1995, Kennedy and Eberhart proposed PSO as a powerful global optimization algorithm inspired by the behavior of bird blocks [3]. Since then, the PSO has attracted the attentions of researchers around the globe, and a number of variants of PSO have been continually proposed [7,8].

Like PSO, most of swarm intelligence algorithms are inspired by some intelligent colony behaviors in nature. In this paper, inspired by the emergent swarm behavior of fireworks, a novel swarm intelligence algorithm called Fireworks Algorithm (FA) is proposed for function optimization. The FA is presented and implemented by simulating the explosion process of fireworks. In the FA, two explosion (search) processes are employed and mechanisms for keeping diversity of sparks are also well designed. To validate the performance of the proposed FA, comparison experiments were conducted on nine benchmark test functions among the FA, the Standard PSO (SPSO), and the Clonal PSO (CPSO) [8]. It is shown that the FA clearly outperforms the SPSO and the CPSO in both optimization accuracy and convergence speed.

The remainder of this paper is organized as follows. Section 2 describes the framework of the FA and introduces two types of search processes and mechanisms for keeping diversity. In Section 3, experimental results are presented to validate the performance of the FA. Section 4 concludes the paper.

2 Fireworks Algorithm

2.1 FA Framework

When a firework is set off, a shower of sparks will fill the local space around the firework. In our opinion, the explosion process of a firework can be viewed as a search in the local space around a specific point where the firework is set off through the sparks generated in the explosion. When we are asked to find a point x_j satisfying $f(x_j) = y$, we can continually set off 'fireworks' in potential space until one 'spark' targets or is fairly near the point x_j. Mimicking the process of setting off fireworks, a rough framework of the FA is depicted in Fig. 1.

In the FA, for each generation of explosion, we first select n locations, where n fireworks are set off. Then after explosion, the locations of sparks are obtained and evaluated. When the optimal location is found, the algorithm stops. Otherwise, n other locations are selected from the current sparks and fireworks for the next generation of explosion.

From Fig. 1, it can be seen that the success of the FA lies in a good design of the explosion process and a proper method for selecting locations, which are respectively elaborated in subsection 2.2 and subsection 2.3.

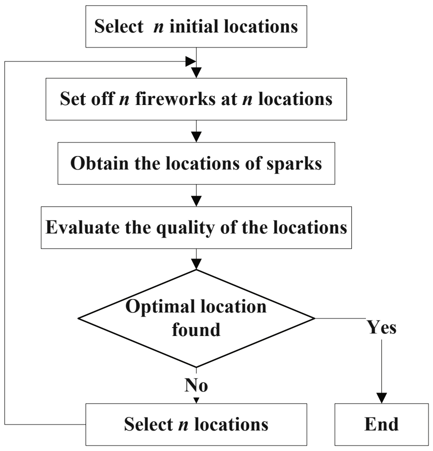

Select *n* initial locations
Set off *n* fireworks at *n* locations
Obtain the locations of sparks
Evaluate the quality of the locations
Optimal location found
Yes
No
Select *n* locations
End

Fig. 1. Framework of fireworks algorithm

2.2 Design of Fireworks Explosion

Through observing fireworks display, we have found two specific behavior of fireworks explosion. When fireworks are well manufactured, numerous sparks are generated, and the sparks centralize the explosion center. In this case, we enjoy the spectacular display of the fireworks. However, for a bad firework explosion, quite few sparks are generated, and the sparks scatter in the space.

The two manners are depicted in Fig. 2. From the standpoint of a search algorithm, a good firework denotes that the firework locates in a promising area which may be close to the optimal location. Thus, it is proper to utilize more sparks to search the local area around the firework. In the contrast, a bad firework means the optimal location may be far from where the firework locates. Then, the search radius should be larger. In the FA, more sparks are generated and the explosion amplitude is smaller for a good firework, compared to a bad one.

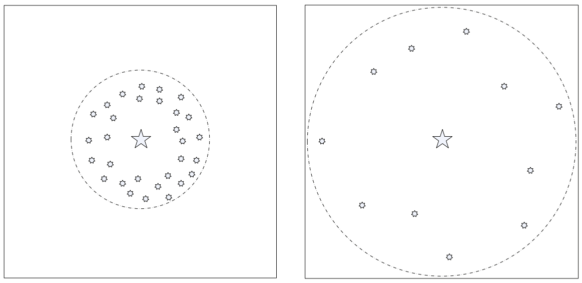

(a) Good explosion (b) Bad explosion

Fig. 2. Two types of fireworks explosion

Number of Sparks. Suppose the FA is designed for the general optimization problem:

$$\text{Minimize } f(\boldsymbol{x}) \in \mathbb{R},\ \boldsymbol{x}_{\min} \leqslant \boldsymbol{x} \leqslant \boldsymbol{x}_{\max} \ , \tag{1}$$

where $\boldsymbol{x} = x_1, x_2, \ldots, x_d$ denotes a location in the potential space, $f(\boldsymbol{x})$ is an objective function, and $\boldsymbol{x}_{\min}$ and $\boldsymbol{x}_{\max}$ denote the bounds of the potential space.

Then the number of sparks generated by each firework $\boldsymbol{x_i}$ is defined as follows.

$$s_i = m \cdot \frac{y_{\max} - f(\boldsymbol{x_i}) + \xi}{\sum_{i=1}^{n} (y_{\max} - f(\boldsymbol{x_i})) + \xi} \ , \tag{2}$$

where m is a parameter controlling the total number of sparks generated by the n fireworks, $y_{\max} = \max(f(\boldsymbol{x_i}))$ $(i = 1, 2, \ldots, n)$ is the maximum (worst) value of the objective function among the n fireworks, and ξ, which denotes the smallest constant in the computer, is utilized to avoid zero-division-error.

To avoid overwhelming effects of splendid fireworks, bounds are defined for s_i, which is shown in Eq. 3.

$$\hat{s}_i = \begin{cases} round(a \cdot m) & \text{if } s_i < am \\ round(b \cdot m) & \text{if } s_i > bm \ , \quad a < b < 1 \ , \\ round(s_i) & otherwise \end{cases} \tag{3}$$

where a and b are const parameters.

Amplitude of Explosion. In contrast to the design of sparks number, the amplitude of a good firework explosion is smaller than that of a bad one. Amplitude of explosion for each firework is defined as follows.

$$A_i = \hat{A} \cdot \frac{f(\boldsymbol{x_i}) - y_{\min} + \xi}{\sum_{i=1}^{n} (f(\boldsymbol{x_i}) - y_{min}) + \xi} \ , \tag{4}$$

where $\hat{A}$ denotes the maximum explosion amplitude,and $y_{\min} = \min(f(\boldsymbol{x_i}))$ $(i = 1, 2, \ldots, n)$ is the minimum (best) value of the objective function among the n fireworks.

Generating Sparks. In explosion, sparks may undergo the effects of explosion from random z directions (dimensions). In the FA, we obtain the number of the affected directions randomly as follows.

$$z = round(d \cdot rand(0, 1)) \ , \tag{5}$$

where d is the dimensionality of the location $\boldsymbol{x}$, and $rand(0, 1)$ is an uniform distribution over [0,1].

The location of a spark of the firework $\boldsymbol{x_i}$ is obtained using Algorithm 1. Mimicking the explosion process, a spark's location $\tilde{\boldsymbol{x}}_j$ is first generated. Then if the obtained location is found to fall out of the potential space, it is mapped to the potential space according to the algorithm.

Algorithm 1. Obtain the location of a spark

```
Initialize the location of the spark: $\tilde{\boldsymbol{x}}_{\boldsymbol{j}} = \boldsymbol{x_i}$;
$z$=$round(d \cdot rand(0, 1))$;
Randomly select $z$ dimensions of $\tilde{\boldsymbol{x}}_{\boldsymbol{j}}$;
Calculate the displacement: $h = A_i \cdot rand(-1, 1)$;
for each dimension $\tilde{x}_k^j \in$ {pre-selected $z$ dimensions of $\tilde{\boldsymbol{x}}_{\boldsymbol{j}}$} do
  $\tilde{x}_k^j = \tilde{x}_k^j + h$;
  if $\tilde{x}_k^j < x_k^{\min}$ or $\tilde{x}_k^j > x_k^{\max}$ then
    map $\tilde{x}_k^j$ to the potential space: $\tilde{x}_k^j = x_k^{\min} + \mid \tilde{x}_k^j \mid \%(x_k^{\max} - x_k^{\min})$;
  end if
end for
```

To keep the diversity of sparks, we design another way of generating sparks — Gaussian explosion, which is show in Algorithm 2. A function $Gaussian(1, 1)$, which denotes a Gaussian distribution with mean 1 and standard deviation 1, is utilized to define the coefficient of the explosion. In our experiments, $\hat{m}$ sparks of this type are generated in each explosion generation.

Algorithm 2. Obtain the location of a specific spark

```
Initialize the location of the spark: $\hat{\boldsymbol{x}}_{\boldsymbol{j}} = \boldsymbol{x}_{\boldsymbol{i}}$;
$z$=$round(d \cdot rand(0, 1))$;
Randomly select $z$ dimensions of $\hat{\boldsymbol{x}}_{\boldsymbol{j}}$;
Calculate the coefficient of Gaussian explosion: $g = Gaussian(1, 1)$;
for each dimension $\hat{x}_k^j \in$ {pre-selected $z$ dimensions of $\hat{\boldsymbol{x}}_{\boldsymbol{j}}$} do
  $\hat{x}_k^j = \hat{x}_k^j \cdot g$;
  if $\hat{x}_k^j < x_k^{\min}$ or $\hat{x}_k^j > x_k^{\max}$ then
    map $\hat{x}_k^j$ to the potential space: $\hat{x}_k^j = x_k^{\min} + \mid \hat{x}_k^j \mid \%(x_k^{\max} - x_k^{\min})$;
  end if
end for
```

2.3 Selection of Locations

At the beginning of each explosion generation, n locations should be selected for the fireworks explosion. In the FA, the current best location $\boldsymbol{x}^*$, upon which the objective function $f(\boldsymbol{x}^*)$ is optimal among current locations, is always kept for the next explosion generation. After that, $n-1$ locations are selected based on their distance to other locations so as to keep diversity of sparks. The general distance between a location $\boldsymbol{x}_{\boldsymbol{i}}$ and other locations is defined as follows.

$$R(\boldsymbol{x}_{\boldsymbol{i}}) = \sum_{j \in K} d(\boldsymbol{x}_{\boldsymbol{i}}, \boldsymbol{x}_{\boldsymbol{j}}) = \sum_{j \in K} \parallel \boldsymbol{x}_{\boldsymbol{i}} - \boldsymbol{x}_{\boldsymbol{j}} \parallel \ , \tag{6}$$

where K is the set of all current locations of both fireworks and sparks.

Then the selection probability of a location $\boldsymbol{x}_{\boldsymbol{i}}$ is defined as follows.

$$p(\boldsymbol{x}_{\boldsymbol{i}}) = \frac{R(\boldsymbol{x}_{\boldsymbol{i}})}{\sum_{j \in K} R(\boldsymbol{x}_{\boldsymbol{j}})} \ . \tag{7}$$

When calculating the distance, any distance measure can be utilized including Manhattan distance, Euclidean distance, Angle-based distance, and so on [9]. When $d(\boldsymbol{x}_{\boldsymbol{i}}, \boldsymbol{x}_{\boldsymbol{j}})$ is defined as $\mid f(\boldsymbol{x}_{\boldsymbol{i}}) - f(\boldsymbol{x}_{\boldsymbol{j}}) \mid$, the probability is equivalent to the definition of the immune density based probability in Ref. [10].

2.4 Summary

Algorithm 3 summarizes the framework of the FA. During each explosion generation, two types of sparks are generated respectively according to Algorithm 1 and Algorithm 2. For the first type, the number of sparks and explosion amplitude depend on the quality of the corresponding firework ($f(\boldsymbol{x}_{\boldsymbol{i}})$). In the contrast, the second type is generated using a Gaussian explosion process, which conducts search in a local Gaussian space around a firework. After obtaining the locations of the two types of sparks, n locations are selected for the next explosion generation. In the FA, approximate $n + m + \hat{m}$ function evaluations are done in each generation. Suppose the optimum of a function can be found in T generations, then we can deduce that the complexity of the FA is $O(T * (n + m + \hat{m}))$.

Algorithm 3. Framework of the FA

```
Randomly select $n$ locations for fireworks;
while stop criteria=false do
    Set off $n$ fireworks respectively at the $n$ locations:
    for each firework $\boldsymbol{x_i}$ do
        Calculate the number of sparks that the firework yields: $\hat{s}_i$, according to Eq. 3;
        Obtain locations of $\hat{s}_i$ sparks of the firework $\boldsymbol{x_i}$ using Algorithm 1;
    end for
    for k=1:$\hat{m}$ do
        Randomly select a firework $\boldsymbol{x_j}$;
        Generate a specific spark for the firework using Algorithm 2;
    end for
    Select the best location and keep it for next explosion generation;
    Randomly select $n-1$ locations from the two types of sparks and the current
    fireworks according to the probability given in Eq. 7;
end while
```

3 Experiments

3.1 Benchmark Functions

To investigate the performance of the proposed FA, we conducted experiments on nine benchmark functions. The feasible bounds for all functions are set as $[-100, 100]^D$. The expression of the functions, initialization intervals and dimensionalities are listed in Table 1.

Table 1. Nine benchmark functions utilized in our experiments

Function	Expression	Initialization	D
Sphere	$F_1 = \sum_{i=1}^{D} \boldsymbol{x}_i^2$	$[30, 50]^D$	30
Rosenbrock	$F_2 = \sum_{i=1}^{D-1}(100(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i^2)^2 + (\boldsymbol{x}_i - 1)^2)$	$[30, 50]^D$	30
Rastrigrin	$F_3 = \sum_{i=1}^{D}(\boldsymbol{x}_i^2 - 10\cos(2\pi\boldsymbol{x}_i) + 10)$	$[30, 50]^D$	30
Griewank	$F_4 = 1 + \sum_{i=1}^{D} \frac{\boldsymbol{x}_i^2}{4000} - \prod_{i=1}^{D}\cos(\frac{\boldsymbol{x}_i}{\sqrt{i}})$	$[30, 50]^D$	30
Ellipse	$F_5 = \sum_{i=1}^{D} 10^{4\frac{i-1}{D-1}} \boldsymbol{x}_i^2$	$[15, 30]^D$	30
Cigar	$F_6 = \boldsymbol{x}_1^2 + \sum_{i=2}^{D} 10^4 \boldsymbol{x}_i^2$	$[15, 30]^D$	30
Tablet	$F_7 = 10^4 x_1^2 + \sum_{i=2}^{D} x_i^2$	$[15, 30]^D$	30
Schwefel	$F_8 = \sum_{i=1}^{D}((\boldsymbol{x}_1 - \boldsymbol{x}_i^2)^2 + (\boldsymbol{x}_i - 1)^2)$	$[15, 30]^D$	30
Ackley	$F_9 = 20 + \mathrm{e} - 20exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}\boldsymbol{x}_i^2}\right) -exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi\boldsymbol{x}_i^2)\right)$	$[15, 30]^D$	30

3.2 Comparison Experiments among the FA, the CPSO and the SPSO

In this section, we compare the performance of the FA with the CPSO and the SPSO in terms of both convergence speed and optimization accuracy.

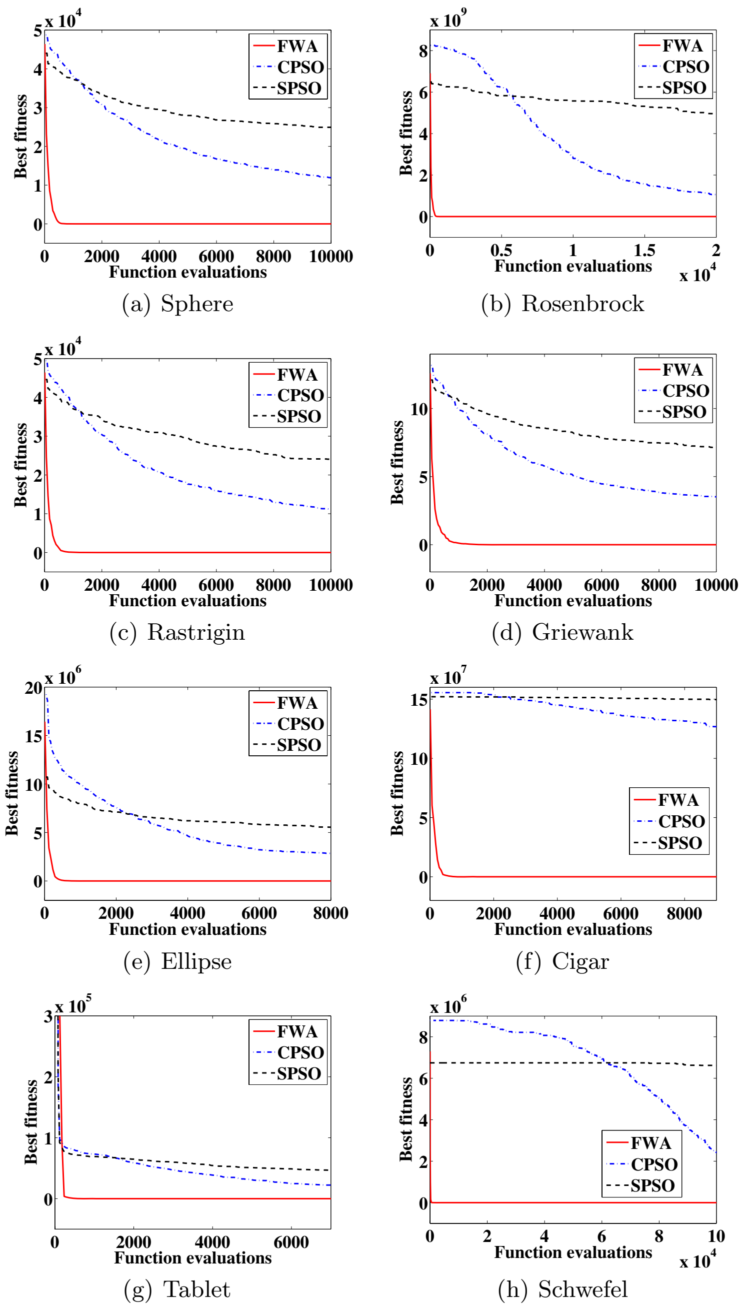
Table 2. Statistical mean and standard deviation of solutions found by the FA, the CPSO and the SPSO on nine benchmark functions over 20 independent runs

Function	Function evluations	FA's mean (StD)	CPSO's mean (StD)	SPSO's mean (StD)
Sphere	500000	0.000000 (0.000000)	0.000000 (0.000000)	1.909960 (2.594634)
Rosenbrock	600000	9.569493 (12.12829)	33.403191 (42.513450)	410.522552 (529.389139)
Rastrigrin	500000	0.000000 (0.000000)	0.053042 (0.370687)	167.256119 (42.912873)
Griewank	200000	0.000000 (0.000000)	0.632403 (0.327648)	2.177754 (0.294225)
Ellipse	500000	0.000000 (0.000000)	0.000000 (0.000000)	53.718807 (68.480173)
Cigar	600000	0.000000 (0.000000)	0.000000 (0.000000)	0.002492 (0.005194)
Tablet	500000	0.000000 (0.000000)	0.000000 (0.000000)	1.462832 (1.157021)
Schwefel	600000	0.000000 (0.000000)	0.095099 (0.376619)	0.335996 (0.775270)
Ackley	200000	0.000000 (0.000000)	1.683649 (1.317866)	12.365417 (1.265322)

The parameters of both the CPSO and the SPSO are set as those in Ref. [8]. For the FA, the parameters were selected by some preliminary experiments. We found that the FA worked quite well at the setting: $n = 5$, $m = 50$, $a = 0.04$, $b = 0.8$, $\hat{A} = 40$, and $\hat{m} = 5$, which is applied in all the comparison experiments.

Table 2 depicts the optimization accuracy of the three algorithms on nine benchmark functions, which are averaged over 20 independent runs. It can be seen that the proposed FA clearly outperforms both the CPSO and SPSO on all the functions. In addition, the FA can find optimal solutions on most benchmark functions in less than 10000 function evaluations, as shown in Table 3. However, the optimization accuracy of the CPSO and the SPSO is unacceptable within 10000 function evaluations.

Besides optimization accuracy, convergence speed is quite essential to an optimizer. To validate the convergence speed of the FA, we conducted more thorough experiments. Fig. 3 depicts the convergence curves of the FA, the CPSO and the SPSO on eight benchmark functions averaged over 20 independent runs. From these results, we can arrive at a conclusion that the proposed FA has a much faster speed than the CPSO and the SPSO. From Table 3, we can find that the FA can find excellent solutions with only 10000 times of function evaluations. This also reflects the fast convergence speed of the proposed FA.



(a) Sphere (b) Rosenbrock

(c) Rastrigin (d) Griewank

(e) Ellipse (f) Cigar

(g) Tablet (h) Schwefel

Fig. 3. Convergence curves of the FA, the CPSO and the SPSO on eight benchmark functions. The function fitness are averaged over 20 independent runs.

Table 3. Statistical mean and standard deviation of solutions found by the FA, the CPSO and the SPSO on nine benchmark functions over 20 independent runs of 10000 function evaluations

Function	FA's mean (StD)	CPSO's mean (StD)	SPSO's mean (StD)
Sphere	0.000000 (0.000000)	11857.425781 (3305.973067)	24919.099609 (3383.241523)
Rosenbrock	19.38330 (11.94373)	2750997504.000000 (1741747548.420642)	5571942400.000000 (960421617.568024)
Rastrigrin	0.000000 (0.000000)	10940.148438 (3663.484331)	24013.001953 (4246.961530)
Griewank	0.000000 (0.000000)	3.457273 (0.911027)	7.125976 (0.965788)
Ellipse	0.000000 (0.000000)	2493945.500000 (1199024.648305)	5305106.500000 (1117954.409340)
Cigar	0.000000 (0.000000)	122527168.000000 (28596381.089661)	149600864.000000 (13093322.778560)
Tablet	0.000000 (0.000000)	15595.107422 (8086.792234)	42547.488281 (8232.221882)
Schwefel	4.353733 (1.479332)	8775860.000000 (1217609.288290)	6743699.000000 (597770.084232)
Ackley	0.000000 (0.000000)	15.907665 (1.196082)	18.423347 (0.503372)

3.3 Discussion

As shown in the experiments, the FA has a faster convergence speed and a better optimization accuracy, compared to the PSO. We consider the success of the FA lies in the following two aspects.

– In the FA, sparks suffer the power of explosion from z dimensions simultaneously, and the z dimensions are randomly selected for each spark $\tilde{\boldsymbol{x}}_{\boldsymbol{i}}$. Thus, there is a probability that the differences between the firework and the target location happen to lie in these z dimensions. In this scenario, the sparks of the firework can move towards the target location from z directions simultaneously, which endues the FA with a fast convergence speed.
– Two types of sparks are generated to keep the diversity of sparks, and the specific selection process for locations is a mechanism for keeping diversity. Therefore, the FA has the capability of avoiding premature convergence.

4 Conclusions

Mimicking the explosion process of fireworks, the so-called FA is proposed and implemented for function optimization. The experiments among the FA, the CPSO and the SPSO have shown that the proposed FA has a promising performance. It clearly outperforms the CPSO and the SPSO on nine benchmark functions in terms of both optimization accuracy and convergence speed, which endues the FA with a promising prospect of application and extension. In future work, we will seek a deep theoretical analysis on the FA and try to apply the FA to some practical engineering applications. Finally, we intend to discuss the relationship between the FA and other general-purpose optimization algorithms.

Acknowlegements. This work was supported by National Natural Science Foundation of China (NSFC), under Grant No. 60875080 and 60673020, and partially supported by the National High Technology Research and Development Program of China (863 Program), with Grant No. 2007AA01Z453. Authors appreciate Mr. Chao Rui for his facility with the early phase of the experiments.

References

1. Garnier, S., Gautrais, J., Theraulaz, G.: The biological principles of swarm intelligence. Swarm Intelligence 1(1), 3–31 (2007)
2. Das, S., Abraham, A., Konar, A.: Swarm intelligence algorithms in bioinformatics. Studies in Computational Intelligence 94, 113–147 (2008)
3. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948 (1995)
4. 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)
5. De Castro, L.N., Von Zuben, F.J.: Learning and optimization using the clonal selection principle. IEEE Transactions on Evolutionary Computation 6(3), 239–251 (2002)
6. Beni, G., Wang, J.: Swarm intelligence in cellular robotic systems. In: Proceedings of NATO Advanced Workshop on Robots and Biological Systems (1989)
7. Bratton, D., Kennedy, J.: Defining a standard for particle swarm optimization. In: Proceedings of IEEE Swarm Intelligence Symposium, pp. 120–127 (2007)
8. Tan, Y., Xiao, Z.M.: Clonal particle swarm optimization and its applications. In: Proceedings of IEEE Congress on Evolutionary Computation, pp. 2303–2309 (2007)
9. Perlibakas, V.: Distance measures for PCA-based face recognition. Pattern Recognition Letters 25(6), 711–724 (2004)
10. Lu, G., Tan, D., Zhao, H.: Improvement on regulating definition of antibody density of immune algorithm. In: Proceedings of the 9th International Conference on Neural Information Processing, vol. 5, pp. 2669–2672 (2002)

Bacterial Foraging Optimization Algorithm with Particle Swarm Optimization Strategy for Distribution Network Reconfiguration

Tianlei Zang, Zhengyou He, and Deyi Ye

School of Electrical Engineering, Southwest Jiaotong University,
610031, Sichuan Province, China
zangtianlei@126.com, hezy@swjtu.cn, 20061648@163.com

Abstract. Distribution network reconfiguration for loss minimization is a complex, large-scale combinatorial optimization problem. In this paper, a novel method called bacterial foraging optimization algorithm with particle swarm optimization strategy (BF-PSO) algorithm is applied to solve this problem. To verify the effectiveness of the proposed method, the optimization calculations of IEEE 69-bus testing system by the presented method are conducted and the calculation results are compared with pertinent literatures. Simulation results show that the proposed algorithm possesses fast convergence speed while the quality of solution and stability is ensured.

Keywords: power system, distribution network reconfiguration, power loss reduction, bacterial foraging optimization algorithm, BF-PSO algorithm.

1 Introduction

As an integral part of electric power system, power distribution system plays an important role in connecting electric power generation and the consumers. The distribution network is generally designed in closed loop and operated in open loop, and it involves large quantity of sectionalizing switches and a small amount of tie switches. The distribution network reconfiguration (DNR) is to change the network topology through opening and closing these switches in order to optimize network operation parameters.

Theoretically, the reconfiguration of distribution network is a complicated problem of multi constraints, high dimensions and large-scale nonlinear combinatorial optimization which is a NP-hard problem. At present, there are a lot of methods for the reconfiguration of distribution network, such as the traditional optimal methods [1], the heuristic method [2], artificial intelligence method such as genetic algorithm (GA) [3], ant swarm algorithm (ASA) [4], particle swarm algorithm (PSO) [5], etc. Although, in order to improve the convergence and calculation speed, there appeared a lot of improved and hybrid algorithms [6]-[10], the problem is not fundamentally solved.

In this paper, a new distribution network reconfiguration method employing bacterial foraging optimization algorithm with particle swarm optimization (BF-PSO) as optimal means is proposed. The proposed method is tested on IEEE 69-bus networks.

This paper consists of 5 sections. Section 2 formulates the problem of reconfiguration for distribution network loss minimization. Section 3 presents bacterial foraging optimization algorithm and BF-PSO algorithm. Section 4 provides the BF-PSO algorithm example analysis results for distribution network reconfiguration. Finally, in section 5, the main conclusions are addressed.

2 Formulation of DN Reconfiguration Problem

2.1 Mathematical Model of DN Reconfiguration Problem

In a more constrained energy environment, the electric distribution operators are more and more interested in the minimization of energy losses. So, this paper considers only the losses minimization while satisfying operation constrains. The loss-minimum reconfiguration problem generally can be mathematically expressed as:

$$\min f = \sum_{i=1}^{n} k_i r_i \,(P_i^2 + Q_i^2)/V_i^2 \,. \tag{1}$$

Where r_i is the resistance of branch i; P_i and Q_i are the active power and reactive power of branch i respectively; V_i is the voltage of the end node of branch i; n is the total number of branches; k_i represents the topological status of the branch i, $k_i = 1$ if the branch i is closed, and $k_i = 0$ if the branch i is open; f is the system losses. The controlling switches depend on the constraints: power flow constraints, network voltage constraints, capability constraints and topology constraints in terms of network structure requirements. Subject to the following:
1) Power flow constraint

$$\mathbf{AP} = \mathbf{D}\,. \tag{2}$$

Where $\mathbf{A}$ is node-branch associated matrix; $\mathbf{P}$ is the vector of power flow; $\mathbf{D}$ is the vector of load requirement.
2) Voltage constraint

$$V_{\min} \le V_i \le V_{\max}\,. \tag{3}$$

Where $V_{\min}$ and $V_{\max}$ are respectively the lower bound and upper bound of nodal voltage.
3) Capability constraint

$$S_i \le S_{i\max}\,. \tag{4}$$

Where $S_{i\max}$ is the upper bound of the branch capacity.
4) Radial Structure Constraint

$$g \in G\,. \tag{5}$$

Where g is the current network structure, and G is the set of all allowed radial network structures (i.e. the set excluding loop and island).

2.2 Process of DN Reconfiguration Problem

Fig.1 is the process flow of reconfiguration of distribution network.

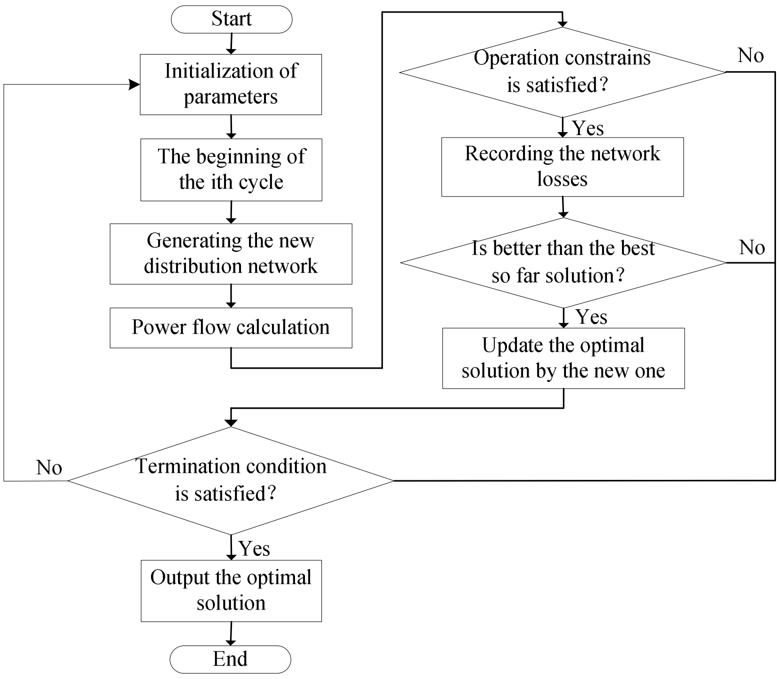

Start
Initialization of parameters
The beginning of the ith cycle
Generating the new distribution network
Power flow calculation
Operation constrains is satisfied?
No
Yes
Recording the network losses
Is better than the best so far solution?
No
Yes
Update the optimal solution by the new one
No
Termination condition is satisfied?
Yes
Output the optimal solution
End

Fig. 1. Flow chart of distribution network reconfiguration

3 Bacterial Foraging Optimization Algorithm with Particle Swarm Optimization Strategy

3.1 The Classical Bacterial Foraging Optimization Algorithm

In the process of foraging, E. coli bacteria undergo four stages, namely, chemotaxis, swarming, reproduction, elimination and dispersal. In search space, BFOA seek optimum value through the chemotaxis of bacteria, and realize quorum sensing via assembling function between bacterial, and satisfy the evolution rule of the survival of the fittest make use of reproduction operation, and use elimination-dispersal mechanism to avoid falling into premature convergence [11].

1) Chemotaxis

This process was simulated by two different moving ways: run or tumble. This alternation between the two modes will move the bacterium, which enables it to "search" for nutrients. Supposing $\theta^i(j,k,l)$ represents the position of the each member in the population of *S* bacterial at the *j*th chemotactic step, *k*th reproduction step, and *l*th elimination. The movement of bacterium may be presented by:

$$\theta^i(j+1,k,l)=\theta^i(j,k,l)+C(i)\phi(j)\,. \tag{6}$$

Where $C(i)>0$ is the size of the step taken in the random direction specified by the rumble. $\phi(j)$ indicates a unit length vector in the random direction.

2) Swarming

To achieve swarming, function to model the cell-to-cell signaling via an attractant and a repellent. The mathematical representation can be represented by:

$$\begin{aligned}J_{cc}(\theta,P(j,k,l)) &= \sum_{i=1}^{S} J_{cc}^{i}(\theta,\theta^i(j,k,l))\\ &= \sum_{i=1}^{S}\left[-d_{attract}\exp\left(-\omega_{attract}\sum_{m=1}^{p}(\theta_m-\theta_m^i)^2\right)\right]\\ &\quad+\sum_{i=1}^{S}\left[-h_{repellent}\exp\left(-\omega_{repellent}\sum_{m=1}^{p}(\theta_m-\theta_m^i)^2\right)\right]\end{aligned}\,. \tag{7}$$

Where $J_{cc}(\theta,P(j,k,l))$ is the cost function value to be added to the actual cost function. S is the total number of bacteria and p is the number of parameters to be optimized which are presented in each bacterium. $d_{attract}$ is the depth of the attractant released by the cell and $\omega_{attract}$ is a measure of the width of the attractant signal. $h_{repellent}=d_{attract}$ is the height of the repellent effect and $\omega_{repellent}$ is a measure of the width of the repellent.

3) Reproduction

According to the rules of evolution, individual will reproduce themselves in appropriate conditions in a certain way. For bacterial, a reproduction step takes place after all chemotactic steps.

$$J_{health}^{i}=\sum_{j=1}^{N_c+1}J(i,j,k,l)\,. \tag{8}$$

Where J_{health}^{i} is the health of bacterium i. Sort bacteria and chemotactic parameters $C(i)$ in order of ascending cost (higher cost means lower health). To keep a constant population size, bacteria with the highest J_{health} values die. The remaining bacteria are allowed to split into two bacteria in the same place.

4) Elimination-Dispersal

From the evolutionary point of view, elimination and dispersal were used to guarantee diversity of individuals and to strengthen the ability of global optimization. In BFOA, bacteria are eliminated with a probability of P_{ed}. In order to keep the number of bacteria in the population constant, if a bacterium is eliminated, simply disperse one to a random location on the optimization domain.

3.2 BF-PSO Algorithm

In 2008, W. Korani proposed the BF-PSO algorithm [12], it combined both algorithms BF and PSO. The aim is to make use of PSO ability to exchange social information and BF ability to find a new solution by elimination and dispersal. In the BF-PSO, the unit length random direction of tumble behavior can be decided by the global best position and the best position of each bacterium. During the chemotaxis loop, the update of the tumble direction is determined by:

$$\phi(j+1) = w \cdot \phi(j) + C_1 \cdot R_1(P_{lbest} - P_{current}) + C_2 \cdot R_2(P_{gbest} - P_{current}) . \tag{9}$$

Where P_{lbest} is the best position of each bacterial and P_{gbest} is the global best bacterial. The brief pseudo-code of the BF-PSO has been provided in [12].

4 Algorithm Example Analysis for DN Reconfiguration

In order to study BF-PSO algorithm performance, tests are done on IEEE 69-bus system [13] as shown in Fig.2. The system consists of 73 branches, 68 sectionalizing switches and 5 tie switches. Prior to reconfiguration, the open switches of power distribution network are TS11-66, TS13-20, TS15-69, TS27-54 and TS39-48. The rated voltage is 12.66 kV and the total load is 3802.2kW+j2694.6kvar, while the initial system power loss is 225.0kW.

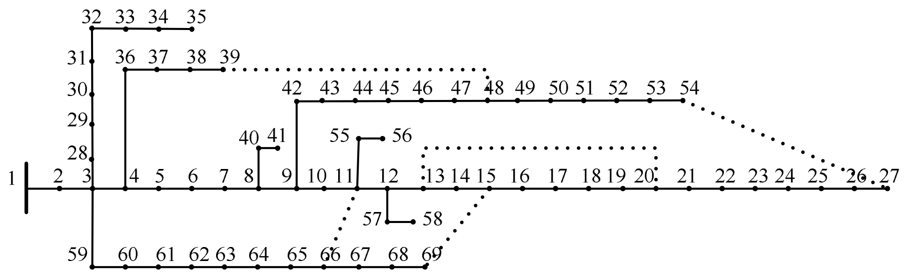


Fig. 2. IEEE 69-bus test system

The Article writes a distribution network reconfiguration program in Matlab Language. Use a computer of 2.66 GHz CPU and Windows 2002 operating system to do calculation. To ensure the accuracy and speed of the solution process as much as possible, the parameters of the algorithm are adjusted in this paper on the basis of empirical values. Selecting swarm scale is 50, the chemotactic loop number N_c is 4, the number of reproduction steps N_{re} is 4, the number of elimination and dispersal events N_{ed} is 2, the probability of elimination and dispersal P_{ed} is 0.25, inertia weight is 0.9, C_1 and C_2 is 1.2, and the maximum iteration number is 100. The power flow is calculated through back/forward sweep method, and the convergence condition is the

Table 1. Result Before and After Reconfiguration

	The open switches					Power losses (kW)	The minimum node voltage(pu)
Before DNR	39-48	54-27	13-20	11-66	15-69	225.00	0.909
Optimal Reconfiguration Scheme	44-45 45-46 46-47 47-48	50-51	13-20	11-66	14-15	99.670	0.943
Suboptimal	Ditto	50-51	13-20	11-66	13-14	99.766	0.943
Scheme	Ditto	50-51	13-20	11-66	12-13	99.872	0.943
The Worst Scheme	Ditto	52-53	13-20	11-66	13-14	100.807	0.941

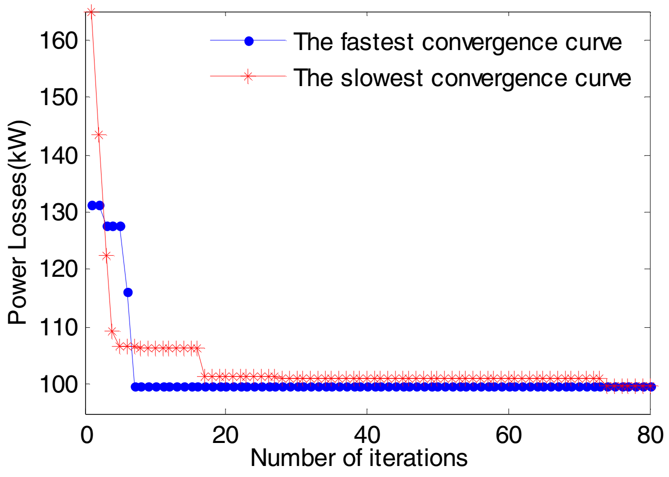


Fig. 3. The convergence curve of the BF-PSO method

maximum correction value of the voltage magnitude of all the nodes is less than 10^{-6} kV.

The BF-PSO algorithm program is continuous operated for 100 times based on the case. After simulation, the calculated results are shown in Table 1. The convergence curve of the BF-PSO method is shown in Fig. 3.

It is clear that network loss decreased substantially and the minimum node voltage rose, after introduction of BF-PSO algorithm for distribution network reconfiguration. The active power loss is reduced dramatically from 225 kW to 99.67 kW. The minimum node voltage is improved from 0.909 to 0.943. The statistical analysis of optimization results are shown in Table 2.

Table 2. The statistical analysis of optimization results

	Minimum	Maximum	Mean	Standard deviation
Power losses(kW)	99.670	100.807	99.887	0.374
Convergence generations	7	74	14.38	9.31

As shown in Table 2, the optimal solution is obtained after about an average of 14-15 iterations. So the convergence rate of the proposed method is very quickly. Meanwhile, these results demonstrate the good stability and effectiveness of the proposed method. The optimal topology of distribution network is shown in Fig. 4.

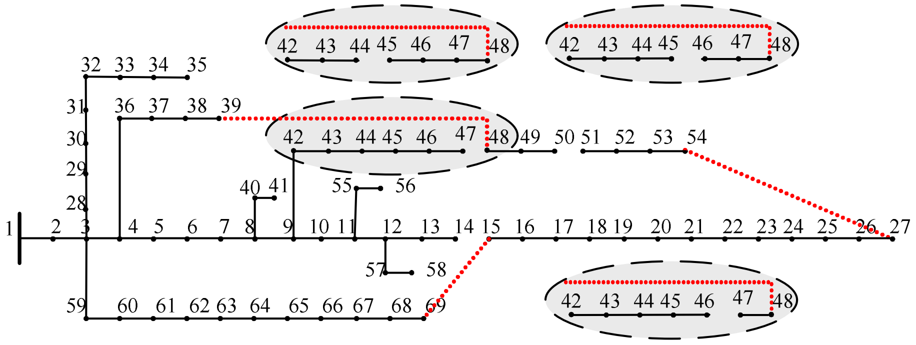


Fig. 4. The optimal topology of distribution network

The result is compared with the results obtained by other methods as shown in Table 3. While the computing environments of running the algorithm vary, but the optimization results have reached the optimal results can be considered by the adjustment of algorithm parameters. So this also validates the effectiveness of the proposed method.

Table 3. Comparison of reconfiguration methods

	BF-PSO	IGA[6]	ITS[7]	BPSO[8]	ACS[9]	RGA[10]
The losses after DNR(kW)	99.670	100.697	101.000	101.010	101.098	102.100
The minimum node voltage(pu)	0.943	0.933	0.947	0.942	0.943	0.926

5 Conclusion

This study has presented the bacterial foraging optimization algorithm with particle swarm optimization strategy (BF-PSO) applied to solve the distribution network reconfiguration problem. Through the simulation calculation for reconfiguration optimization of IEEE-69 bus system and the comparison with other optimization methods, the validity and effectiveness of the proposed method have been demonstrated.

References

1. Deng, Q., Sun, C.-X., Zhou, Q., Zhang, X.-X., Cheng, Q.-Y.: Realization of Distribution Power Recoering with Dynamic Programming. Journal of Chongqing University (Natural Science Edition) 29(3), 40–44 (2006)
2. Tang, Q.-G., Lin, J.-D.: Reconfiguration of Distribution Network Algorithm of Power Loss Reduction based on Heuristic Search. RELAY 35(14), 10–12 (2007)
3. Esther, R.R., Antonio, G.E., Jesús, R.S., Francisco, L.I.: Path-based distribution network modeling: Application to reconfiguration for loss reduction. IEEE Transactions on Power Systems 20(2), 556–564 (2005)
4. Su, C.-T., Chang, C.-F., Chiou, J.-P.: Distribution network reconfiguration for loss reduction by ant colony search algorithm. Electric Power Systems Research 75(2-3), 190–199 (2005)
5. Sanjib, G., Sahoo, N.C., Das, D.: Multi-objective Expansion Planning of Electrical Distribution Networks Using Comprehensive Learning Particle Swarm Optimization. In: Advances in Soft Computing, vol. 58, pp. 193–202 (2009)
6. Tang, B., Luo, A., Wang, J.: The Improved Encoding Strategy of Genetic Algorithm and Its Application in Reconfiguration of Distribution Networks. RELAY 32(13), 35–39 (2004)
7. Zhang, Z.-H., Li, M.-L., Xiong, N., Zhang, W.-C.: An Improved Tabu Search for Reconfiguration of Distribution Systems. RELAY 35(10), 41–44 (2007)
8. Lu, Z.-G., Yang, G.-L., Zhang, X.-H., Wen, Y.: Reconfiguration of Distribution Network based on Improved Particle Swarm Optimization. Power System Protection and Control 37(7), 30–34 (2007)
9. Yao, L.-X., Ren, Y.-N., Fei, J.-A.: Ant Colony System Algorithm for Distribution Network Reconfiguration. Proceedings of the CSU-EPSA 19(6), 35–39 (2007)
10. Bi, P.-X., Liu, J., Liu, C.-X., Zhang, W.-Y.: A Refined Genetic Algorithm for Power Distribution Network Reconfiguration. Automation of Electronic Power Systems, 57–61 (2002)
11. Shen, H., Zhu, Y.L., Zhou, X.M., Guo, H.F., Chang, C.G.: Bacterial foraging optimization algorithm with particle swarm optimization strategy for global numerical optimization. In: 2009 World Summit on Genetic and Evolutionary Computation, pp. 497–504 (2009)
12. Korani, W.: Bacterial foraging oriented by particle swarm optimization strategy for PID tuning. In: GECCO 2008: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation 2008, pp. 1823–1826 (2008)
13. Chiang, H.-D., Rene, J.-J.: Optimal network reconfigurations in distribution systems: Part 2 - Solution algorithms and numerical results. IEEE Transactions on Power Delivery 5(3), 1568–1574 (1990)

Optimization Design of Flash Structure for Forging Die Based on Kriging-PSO Strategy

Yu Zhang[1,*], Zhiguo An[1], and Jie Zhou[2]

[1] School of Mechatronics and Automotive Engineering, Chongqing Jiaotong University, Chongqing 400074, China
400074@gmail.com
[2] College of Materials Science and Engineering, Chongqing University, Chongqing 400044, China
a65105410@cta.cqu.cn

Abstract. The article concern the influences of the novel flash structure with resistance wall on the forming load and die wear during the closed-die forging process. If the geometrical parameters of the resistance wall are not properly designed, the forming load and die wear may be great; this can affect the life of the forging die. Yet, no procedure is known to optimize and decide this flash structure at the exit. So, the optimization calculations were carried out using an authors optimal strategy, called Kriging-PSO strategy. According to this strategy we got the optimum parameters of the resistance wall. The strategy incorporates finite element simulations software Deform and includes particle swarm optimization (PSO) with a fast convergence and Kriging interpolation algorithm. The optimization results which represent the best die design were realized in industries.

Keywords: Forging, Kriging, Particle swarm optimization (PSO), Plastic deformation, Finite element method.

1 Introduction

In the use of conventional forging process, the common forms of flash cave are shown in Fig.1 [1].At present, the flash cave with parallel land which shown in Fig.1 (a) is the most widely used form in industrial conditions. Because of the generation of resistance by friction mainly depends on the land, this greater force can be obtained only in the later stage. It indicates that there is disadvantage for the metal filling in the complex cavity. The flash cave with wedged land is shown in Fig.1 (b). Due to the slope can exert horizontal forces against billet except friction force; the resistance can be obtained enough to fill cavity in the initial phase of forming. However, the follow-up trimming become difficult because of the metal thickening on the flash transition region. V-shaped cave as shown in Fig.1(c) apply only to simple forging parts which can be easy filled.

* Corresponding author.

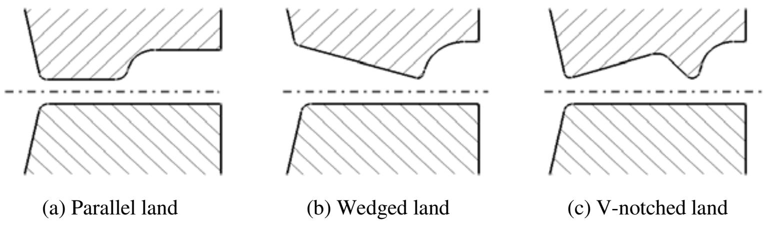
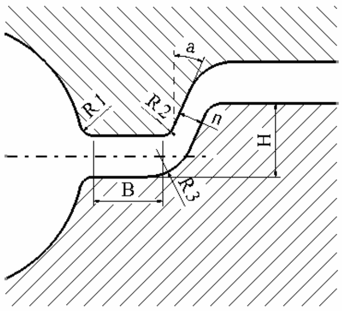


Fig. 1. Conventional flash cave in closed-die forging



*R*1, *R*2, *R*3 - fillet *B* - land width *H* - wall height *n* - wall gap *a* - wall angle

Fig. 2. The resistance wall structure

For the above reasons, the flash cave is replaced by the resistance wall structure as shown in Fig.2 which kept the parallel land but changed the gutter part to the wall. The resistance wall combines the advantages of the flash cave with parallel land and with wedged land: it can provide the horizontal resistance by means of the slope of wall and avoid the trimming difficulty with the parallel land instead of the wedged land.

The practice proves that the resistance wall structure can improve the filling ability to a great extent by producing larger horizontal resistance and forcing the billet to fill the die cavity in the initial stage of forming. The major advantage of the resistance wall is more material utilization compared with the conventional flash caves as shown in Fig.1.

In designing the resistance wall which as a novel flash structure, there are no ready data for reference. So, in the practical engineering, the forming load would greatly increase when the parameters of resistance wall were improper and the die life would be shorted.

In this research, the parameters of the resistance wall structure were optimized with the goal to decrease abrasive wear of die and forming load. First, the latin hypercube sampling was carry out; secondly, the sample points were simulated with finite element method; thirdly, the Kriging model was established and finally the optimum parameters of the resistance wall structure were obtained by using the particle swarm algorithm for global optimization.

2 Principle And Model

2.1 Surrogate Model

In the current engineering design, the analysis of the metal plastic forming problem is done based on the finite element simulation and trial and error method. However, since the forging is a complicated three-dimension bulk forming technique, the FEM simulation may take a long time and the trial and error method not only consumed many hours but also can't obtain the optimum result. So, the optimization method based on surrogate model is an effective approach to solve this problem.

The process of modeling is divided into three stages: generate sample points, FEM simulation and construction surrogate model.

The most commonly sampling methods include the orthogonal experiment design, latin hypercube sampling, etc. The latin hypercube sampling is a space filling design and can reflects the characteristics of the whole design space with less sampling points, so it is widely applied in computer simulation and experiment.

The surrogate model can be established taking the simulation result as the response and the parameters of the resistance wall as the design variables. The common models include the response surface modes (RSM), the artificial neural network (ANN) model, the Kriging model, and others. The Kriging model can fit the very complicated problem with less uniform sample points. So, the Kriging model is adopted in this study.

2.2 Kriging Model

The Kriging model is an unbiased estimate model, which is established by minimizing the variance with the statistical techniques based on stochastic process. Due to the existence of the stochastic part in the structure, this model has the ability for the partial estimation with the action of correlation function. So, it has good fitting performance for solving the nonlinear problem [2].

The Kriging model can be expressed as follows:

$$Y(x) = f^T(x)\beta + Z(x) \tag{1}$$

Where $f^T(x)\beta$ is the regression model, $f(x)=[f_1(x),...,f_k(x)]^T$, where k denotes the number of the basis function in regression model, $\beta=[\beta_1,...,\beta_k]^T$ is the coefficient vector, x is the design variables, $Y(x)$ is the objective, and $Z(x)$ is the random fluctuation with the mean value is 0 and the covariance is given as

$$\mathrm{cov}[z(w), z(x)] = \sigma^2 R(\theta, w, x) \tag{2}$$

Where $R(\theta, w, x)$ is correlation function, θ is the parameter vector of correlation function.

The polynomial coefficients of regression model in the part one of Eq.(1) and the variance estimate of random fluctuation in the part two of Eq.(1) can be obtained by solving minimum variance of the forecasting model using the unbiased estimate conditions as the constrained condition. Then the approximated model can be established [3].

2.3 Particle Swarm Optimization

Particle swarm optimization (PSO) algorithm is a computation technique with swarm intelligence for global optimization [4-6]. In PSO, the each solution of the optimization problem is regarded as a particle in a D-dimensional space.

The velocity and position of each particle in the solution space can be updated by the individual best particle position *pbest* and the global best particle position *gbest*. It can be described as:

$$V_{j,g}^{t+1} = w \times V_{j,g}^{t} + c_1 \times r_1 \times (pbest_{j,g} - x_{j,g}^{t}) + c_2 \times r_2 \times (gbest_g - x_{j,g}^{t}) \tag{3}$$

$$x_{j,g}^{t+1} = x_{j,g}^{t} + v_{j,g}^{t+1}, \quad j = 1,2,...,n \quad g = 1,2,...,m \tag{4}$$

where n is the size of swarm, m is the dimension number of the velocity and position, t is the iteration number, w is a inertia weight factor, $v_{j,g}^{t}$ is the gth dimension velocity of particle j at iteration t, $x_{j,g}^{t}$ is the gth dimension position of particle j at iteration t, c_1 is a cognition weight factor, c_2 is a social weight factor, r_1 and r_2 are two random numbers, $pbest_{j,g}$ is the gth dimension of the own best position of particle j, $gbest_g$ is the gth dimension of the best particle in the swarm.

The particle search in the solution space and change the velocity according to Eq. (3). The new position can be updated using Eq. (4). Putting the new position $x_{j,g}^{t+1}$ into the objective function $f(x)$, then the fitness of current particle can be got:

$$F(j) = f(x_{j,g}^{t+1}) \tag{5}$$

3 Optimization Strategy

In this study, PSO algorithm and Kriging model were coupled by the software MATLAB. It was used to optimize the parameters of the resistance wall.

This methodology applied the latin hypercube sampling (LHS) to generate initial experiment samples of the resistance wall. Then the Kriging models of forming load and die wear were obtained by Kriging interpolation based on numerical simulation results: Load-Kriging model and Wear-Kriging model. These surrogate models were converted to single objective function as fitness function by using a weighting method. This problem was optimized by particle swarm optimization algorithm.

As shown in Fig.3, the optimization procedure can be described as three steps. Step 1, the Kriging model is established based on the simulation. Step 2, the multi-objective problem is transformed into a single objective problem. Step 3, the problem can be optimized using PSO.

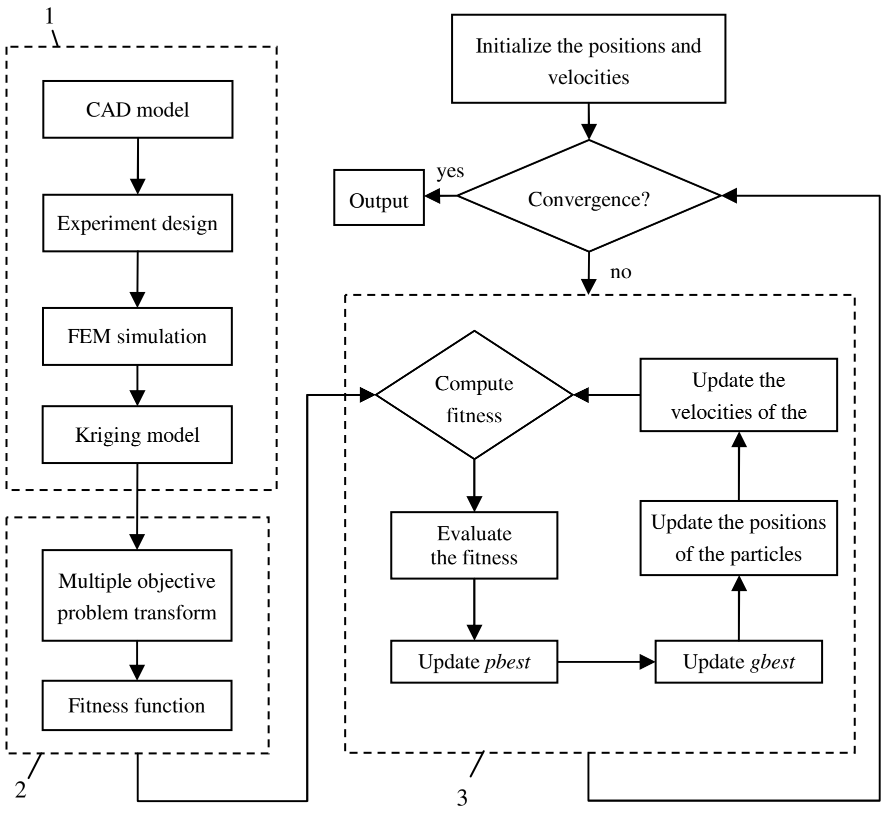


Fig. 3. Flowchart of the Kriging-PSO strategy

4 The Application

In this paper, the finish forging-die of crankshaft was taken as an example. Due to the maximum forming load and the die wear appeared in the finish forging phase, only the resistance wall structure of the finish forging-die was optimized.

Because of the forging process was performed on hot die forging press which can ensure the height of flash by adjusting the die shut height, the height of flash was set at 6 mm in simulations. The $R2$ in the Fig.2 was set at 4 mm based on practical experiences. Finally, the parameters which need to be optimized were chosen as $R1$, $R3$, B, H, a, n. The range of these parameters is presented in Table 1.

The least number of LHS can be obtained according to the equation [7]:

$$n = (m+1)(m+2)/2 = 28$$

Where n is the number of LHS and m is the number of design variables.

Table 1. Design Variables

Variable	$R1$ / mm	$R3$ / mm	B / mm	H / mm	a / deg	n / mm
Range	1~3	2~8	15~25	12~30	5~20	2~4

The plastic forming simulations for the resistance wall with different parameters according to the 28 samples were done and the forming load and die wear were calculated as the responses to construct the surrogate models.

The FEM simulation was done by using Deform software. The billet material is 42CrMo which came from the results of pre-forming. The other simulation parameters were set as: the forming speed 250mm/s, the die temperature 250℃. Archard's model was selected as wear model of the forging die with the die hardness HRC42 [8].

In this paper, the Kriging models were constructed using DACE toolbox for matlab [9] with the polynomials as model of regression and the Gaussian function as correlation function. The parameter of correlation function, θ in the Eq. (2) has an obvious influence on the fitting precision. The optimal value of θ can be calculated using maximum likelihood estimation but it is very difficult to solve [10]. The purpose of this paper was to obtain the design variables which have the minimum response, and only the forecast tendency is needed, so the fitting precision of the model with θ=3 can meet the needs completely.

To indicate the Wear-Kriging model with W_{model} and the Load-Kriging model with L_{model}, then:

$$W_{\text{mod}el} = \text{mod}\,el(S, Y_w) \tag{6}$$

$$L_{\text{mod}el} = \text{mod}\,el(S, Y_l) \tag{7}$$

Where S is the design variable, Y_l is the forming load and Y_w is the die wear according to simulation.

The multi-objective optimization problem was transformed into single-objective by using the linear weighted summation method: the weight factors, w_i were assigned to the each sub-objective functions according to the significance degree.

According to Eq. (6) and Eq. (7), the objective function was formulated as

$$\min f(x) = w \bullet W_{\text{mod}el}(x) + (1-w) \bullet L_{\text{mod}el}(x) \tag{8}$$

In engineering practice, the value of weight factor w can be decided reasonably by comprehensive comparison of the cost for the die wear and the energy consumption of load based on the technical economy analysis. In this paper, the w was 0.7 according to the practical experience.

5 Results and Discussion

The optimization strategy was implemented by Matlab. Due to the order of magnitude of the wear (1e-5 mm) and the load (1e+7 N) was different widely, the scale transformation was applied on the Eq. (8) to avoid the small order part was neglected. According to Eq. (5) and Eq. (8), the fitness function of PSO was transformed to:

$$F(j) = w \bullet W_{model}(x_{j,g}^{t+1}) \times 10^{5} + (1-w) \bullet L_{model}(x_{j,g}^{t+1}) \times 10^{-7} \tag{9}$$

The main parameters of PSO were set as: number of particles was 30, iteration time was 200, maximum velocity of the particle was 4, cognition weight factor *c*1 and social weight factor *c*2 was 2, inertia weight factor *w* was 0.9.

The minimum fitness value evolving process is shown as Fig.4 with iteration times as the abscissa and fitness according to Eq. (9) as the ordinate.

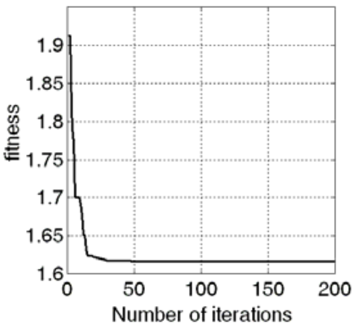


Fig. 4. Search traces for optimum solution by PSO

It can be seen from the Fig.4 that the convergence is fast and reaches stability after 50 iterations. The optimum parameters for the resistance wall were obtained with 200 iterations, as shown in Table 2. The search traces for optimum solution of wear depth and forming load by PSO are shown as Fig.5 (a) and Fig.5 (b) respectively.

Table 2. Optimum parameters for the resistance wall

Variable	*R1* / mm	*R3* / mm	*B* / mm	*Hq* / mm	*a* / deg	*n* / mm
Optimum value	2. 1942	6. 5945	22. 9326	28. 3349	7. 3336	3. 1843

It can be seen from the figure that the load increases with the decrease of wear after 50 iteration times. This optimization result was reasonable in consideration of that wear was the main factor of affect on die life and the weight of wear was larger than that of load according to Eq.(8).

The parameters in Tab.2 were rounded up to Eq. (10) for the convenience of manufacturing for the forging die.

$$P=[R1\ R3\ B\ Hq\ a\ n]=[2.0\ 6.5\ 23.0\ 28.5\ 7.0\ 3.0] \tag{10}$$

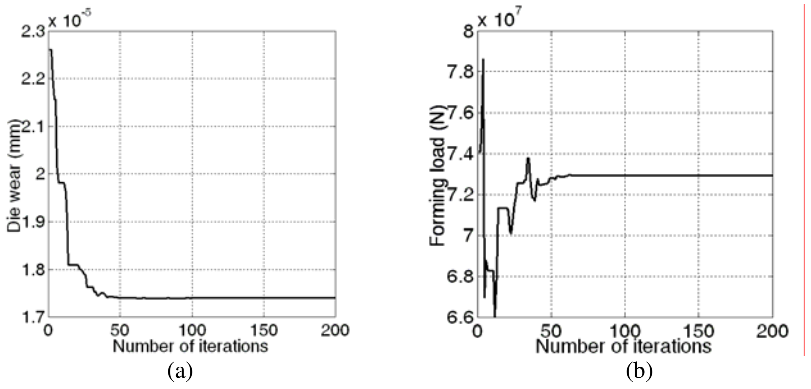


Fig. 5. (a) search traces for optimum solution of wear, (b) search traces for optimum solution of forming load

A simulation of the resistance wall with these parameters in Eq. (10) was done to test the optimization results. The simulation result is shown in Fig.6 (a) and Fig.6 (b).

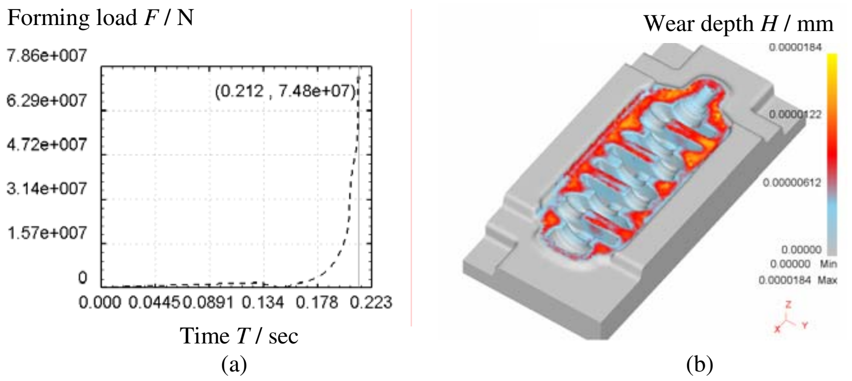


Fig. 6. (a) The simulation result of forming load, (b) The simulation result of wear

The comparison of optimum values and simulations is shown as Tab.3. The deviation of wear and load corresponding the parameters of the resistance wall according to Eq. (10) is 5.4% and 2.0% respectively and it is acceptable.

Table 3. Comparison of responses

Response	PSO	Simulation	Deviation
Wear(mm)	1.74e-5	1.84e-5	5.4%
Load(N)	7.33e+7	7.48e+7	2.0%

6 Conclusions

Taking the resistance wall of the forging die for crankshaft as the research object, the inner relation between load and wear was constructed using Kriging model with decreasing forming load and increasing life of die as the objective. The optimum parameters of the resistance wall were obtained by applying the particle swarm optimization algorithm for global optimization. It provides a kind of quantitative analysis method and basis for the design of the resistance wall.

In this paper, the Kriging model was coupled with the particle swarm optimization algorithm, which was implemented in MATLAB. The proposed method makes full use of the advantages of the Kriging model which is suitable for the computer simulation tests and takes advantage of the PSO which is simple and has a fast convergence speed than the genetic algorithms.

References

1. Samolyk, G., Pater, Z.: Use of slfet for design of flash cave with v-notched lands in a closed-die forging. Journal of Materials Processing Technology, 162–163, 558–563 (2005)
2. Kleijnen, J.P.C.: An overview of the design and analysis of simulation experiments for sensitivity analysis. European Journal of Operational Research 164, 287–300 (2005)
3. Lebaal, N., Schmidt, F., Puissant, S.: Design and optimization of three-dimensional extrusion dies, using constraint optimization algorithm. Finite Elements in Analysis and Design 45, 333–340 (2009)
4. Wang, H., Li, G., Zhong, Z.: Optimization of sheet metal forming processes by adaptive response surface based on intelligent sampling method. Journal of Materials Processing Technology 197, 77–88 (2008)
5. Sathiya, P., Aravindan, S., Haq, A.N., et al.: Optimization of friction welding parameters using evolutionary computational techniques. Journal of Materials Processing Technology 209, 2576–2584 (2009)
6. Praveen, C., Duvigneau, R.: Low cost pso using metamodels and inexact pre-evaluation:application to aerodynamic shape design. Comput. Methods Appl. Mech. Engrg. 198, 1087–1096 (2009)
7. Liu, W., Yang, Y.Y.: Multi-objective optimization of sheet metal forming process using pareto based genetic algorithm. Journal of Materials Processing Technology 208, 499–506 (2008)
8. Behrens, B.A.: Finite element analysis of die wear in hot forging process. CIRP Annals Manufacturing Technology 57, 305–308 (2008)
9. Lophaven, S.N., Nielsen, H.B., Sondergaard, J.: DACE a matlab Kriging toolbox (2005), http://www.imm.dtu.dk/~hbn/dace/
10. Raza, W., Kim, K.Y.: Shape optimization of wire wrapped fuel assembly using Kriging metamodeling technique. Nuclear Engineering and Design 238, 1332–1341 (2008)

A Scatter Search Algorithm for the Slab Stack Shuffling Problem

Xu Cheng and Lixin Tang

Liaoning Key Laboratory of Manufacturing System and Logistics, The Logistics Institute, Northeastern University, Shenyang, 110004, China
qhjytlx@mail.neu.edu.cn

Abstract. Slab Stack Shuffling (SSS) problem is a kind of warehousing operations management problem abstracted from steel industry. SSS problem is to choose appropriate slabs for hot rolling schedule with the objective of minimizing shuffles during the retrieval process. Different from previous literatures, the substitute of slabs is a set of slabs which satisfy the given order demand. The problem in this paper considers balancing the shuffles between two sub-yards and the measurement of one shuffle is also different. The problem is formulated as an integer programming model by considering above practical requirements. The complexity of the model motivated us to develop a scatter search algorithm to solve the problem approximately. Problem-oriented coding scheme and solution combination method are proposed in scatter search. The computational results tested on real data show that the shuffles are decreased by 36.9% in average compared with the manual schedule.

Keywords: Steel industry, The SSS problem, Scatter Search algorithm.

1 Introduction

Warehousing operations management problem has been a hot topic with numerous innovations in warehouse during recent years. The efficiency and effectiveness of the warehouse operation play a vital role in determining a company's competitiveness. SSS problem is a kind of such problem abstracted from slab yard of steel industry. The schematic diagram of slab yard is shown as follows:

Steel production is a multi-stage process. A slab yard serves as a buffer between the continuous casting stage and the steel rolling stage. Finished products of continuous-casting are sent to slab yard, waiting for hot-rolling. In the yard, steel slabs are stored in stacks. The whole yard is divided into two sub-yards and two cranes in each sub-yard for retrieving slabs. Shuffle is needed when picking up a slab for heating and rolling, if it is not on the top of a stack.

Properly selection of slabs can reduce shuffles during the retrieval process, improve working efficiency of cranes in the slab yard and decrease the time for feed preparation of hot-rolling. Reduction in shuffles during the retrieval process can also promote the production efficiency of the steel enterprise indirectly. The research can promote the management level of production and logistic in steel making enterprises, and decrease the logistics operation cost.

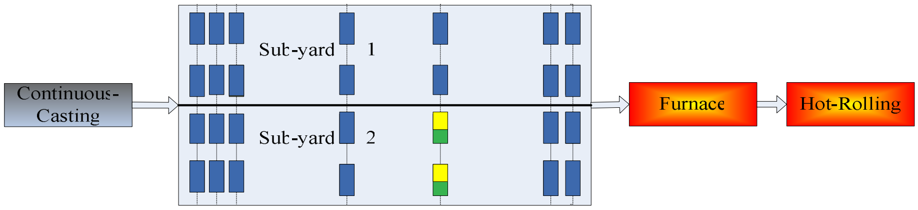


Fig. 1. Schematic Diagram of the Slab Yard

Only some attentions have been paid on SSS problem in previous literatures. Assaf et al. [1] researched on the integrated problem of production schedule generation and slab yard, focusing on the full usage of slab hot energy which did not consider the shuffles during retrieval process. Tang et al. [2] proposed a two-phase algorithm. An initial feasible solution was generated first and improved using local search. Experimental results show that the proposed algorithm yields significant better solutions than the old algorithm with an average improvement of 15%. Singh et al. [3] proposed a Parallel GA to solve the SSS problem based on Tang's model, and shuffles decrease by 6%.

The SSS problem studied in this paper consists of three major differences from the previous literatures. (1) The substitute of slabs is a set of slabs which satisfy the weight and technological demand of the given order. (2) Differences of shuffles between two sub-yards are taken into account in order to balance the workload between them. (3) The way to measure shuffles is different either. In researches before, moving a barrier slab aside and replacing it onto the original stack is considered as one shuffle. In this paper, moving a barrier slab onto a new stack is called a shuffle. Based on the above, the problem is formulated as an integer programming model by considering above practical requirements. The complexity of the model motivated us to develop a scatter search algorithm to solve the problem approximately.

The paper is organized as follows. Section 2 gives a detailed description of the problem and a mixed integer programming model. The scatter search algorithm for solving the problem is presented in Section 3. Experimental results of the scatter search algorithm tested on real data are reported in Section 4. Section 5 gives conclusions.

2 Problem Description and Mathematical Formulation

2.1 Problem Statement

Slabs stacked in the yard are raw materials for hot-rolling. Given an order, planers will choose some slabs from the yard which meet the order demand to form a hot-rolling schedule. If the chosen slab (target slab) stacked in the lower tier of a stack, the crane should move away the slabs above it in advance to make the target slab expose on the top, then 'shuffle' occurs. The increase in shuffles will lead a longer time spent on feed preparation which delay hot-rolling production and also disturb the coordinate production between continuous-casting and hot-rolling.

Considering the total weight of order demands and slab-matching constraints, SSS problem is to select the slabs for hot-rolling schedule, aiming at the minimum shuffles during retrieval process. There is a low accuracy requirement for shuffles calculation in practical production. For convenience of shuffles calculation, two assumptions are proposed followed:

Assumption 1: Barrier slabs play one time during the retrieval process. Since stacks in slab yard are enough, in this paper we suppose that barrier slabs can be always moved onto a stack which has none of target slab in. In other words, barrier slabs only play the role of barrier for one time during the whole feed preparing process which we called 'barrier slabs play one time' assumption.

Assumption 2: For target slabs in a same stack, the slab in higher tier is taken out from the yard before the slab in lower tier.

According to the characteristic of the SSS problem in this paper, the concept of 'order family' in this paper replaces 'slab family' which is reported in previous literatures. 'Slab family' is a set of slabs which are substitutable for a given slab, and ' order family' is a set of slabs which satisfy the technological requirement (width, steel-grade and weight of slab) of a given order. Generally, 'order family' is larger in size than 'slab family', which means a larger range of choice for substitute slabs. Resulting from this, there will be more chance to gain a better solution by 'order family' during the optimization process.

2.2 Mathematical Model for SSS

A mathematical model is constructed for the problem in this section.

Parameters:

Ω	Order set, $\Omega = \{1, 2, \ldots, i, \ldots, M\}$
Φ	Slab set, $\Phi = \{1, 2, \ldots, j, \ldots, N\}$
C_i	Candidate slab set for the ith order, $C_i \cap C_k \supseteq \varnothing$, for any $i, k \in \Omega, i \neq k$
W_i	Total weight of the ith order demand
φ_j	Initial stack of slab j.
D_j	The number of slabs above slab j in φ_j
R_j	The region of φ_j, $R_j = \{1, 2\}$
H_j	The weight of slab j

Decision variable X_{ij}:

$$X_{ij} = \begin{cases} 1 & \text{if slab } j \text{ is chosen for the } i\text{th order in the rolling schedule} \\ 0 & \text{otherwise} \end{cases}$$

for $i \in \Omega, j \in \Phi$.

The model of the SSS is below:

$$\min \quad k_1 \sum_{i=1}^{m}\sum_{j=1}^{n} S_{ij}X_{ij} + k_2 \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} S_{ij}X_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{n} X_{ij}H_j} + k_3 \frac{|N_1 - N_2|}{N_1 + N_2 + 1} + k_4 \sum_{i\in\Omega}\left(W_i - \sum_{j\in\Phi} X_{ij}H_j\right). \tag{1}$$

s.t.

$$\sum_{j \in C_i} X_{ij} = 1 \qquad i \in \Omega \tag{2}$$

$$\sum_{j=1}^{n} X_{ij} H_j \le W_i \qquad i \in \Omega \quad j \in \Phi \tag{3}$$

$$X_{ij} = 0 \qquad i \in \Omega, \quad j \in \Phi \setminus C_i \tag{4}$$

$$X_{ij} \in \{0,1\} \qquad i \in \Omega \quad j \in \Phi \tag{5}$$

Where,

$$\begin{aligned} & S_{ij} = D_j - \max D_{j'} - 1 \\ & j' \in R, R = \left\{ j' \middle| \varphi_j = \varphi_{j'}, D_j > D_{j'}, X_{ij'} = 1, i \in M, j' \in \Phi \right\} \end{aligned} \tag{6}$$

$$N_1 = \sum_{i=1}^{m} \sum_{j_1 \in R_1} S_{ij_1} X_{ij_1}, \quad j_1 \in R_1 = \left\{ j_1 \middle| r_{j_1} = 1, j_1 \in \Phi \right\} \tag{7}$$

$$N_2 = \sum_{i=1}^{m} \sum_{j_2 \in R_2} S_{ij_2} X_{ij_2}, \quad j_2 \in R_2 = \left\{ j_2 \middle| r_{j_2} = 2, j_2 \in \Phi \right\} \tag{8}$$

S_{ij} denotes the shuffles occur when slab j is selected for the ith order, N_1 and N_2 denote the shuffles occur in sub-yard 1 and 2 during the whole feed preparation process respectively. The objective function is minimizing the sum of four items. The first item is the total shuffles during the feed preparation, the second one denotes the average shuffles per unit weight which can measure the shuffles of schedule more exactly, the third one is to measure the differences of shuffles between two sub-yards, and the last one denotes the total differences between the total weight of each order demands and the total weight of the slabs chosen for it. Constraint (2) ensures each slab can be chosen for only one order. Constraint (3) ensures that the total weight of slabs chosen for an order should be less than the demands of that order, Constraint (4) ensures the slab can not be chosen for order i if it is not in C_i. Constraint (5) defines the range of decision variable.

Based on the description mentioned above, there are mainly two differences between the model proposed in this paper and the previous one. (1) The objective function evaluates four factors in this model and there is only one item with total shuffles in previous model. (2) Based on assumptions proposed before, there is no interaction in calculating shuffles when retrieving different slabs. For the complexity of the model, a scatter search algorithm is proposed to solve the problem.

3 Scatter Search Algorithm for the SSS Problem

Scatter search algorithm is a population-based meta-heuristic method. Different from genetic algorithm, inducement of reference set and designation of solution combination method can keep diversification and intensification of the solutions during the optimization process. The effectiveness of scatter search has been proved in solving some hard optimization problems such as linear ordering problem [5], Graph Coloring [6] and so on. Scatter search is designed to operate on a set of good solutions, called reference set, including some solutions of good objective function value and some of good diversity. The solutions in the reference set are combined to generate new solutions by combination method. A typical scatter search algorithm includes five methods: a diversification generation method, an improvement method, a reference set update method, a subset generation method, a solution combination method [7].

3.1 Coding Scheme

In the coding scheme, a chromosome was constructed by a sequence of numbers. Each position in the sequence corresponds to a slab in the rolling schedule. The value at a position in the chromosome indicates the stack information of a chosen slab, including the stack and the layer message. The slabs chosen for the same order are concentrated in the same order segment.

3.2 Initial Population Generation Method

To generate a collection of diverse initial solutions, we propose heuristic algorithm together with random generation method based on the analysis of problem property.

3.2.1 Heuristic Method

Based on the above, two rules should to be obeyed when selecting slabs for a given order. First, the physical properties (width, steel-grade, weight and so on) and chemical properties of the chosen slab should keep consistent with the technological requirement of the given order completely. Second, for any order, the total weight of chosen slabs must be less than the order demands. In the SSS problem, we take orders as a series of knapsacks, and slabs are the goods waiting to be packed. Then the SSS can be seen as a multiple knapsacks problem subjected to order weight constraints aiming at minimum total shuffles. Two cases may occur when choose slabs for a given order:
Case 1: The chosen slabs are stacked in the same stack, and stacked one by one from the top of the stack.
Case 2: The chosen slabs are stacked in different stacks.
Then we can get the following property.

Property 1: *Case 1* has fewer shuffles than *Case 2* during the retrieval process.

Proof: Obviously shuffle of *Case 1* is 0, and shuffle in *Case 2* is at least 0. If any stack has shuffles, shuffles of *Case 2* is more than case 1. □

For the *i*th order of the schedule, suppose slabs in C_i are stacked in m stacks. Let WE_k denotes the total weight of candidate slabs in stack k, and SH_k denotes the shuffles of candidate slabs in stack k. Sort m stacks by WE_k/SH_k in descending order, then a stack list $\{stack_1, stack_2,\ldots,stack_k,\ldots,stack_m\}$ is obtained. Taking stack as unit, add slabs in C_i of stacks into schedule successively until stack k such that $\sum_{j=1}^{k} WE_j \le W_i$ and $\sum_{j=1}^{k+1} WE_j > W_i$.

Then delete the chosen slabs from Φ and slab selection for order i is complete. Select slabs for the next order by the same method, until all the orders are finished.

Based on different sorting method for *stack*, we gain some other initial solutions.

3.2.2 Random Generation Method

For slab selection of the *i*th order, stack l is randomly generated. If exists unselected slab set $\prod$ in stack l , where $\prod \in C_i$, and the total weight of slabs in $\prod$ is less than permitting weight of order i, then add $\prod$ to the hot-rolling schedule and update permitting weight of order i. Otherwise generate a new stack index randomly and repeat the above process. And order i will finish choosing until a given number of random stack index are generated. Then make selections for the next order.

3.3 Solution Improvement Method

In the greedy heuristic, choosing slabs as unit for stack can result in differences between order demands and the total weight of chosen slabs. To reduce such differences, local search is executed. For weight differences in order i, add any unchosen slab in C_i whose weight is less than that differences to the schedule, until there is no differences any more or no such slabs left. This strategy can also be used to improve the solutions randomly generated.

3.4 Reference Set Generation and Update Method

This method is used to create and maintain a set of reference solutions. The reference set (*RefSet*) consists of b_1 high quality solutions and b_2 diverse ones. $RefSet1=\{x_1, x_2,..., x_{b1}\}$, $RefSet2=\{x_{b1+1}, x_{b1+2},..., x_b\}$, $|RefSet|=b=b_1+b_2$. b_1 high quality solutions are the best b solutions in P as measured by the objective value, and b_2 diverse ones are the solutions with the maximum distance from the high quality solutions. In order to find diverse solutions from *P-RefSet*, it is necessary to define a diverse measure for solutions.

Distance from solution A to solution B: For a slab in solution A, if the same slab exists in solution B, the distance value stays the same as before, otherwise add one. Check each slab, the sum of distance value is the distance from A to B.

Combination method is used to generate new solutions. If the new solution is better than anyone in *RefSet*, then add it into *RefSet* to replace the worst one to preserve the *RefSet* size.

3.5 The Solutions Combination Method

This method uses 2-element subsets for solutions combination to generate new solutions. In a chromosome, orders with the same technological requirement form an order group. For the same order group of parent solutions, the child solutions copy the order group with the smaller objective value as their solution segments. For the orders in none order group, each child solution copies its parent solution segment respectively.

3.6 The Scatter Search Algorithm for the SSS Problem

Based on the procedures mentioned above, the scatter search algorithm is organized as follows:

Step 1: Set initial population $P=\varnothing$, iteration times $IT=0$, not improved times $NR=0$, maximal iteration times $maxIT=3$, maximal not improved times $maxNR=2$;
Step 2: Generate initial solutions by initial population generation method, and save them in P;
Step 3: Improve the solutions in P by solution improvement method;
Step 4: Generate the *RefSet* including 3 good solutions and 2 diverse ones using reference set update method;
Step 5: Use 2-element subset generation method to generate subsets (*NewSubsets*) of *RefSet*;
Step 6: Select an element from the *NewSubsets* to generate a pair of new child solutions by solution combination method until all elements in *NewSubsets* are used;
Step 7: Set $IT=IT+1$ and choose 5 solutions of best objective value among new obtained solutions and solutions in *RefSet* as new elements in *RefSet*;
Step 8: Judge if the *RefSet* is updated. If the *RefSet* is not updated, set $NR=NR+1$;
Step 9: If $IT< maxIT$ go to Step 5; if $IT = maxIT$ or $NR=maxNR$, stop.

4 Computational Experiments

The scatter search algorithm is implemented using C++ language and tested on a Pentium IV PC with 3.00GHz CPU and 512MB memory. To prove the reliability and feasibility of the scatter search algorithm, five groups of real data collected from a steel enterprise have been tested. The shuffles of real data (manual result in industrial actuality) and the shuffles of the optimized solution gained from the scatter search are calculated under the same criterion. The comparison results gained from the experiments are shown as follows:

The results show that the shuffles of the scatter search algorithm are decreased by 36.9% according to the manual schedule. Since the SSS is an off-line problem, the average runtime 37 seconds of the scatter search algorithm is acceptable.

Table 1. Computational Experiments

Index	Shuffles			
	Manual	Scatter Search	Improvement	Run Time
1	51	29	0.431	61
2	59	52	0.119	41
3	75	43	0.427	11
4	33	19	0.424	24
5	63	35	0.444	18
Average			0.369	37

5 Conclusions

Considering the order demands and slab-matching constraints, SSS problem is to select slabs for hot-rolling schedule, aiming at minimizing the shuffles during the access and charging operations. We propose an integer mathematical model for the SSS problem and adopt scatter search algorithm to solve it. To guarantee diversity of solutions, heuristic and random generation methods are applied to generate initial solutions. In scatter search algorithm, a coding scheme based on stack information and solution combination method are proposed to ensure the feasibility of the obtained solutions. The experiment results with practical data show that the shuffles of solutions obtained by proposed method have been decreased by 36.9% in average compared with the manual schedule.

References

1. Assaf, I., Chen, M., Katzberg, J.: Steel Production Schedule Generation. J. International Journal of Production Research 35, 467–477 (1997)
2. Tang, L.X., Liu, J.Y., Rong, A.Y., Yang, Z.H.: An Effective Heuristic Algorithm to Minimise Stack Shuffles in Selecting Steel Slabs from the Slab Yard for Heating and Rolling. Journal of the Operational Research Society 52(10), 1091–1097 (2001)
3. Tang, L.X., Liu, J.Y., Rong, A.Y., Yang, Z.H.: Modeling and A Genetic Algorithm Solution for the Slab Stack Shuffling Problem when Implementing Steel Rolling Schedules. International Journal of Production Research 40(7), 1583–1595 (2002)
4. Singh, K.A., Srinivas, Tiwari, M.K.: Modelling the Slab Stack Shuffling Problem in Developing Steel Rolling Schedules and Its Solution Using Improved Parallel Genetic Algorithms. International Journal of Production Economics 91(2), 135–147 (2004)
5. Campos, V., Glover, F., Lagua, M.: An Experimental Evaluation of A Scatter Search for the Linear Ordering Problem. Journal of Global Optimization 21, 397–414 (2001)
6. Hamiez, J.P., Hao, J.K.: Scatter Search for Graph Coloring. In: Collet, P., Fonlupt, C., Hao, J.-K., Lutton, E., Schoenauer, M. (eds.) EA 2001. LNCS, vol. 2310, pp. 168–179. Springer, Heidelberg (2002)
7. Marti, R., Laguna, M., Glover, F.: Principles of Scatter Search. European Journal of Operational Research 183(1), 100–114 (2007)

Collaboration Algorithm of FSMAS

Qingshan Li, Dan Jiang, Haishun Yun, and He Liu

Software Engineering Institute, Xidian Univ., Xi'an, China, 710071
qshli@mail.xidian.edu.cn

Abstract. To meet the requirement and solve the problems in system integration field, A Federation Structure based Multi-Agent System (FSMAS) model is proposed in this paper, with emphasis on the collaboration algorithm. This paper presents the process of partition and collaboration of the Agent tasks, the acquaintance first based on CNP algorithm in collaboration. FSMAS is applied to the development of agent-based system integration platform and tools. As a test case, a simulation system is developed which verifies the stability and efficiency of FSMAS in system integration filed.

Keyword: System integration, MAS, Task partition, Collaboration algorithm.

1 Introduction

With the quick development of Internet, how to integrate systems in the open, dynamic and hard-control environment is becoming an important challenge of software[1]. It is hoped to solve the universal problems of distributed character, heterogeneous nature and autonomy in system integration field. Component-based technique is one of the main means for integrated problem, but it can not meet the need of autonomy, intelligence, dynamic property and domain-oriented extensibility.

Agent and MAS (multi-Agent-system) techniques are becoming a new means in the field of system integration because of their characteristics of autonomy, initiative, intelligence, sociality and mobility[2], which could meet the needs of system integration well. Collaboration and communication among Agents are key problems of MAS. A lot of researches have been made, as an instance, CNP (contract net protocol)[3][4] and Blackboard Collaboration[5] are typical works in collaboration. Distributed Sensing System (DSS) of MIT and expert union system UNIONS of Chinese Academy of Science Mathematics were both typical system based-on contract net protocol. However, CNP has its problems such as frequent communication, waste of system resources and limited scope of application. Jiaopinwen, Shizhongzhi etc at home have done some researches about MAS collaboration[6][7].They used the principle of “setting post according need, competing for getting post” to collaborate. However, the evaluation system which used to evaluate the process of collaboration and the unified formal framework were lacked of in their researches. For these situations, this paper focuses on the collaboration algorithm and communication mechanism for system integration.

2 FSMAS Architecture

Generally, there are three types of MAS, they are centralized, distributed and hybrid structure. Centralized structured MAS have the problem of bottleneck controlling. And the distributed one is hard to get the behavior consistent overall. Hybrid structure, which strikes a balance between centralized and distributed, is widely used by MAS system[2][8], and FSMAS(federation-structured-based MAS) is the hybrid structure.

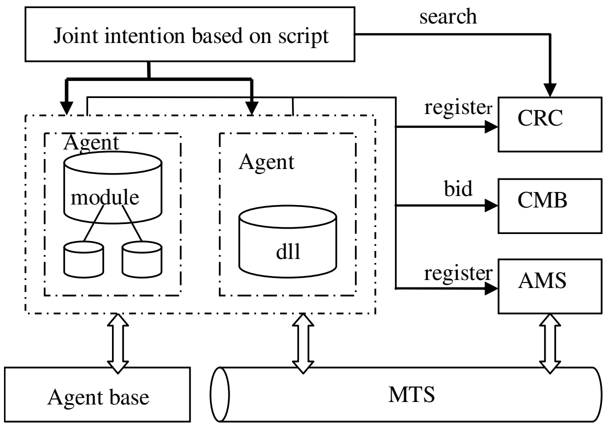


Fig. 1. Framework of FSMAS

Figure1 shows the framework of FSMAS. Several Agents exists in the frameworks and are stored in the Agent base. When an Agent is wrapped, its information would be registered to CRC (Capability Register Center) and AMS (Agent Manage Service). CMB (Common Message Blackboard) is public information blackboard. It is the bidding agency and responsible for maintaining state of tenders, sending bid notice, awarding the bid and dealing with the history tenders. There is a joint intention, which is which is involved of joint action of Agents to fulfill their common commitment.

Figure2 shows the federation structure of FSMAS. Agents are divided into Service Agent and Function Agent. Service Agent is responsible for organizing a federation,

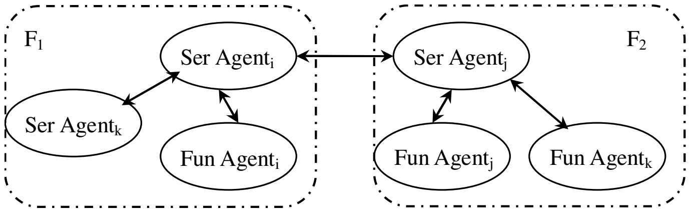


Fig. 2. FSMAS structure

which is consist of some other Agents needed, to provide a certain service. Service Agent maintains an acquaintance-list, which records the Agents it could organize. And in some situations, Service Agent would initiate a bid. Function Agent, as an acquaintance, could only provide some capability to help complete a task. It also could response a bid to complete a task.

3 Collaboration and Communication of FSMAS

3.1 Collaboration Algorithm

Partition and Cooperating Process of Tasks. To collaborate between agents, the first problem to be solved is task partition and allocation. Many scholars have made research on it, of which heuristic allocation algorithm[9] and queuing theory scheduling algorithm[10] are the typical ones. Heuristic allocation algorithm is suitable to large MAS and queuing theory scheduling algorithm are always used when the completion of the task submitted by user just need one agent.

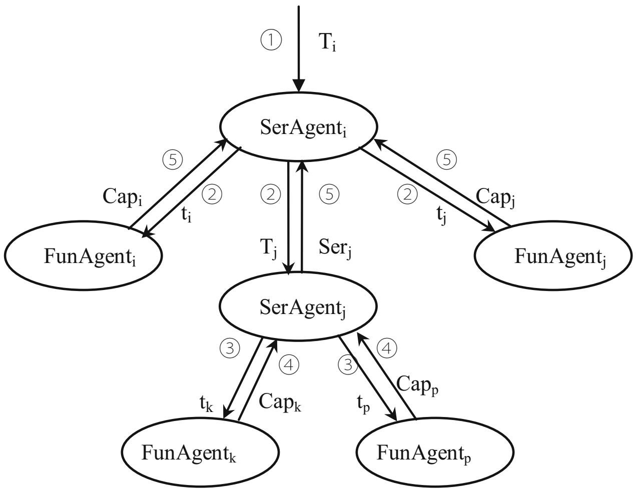


Fig. 3. Partition and cooperating process

In the field of system integration, the complexity of the task is uncertain because of the uncertainty of the task. Neither of algorithms above is used in FSMAS, instead, a more appropriate way is proposed. FSMAS allocates task of high complexity to service agents and service agents partition the complicated tasks to simple tasks and assigns the simple tasks to an appropriate function agent. Tasks are divided into atomic, record as t, and collaborative tasks, record as T. It is defined as *T= {Ti, Tk, Tk ...ti, tj, tk,…}*. Function Agent could finish atomic task while Service Agent finishing collaborative one.Figure3 shows the flow chart as an instance of partition and collaboration of an overall task. The main steps are as follows.

① *$SerAgent_i$* receive a certain task T_i;
② T_i is supposed to partitioned into t_i, t_j and T_j by *$SerAgent_i$*, and distributed to *$FunAgent_i$*, *$FunAgent_i$* and *$SerAgent_j$*;
③ T_j continues to be partitioned into t_k and t_p, which will be distributed to *$FunAgent_k$* and *$FunAgent_k$*;
④ *$FunAgent_k$* and *$FunAgent_k$* provide Cap_k and Cap_p to *$SerAgent_j$* , *$SerAgent_j$* organizes the capabilities to provide Ser_j

$SerAgent_i$ organizes Cap_i , Cap_j and Ser_j to provide Ser_i. This organization has been able to complete the overall task T_i.

The specific algorithm for task partition is as follows.

```
Queue PartitionTask(T_i){
Queue taskqueue;
T_i →{task_1, task_2,….task_k}
taskqueue.pushback({task_1, task_2,….});
for(int i=1; i!=end; i++) {
    If(∃task_i| task_i∈T){
   Queue subtaskqueue=PartitionTask(task_i);
      taskqueue.pushback(subtaskqueue);
  }
}
  return taskqueue;
}
```

Algorithm 1. Task partition Algorithm

Collaborative Algorithm. As the problems of CNP mentioned in the introduction, a new collaborative algorithm "acquaintance first base on CNP" is proposed in this paper. Its main idea is: service Agents choose the Agents in their acquaintance base to complete tasks first, if the Agents in acquaintance base can not complete the task, the service Agent would initiate a bid to get the Agents needed.

As Algorithm2 shows, after partitioning task, the service Agent gets the information of which Agents would be organized according to the definition files. Then, the service Agent get Agents' information in CRC, the information would show which Agents are the acquaintances of the Service Agent and which are not. If all the Agents involved are acquaintances, the service Agent would organize them and return a service; if not, then initiates a bid. When the service Agent gets all the Agents, it would organize them and return a service.

Algorithm3 shows how CMB process a bid when receives it and how the Agents wins the bid. When a service Agent initiates a bid, it would send a message about the bid to CMB, CMB would analyze the message and get some information about the bid after it received the message, then it would query CRC according to the message and get an *AgentSet*, next, it would send *Msg* which records the bid's information to all the Agents in *AgentSet* and wait for reply. CMB would send the information

WinMsg of first replying Agent to the initiator Agent. And to every one in *AgentSet*, when they got the message, they would check their stat first and send *WinMsg* if the stat is idle. Then it would be waiting for invoking.

Using "acquaintance first based on CNP" collaborative algorithm has the advantages as follows: 1. choosing acquaintance to collaborate could greatly enhance the efficiency of the collaboration; 2. the bidding mechanism could ensure the completion of tasks and the success rate of coordination; 3. adding acquaintance automatically after bidding, which prepares for the next collaboration.

```
SerAgentExeTask($T_i$){
taskqueue=PartitionTask($T_i$);
taskqueue×$CRC$→Agentqueue;
    if(($\forall$agent| agent$\in$Agentqueue)→(agent$\in$$Ab$))
      Ser={Ser$_1$,…Ser$_m$, …Cap$_1$, …Cap$_n$}
    |Ser$_i$$\in$Serb$_i$, Cap$_i$$\in$Capbi;
    else {
    Bidagent| agent$\in$Agentqueue && agent$\notin$ $Ab$;
    Bidagent×$BidMod$→bid;
    Initiate(bid);
    Wait();
    Bidagents=Accept(bid);
    Ser={Ser$_1$,…Ser$_m$,…Cap$_1$,…Cap$_n$,Bidagent$_1$,…Bidagent$_k$}
       |Ser$_i$$\in$Serb$_i$,Cap$_i$$\in$Capbi,Bidagent$_i$$\in$Bidagents;
    }
    return Ser;
}
```

Algorithm 2. Service Agent execute a task

```
Bid(bid)
{
  bid×CRC→AgentSet;
  for(Agent$_i$=First(AgentSet);Agent$_i$!=Last(AgentSet);
i++)
   SendMsg(Agent$_i$, Msg);
   Wait();
   Send(WinMsg) | WinMsg is sent to initiator;
  }
WinBid(bid)
{
  if(state==idle)   SendMsg(WinMsg);
}
```

Algorithm 3. Bidding Algorithm

4 Experiment

We have developed an integration platform based on MAS technology and the collaboration algorithm of FSMAS is used. In order to verify the performance of the collaboration algorithm, a typical simulation system is integrated in an experiment. The main modules of the system are wrapped into Agents and needed to be integrated into a running system. In the same computer environment, the system is integrated in two different ways. In the first way, MAS is distributed structure and using CNP to collaborate; in the second way, MAS is FSMAS and using acquaintance first based on CNP algorithm; and FSMAS and acquaintance first based on CNP algorithm are used in the last way. Figure4 shows the efficiency of the system integrating process.

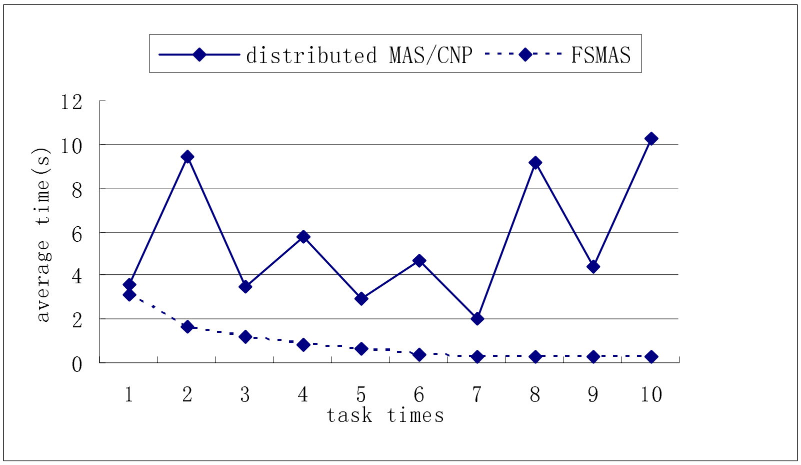


Fig. 4. Comparison of distributed MAS based on CNP and FSMAS based on "acquaintance first"

As Figure4 shows, the solid lines represents the average time used by distributed MAS based on traditional CNP in an integration task, and the dotted lines represents FSMAS'. It can been seen that, when there was only one task, the integration efficiency of two type is not far-off, but with the increasing number of tasks, FSMAS significantly reflected the advantages of not only high operation efficiency but also the much better system stability.

For distributed MAS, all the tasks are completed by bidding. The time for every task used by integration platform is the sum of bidding time, tender dealing time, Agents cooperating time and running time. While the FSMAS only bid at first, as long as the service Agent has certain acquaintances, following tasks are completed through collaboration with acquaintances. The total time for the tasks is the sum for getting acquaintance time and running time. The FSMAS' architecture also helps to improve the efficiency. Service Agents are response for tasks partitioning and organizing. And Function Agents are just response for completing tasks. For distributed

MAS, each Agent would partition and complete tasks and use much more time. Therefore the average efficiency is improved greatly.

5 Conclusion

In order to meet the requirement and to solve the problem of system integration, this paper proposed FSMAS, including the MAS structure, and focus on the collaboration algorithm. By the comparison of the experiments, we can see that FSMAS has some advantages in system integration field, which as follows: (1) system integration is simple, good scalability; (2) collaboration algorithm between the Agents reflects the intelligence and sociality, it also balances the system load; (3) it also significantly reduces system overhead and improves operating efficiency and stability. Of course, there are some disadvantages in FSMAS, such as problems about security management, strategy to improve the speed of service response and so on.

Acknowledgment. This work is supported by the National Pre-Research Project of the 'Eleventh Five-Year-Plan' of China under contract number 513060401 and 'the Fundamental Research Funds for the Central Universities'.

References

1. Li, C.Y., Li, Y., Wu, J., Wu, Z.H.: A Service-Oriented Software Model Supporting Dynamic Evolution. Chinese Journal of Computers 29(7), 1020–1028 (2006)
2. Zlotkin, G., Rosenschein, J.S.: Cooperation and Conflict Resolution Via Negotiation among Autonomous agents in Non-cooperative Domains. IEEE Trans on Systems, Man, and Cybernetics 21(6), 1317–1324 (1991)
3. Smith, R.G.: The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. IEEE Trans. On Computer c-29(12) (1980)
4. Davis, R., Smith, R.G.: Negotiation as a Metapher for Distributed problem Solving. AI 20(1), 63–109 (1983)
5. Liu, X.J., Liu, S.P., Zhang, J.: Construction of Communication and Coordination Mechanism in Amulti-Agent System. Machine Design and Manufacturing Engineering (2002)
6. Jiao, W.P., Shi, Z.Z.: Modeling Dynamic Architectures for Multi Agent System. Chinese Jouranal of Computers 23(7), 732–737 (2000)
7. Jiao, W.P., Shi, Z.Z.: A Study of Cooperation Processes in Multi Agent Systems. Journal of Computer Research and Development 37(8), 904–911 (2000)
8. Wang, J.S., Cui, S.G.: Multi-Agent Technique and its Application. Computer Engineering and Applications 18, 61–62 (2003)
9. He, Y.X., Song, W.X., Peng, F., Chen, X.M.: A Heuristic algorithm to solve the task partition problem in MDOCEM. Mini-Micro Systems 20(12), 893–895 (1999)
10. Hu, S.G., Liu, X.Y., Zhong, Y.X., Wang, K.H.: Research on the coordination algorithms in multi-Agent systems for automatic abstracting. Journal of Computer Research and Development 38(11), 1302–1309 (2001)

GPU-Based Parallelization Algorithm for 2D Line Integral Convolution

Bo Qin, Zhanbin Wu, Fang Su, and Titi Pang

Department of Computer Science & Technology, Ocean University of China, Qingdao, China 266100
qinbo@ouc.edu.cn, wzbluoy@163.com

Abstract. GPU (Graphics Processing Unit) technology provides an efficient method for parallel computation. This paper will present a GPU-based Line Integral Convolution (LIC) parallel algorithm for visualization of discrete vector fields to accelerate LIC algorithm. The algorithm is implemented with parallel operations using Compute Unified Device Architecture (CUDA) programming model in GPU. The method can provide up to about 50× speed-up without any sacrifice on solution quality, compared to conventional sequential computation. Experiment results show that it is useful for in-time remote visualization of discrete vector fields.

Keywords: GPU, parallel computation, LIC, CUDA.

1 Introduction

Discrete vector field data visualization is one of the most challenging issues in scientific visualization as well as in animation and special effects. There are broad prospects for development and application areas. Line integral convolution that is a vector field visualization method based on texture computes the white noise image by convolution filter along the flow line. LIC method shows the direction of streamlines clearly, reflects the structure of the entire discrete vector field, and overcomes the confusion caused by the arrows or stream lines in the visualization; it is extremely important significance for discrete vector field visualization.

LIC is originally proposed by Brian Cabral and Leith Leedom [1] at the 1993 SIGGRAPH conference. However, the algorithm is more time-consuming, so people made a series of improved algorithm based on LIC: Deflev Stalling [2] put forward the Fast LIC method; it employed simple box filter kernels only and minimized the total number of stream lines to be computed. So it reduced computational costs by an order of magnitude compared to the original algorithm. Lisa Forssell [3] proposed Surface LIC. This algorithm extended to the curved grid from the 2D Cartesian grid, and also solved issues such as: the calculation stream lines on curved surface, selection about the input texture, texture mapping etc. Volume LIC proposed by Victoria Interrant [4] used 3D visibility-impeding "halos" to intuitively indicate the presence of depth discontinuities between contiguous elements in a projection and thereby clarify the 3D spatial organization of elements in the flow. Hege [9] extended FastLIC from the box-shaped filter kernels to the piecewise polynomial filter kernels. Han-wei Shen [5] raised algorithm that produced time-accurate, highly coherent flow animations to highlight global features in unsteady flow fields. The algorithms above mainly focused on using the correlation between adjacent points to improve the efficiency, and the result seemed to be well, but there was still more time-consuming.

With the development of computer hardware, especially, accessing to the era of high-performance computing, LIC was parallel computed by cluster. It can gain high solution quality with fast speed, but it concurrently uses a plurality of computers and entails significant network bandwidth. However, GPU-based parallelism is much more cost-effective. The cost of a GPU card is not expensive and the local parallelism obviously causes no overhead on network traffic.

In this paper, a GPU-based method for visualization in discrete vector fields is presented. LIC can be implemented with parallel operations using CUDA programming model in GPU. The method can provide up to about 50× speed-up, compared to conventional sequential computation. The structure of this paper is as follows: In section 2, describe the status of GPU and CUDA. In section 3, introduce the LIC algorithm. The implementation process of LIC in GPU is detailed discussed in section 4. In section 5, analyze the experimental results. Finally, section 6 some conclusions are given and future work is outlined.

2 GPU and CUDA

2.1 GPU

The GPU referring to the commodity off-the-shelf 3D graphics card is specifically designed to be rapidly fast at processing large graphics data sets (e.g. polygons and pixels) for rendering tasks [6,10]. A GPU is usually made up of a body of multiprocessors and each multiprocessor consists of multiple processing units (or ALUs). Each ALU get in touch with a band of local registers. A control unit and some shared memory are shared by all the ALUs of a multiprocessor. There may be over one hundred ALUs in a typical GPU [7, 12].

The kernel function that applied on GPU by the software programs is executed on multiple ALUs in the basic unit of thread. A two-level hierarchy is formed in the threads [11]. There are multiple threads in a warp and a band of warps constitute a block. The global memory for a GPU is usually a DRAM off the GPU chip but on the same board as GPU. It is very high for latency of data to access on global memory and load kernel function. In order to improve the efficiency of GPU usage, the data need to be loaded infrequently and the overall program runtime also need to be dominated by ALUs [7].

2.2 CUDA

CUDA is a software platform for massively parallel high-performance computing on powerful GPUs [14]. The initial GPU computing environment was first called CUDA in 2006 with the launch of the GeForce 8800. GPU computing with CUDA is a new method to transforming the GPU into a massively parallel processor. Hundreds of processors simultaneously communicate and cooperate to solve complex computing problems in GPU computing [13].

CUDA programming model is one of multi-threaded programming models. Standard C/C++ are extended by the CUDA programming model with a minimalist set of parallel programming abstractions, namely a hierarchy of threads, shared memories, and barrier synchronization [15,17]. A sequential host program and one or more parallel kernels which can be executed by the host program on a parallel device constitute a CUDA program. Typically, the host program executes on the CPU and the parallel kernels execute on the GPU [16].

3 Line Integral Convolution

3.1 Main Ideas

It is detailed illustrated this algorithm in Fig.1. Taking a discrete vector field and a white noise image as the input, the algorithm uses a low pass filter to perform one-dimensional convolution on the noise image based on fourth-order Runge-Kutta. The convolution kernel follows the paths of streamlines originating from each pixel in both positive and negative directions. Color image is generated using the weight of the vector field. Output texture and color image blending together generate the final picture. As a result, the output intensity values of the LIC pixels along each streamline are strongly correlated so the global features of the flow field can be easily visualized.

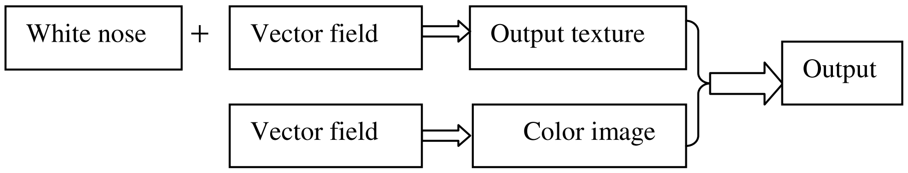


Fig. 1. LIC diagram

Figure 2 shows the visual effects of a two dimensional velocity field using LIC algorithm. Since colors represent different speed, it can clearly see the weight characteristics of data field from the image.

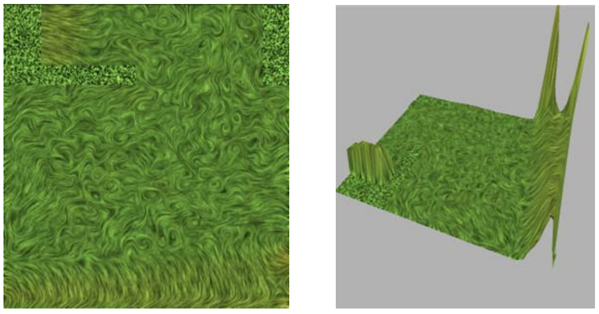


Fig. 2. Flow image generated using LIC algorithm

3.2 Fourth-Order Runge–Kutta Algorithm

Runge-Kutta method is a digital Computing method using differential equation computation [8]. It is widely used for curve fitting due to its precision, stability, and easy programming. So the streamlines are fitted by the fourth-order Runge-Kutta method. The approximate formula as follows (1):

$$Y_{k+1} = Y_k + \sum_{i=1}^{N} C_i K_i \tag{1}$$

$$\textit{Where } K_1 = h * f(x_k, y_k) \ , \ K_i = h * f(x_k + \alpha h, y_k + \sum_{j=1}^{i-1} K_j \beta_{ij})$$

$$h = x_{i+1} - x_i \ , \ 0 \le \alpha, \beta, C_i \le 1 \ , \ i = 2,3 \cdots N.$$

The function $f(x, y)$ is the value of the (x, y) .The h is the horizontal distance between adjacent nodes. α, β and C_i are the parameters.

When N = 4 in the formula (1), it can get the fourth-order Runge-Kutta formula (2):

$$Y_{k+1} = Y_k + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4) \tag{2}$$

$$\textit{Where: } K_1 = f(x_k, y_k) \ , \ K_2 = f(x_k + \frac{h}{2}, y_k + h\frac{K_1}{2})$$

$$K_3 = f(x_k + \frac{h}{2}, y_k + h\frac{K_2}{2}) \ , \ K_4 = f(x_k + h, y_k + hK_3)$$

The local truncation error of this formula can reach to $O(h^5)$. Selecting of the parameters constants in formula, it can get the precision as high as possible.

4 Implementation Process

4.1 Feasibility Analysis

It is used that four-order Runge-Kutta convolution for each pixel in the LIC algorithm, so the result only contributes to one pixel and is no effect on other pixels. It is not difficult to see that the Line Integral Convolution for implementation of the different pixels is independent of each other. Obviously, it can transform the sequential algorithm into parallel computing algorithms. Therefore, we can allocate the Line Integral Convolution for different pixels into multiple threads without worrying interdependency. In GPU-based parallel computing, Line Integral Convolution for different pixels can be processed at the same time because there is no inter-dependency among their computations.

4.2 Implementation in GPU

Figure 3 shows LIC implementation process in GPU. The main idea of the algorithm is proposed in chapter 3.1. Now, it focuses on a detailed description implementation of the algorithm in GPU.

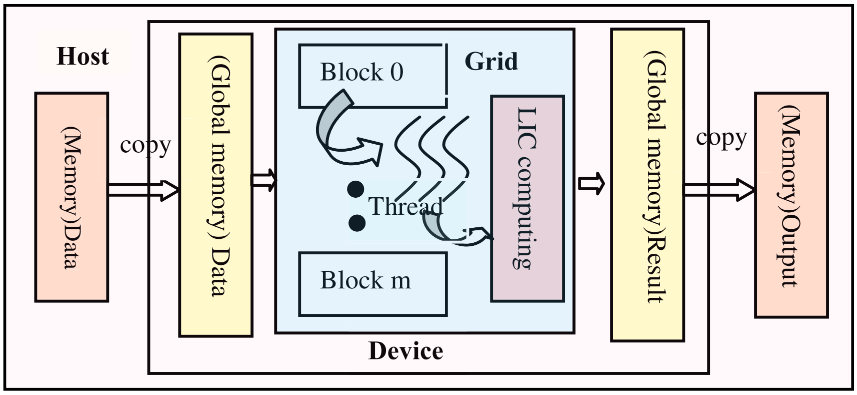


Fig. 3. LIC implementation process graph

Data is copied from host memory to device memory. The GPU global memory provides memory coalescing mechanism for improved access efficiency. The base data in the global memory is saved when the application program is running, in order to reduce the access latency and have a greater chance to coalesce the global memory from different threads in a warp.

Allocate computing task for each thread using domain decomposition method. As shown in Figure 4: here it can be imaged that the block index is i, the thread index is j, the total number of blocks is m, the number of treads in each block is n. The output image can be split horizontally into m and vertically into n, so the total image-grids size is $m \times n$. Therefore, the task of this thread is the image-grid which number is (i, j).

The results are copied from device memory to host memory after the kernel function is called. Loading the kernel function to GPU can be very time-consuming. Since the computation operations for all pixels are the same, the computation instructions are loaded only once and apply them to all of the pixels.

To reduce the processing time during the data transformation between host and device, we assign data to access respectively only once at begin and at end. This arrangement has positive impact on the performance, because the data transformation between host and device takes a large portion of the total execution time.

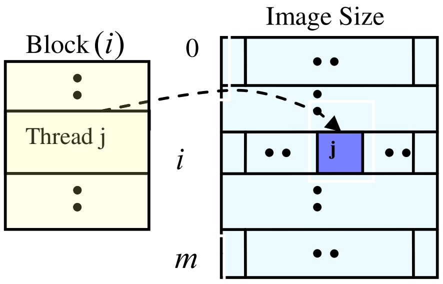


Fig. 4. Task of the *j* th thread in the *i* th block

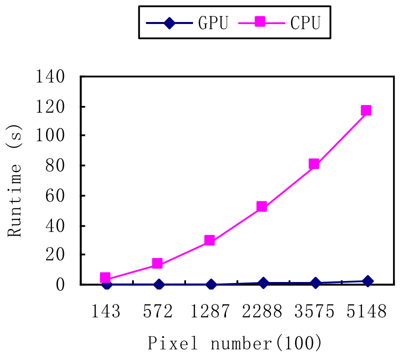


Fig. 5. Runtime comparison between GPU and CPU

5 Results

Experimental PC has one Intel R_Pentium （R） Dual E2140 processor with NVIDIA GeForce 9500 GT. The OS was Windows XP Professional with NVIDIA graphics driver version 178.28. For CUDA program compilation, Microsoft Visual Studio 2005 Professional Edition and CUDA 2.0 SDK were used.

In Figure 5, comparison of the runtimes in between GPU and CPU environment is shown. In the case of the same parameters in LIC, the runtimes are tested in two kinds of environments by changing the number of pixels. From the figure5, it can find out that the runtime increased dramatically in CPU, while the growth of computational time is not very clear in GPU. Figure 6 shows the speedup between GPU and CPU. It is not difficult to see that this parallel technique provide runtime speedup ration from 28× to 55× with the increasing of pixel number. It can be seen that the speedup tends to be steady when the pixel number grows to a certain level. Therefore, it can get the

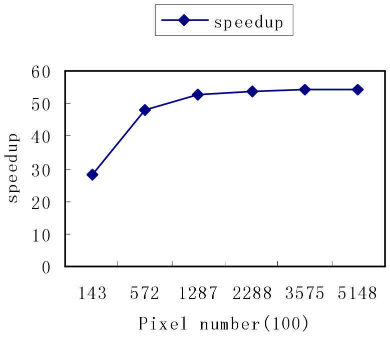


Fig. 6. The speedup ration between GPU and CPU

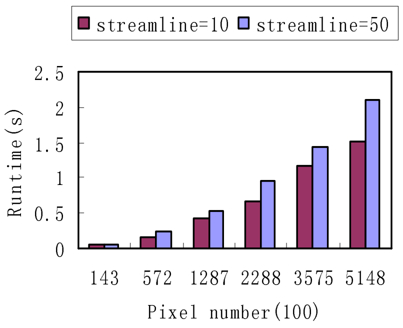


Fig. 7. Runtime with different streamline length on GPU

conclusion: the speedup ration of the LIC algorithm running in the GPU is about 50× when the calculation is large.

Figure 7 shows runtime in GPU when streamline length is different in Runge-Kutta method. Streamline length is an important factor to determine the sequential LIC algorithm execution time. However, it rarely affects the parallel operation. This is because that streamline length only increases the processing time for each thread, but these threads are concurrently executed. So, it can infer: this parallel algorithm is efficient implementation on the GPU.

6 Conclusions and Future Work

In this paper, it designed a parallel LIC with GPU computation to solve the discrete vector field data visualization problem. Although there is still much future work to go related to this study, the results were promising, showing a speedup ration about 50 times, compared to the Intel R Intel R_ Pentium（R）Dual E2140 processor. For the requirement of in-time remote visualization of discrete vector field, it also should make some proposals for future GPU configuration as it applies to LIC. Shared memory would be used and it will greatly improve the performance and the applicability of GPU.

References

1. Cabral, B., Leedom, C.: Imaging Vector Fields Using Line Integral Convolution. In: SIGGRAPH 1993 Conference Proceedings, ACM SIGGRAPH, pp. 263–270 (1993)
2. Stalling, D., Hege, H.C.: Fast and resolution-independent line integral convolution. In: Proceedings of SIGGRAPH 1995, pp. 249–256 (1995)
3. Forssell, L.: Visualizing flow over curvilinear grid surfaces using line integral convolution. In: Proceeding of IEEE Visualization 1994, pp. 240–247 (1994)
4. Interrante, V., Grosch, C.: Strategies for effectively visualizing 3D flow with volume LIC. In: Proceedings of IEEE Visualization, Phoenix, Arizona, pp. 421–424 (1997)
5. Shen, H.K., Kao, D.L.: Uflic: a line integral convolution algorithm for visualizing unsteady flows. In: Proceeding of IEEE Visualization, Phoenix, Arizona, pp. 317–322 (1997)
6. Owens, J., Luebke, D., Govindaraju, N., Harris, M., Krüger, J., Lefohn, A., Purcell, T.: A Survey of General-Purpose Computation on Graphics Hardware. In: Proceedings of Eurographics (2005)
7. Liu, Y., Hu, J.: GPU-Based Parallelization for Fast Circuit Optimization. ACM, New York (2009)
8. Mathews, J.H.: Computer Derivations of Numerical Differentiation Formulae. Mathematics Education in Science and Technology 34(2), 280–287 (2003)
9. Hege, H.C., Stalling, D.: Fast LIC with piecewise polynomial filter kernels. In: Mathematical Visualization—Algorithms and Application via, pp. 259–314. Springer, Heidelberg (1998)
10. Vineet, V., Harish, P., Patidar, S., Narayanan, P.J.: Fast Minimum Spanning Tree for Large Graphs on the GPU. The Association for Computing Machinery, August 1-3 (2009)
11. Tsutsui, S., Fujimoto, N.: Solving Quadratic Assignment Problems by Genetic Algorithms with GPU Computation: A Case Study, July 8-12 (2009)

12. Tzeng, S., Wei, L.-Y.: Parallel White Noise Generation on a GPU via Cryptographic Hash, February 15-17. ACM, New York (2008)
13. Buck, I.: GPU Computing: Programming a Massively Parallel Processor. IEEE, Los Alamitos (2007)
14. NVIDIA: Solves the GPU Computing Puzzle (September 2009)
15. Robilliard, D., Marion, V., Fonlupt, C.: High Performance Genetic Programming on GPU. ACM, New York (June 19, 2009)
16. Garland, M.: Sparse Matrix Computations on Many core GPU's, June 8-13. ACM, New York (2008)
17. NVIDIA Corporation. NVIDIA CUDA Programming Guide, Version 2.0 (November 2007)

Biogeography Migration Algorithm for Traveling Salesman Problem

Hongwei Mo[1,2] and Lifang Xu[3]

[1] National Defence Key Laboratory of Autonomous Underwater Vehicle Technology
[2] Automation College, Harbin Engineering University, 150001 Harbin, China
honwei2004@126.com
[3] Engineering Training Center, Harbin Engineering University, 150001 Harbin, China
mxlfang@163.com

Abstract. Biogeography-based optimization algorithm(BBO) is a new kind of optimization algorithm based on Biogeography. It is designed based on the migration strategy of animals to solve the problem of optimization. In this paper, a new algorithm-Biogeography Migration Algorithm for Traveling Salesman Problem(TSPBMA) is presented. Migration operator is designed. It is tested on four classical TSP problems. The comparison results with the other nature inspired optimization algorithms show that TSPBMA is a very effective for TSP combination optimization. It provides a new way for this kinds of problem.

Keywords: Biogeography, Biogeography-based optimization, Biogeography migration algorithm, Traveling salesman problem.

1 Introduction

In recent years, we have seen that many algorithms inspired by natural phenomenon or mechanisms. In this paper, we mainly focused on such a new kind of algorithm, which is called biogeography based optimization, which is inspired by the science of biogeography. It is very interesting in that it mimics the migration process of animals to design method for solving engineering problems, especially optimization. The science of biogeography can be traced to the work of nineteenth century naturalists such as Alfred Wallace [1] and Charles Darwin [2]. In the early 1960s, Robert MacArthur and Edward Wilson began working together on mathematical models of biogeography. Since their distinct work, biogeography has become a major area of research[3].Mathematical models of biogeography describe how species migrate from one island to another, how new species arise, and how species become extinct. The term "island" here is used descriptively rather than literally. That is, an island is any habitat that is geographically isolated from other habitats.

In view of this, Simon presented the first paper on biogeography inspired algorithm for engineering[4], which is called biogeography based optimization(BBO). In his creative work, he merged the burgeoning field of biogeography with engineering in order to see how the two disciplines can be of mutual benefit. Although the idea of application of biogeography to engineering is similar to those nature inspired algorithms mentioned above, it has completely different mechanisms and process from those ones. It is again to prove the great power of nature. In the past two years, Simon and the other authors had published several papers about BBO. In the first paper on BBO, Simon introduced the main idea of how to use biogeography to design an optimization algorithm and gave us the basic definitions, steps of algorithms. The experiments results showed that BBO is indeed effective in solving these problems. In[5], Simon simplifies the original BBO in order to analyze its theory. They present a simplified version of BBO and then analyze its population using probability theory. Their analysis provides approximate values for the expected number of generations before the population's best solution improves. In[6], they develop a Markov analysis of BBO, including the option of elitism. Their analysis gives the probability of BBO convergence to each possible population distribution for a given problem. Analytical comparisons on three simple problems show that with high mutation rates the performance of GAs and BBO is similar, but with low mutation rates BBO outperforms GAs. In[7],Simon et al. propose a novel variation to biogeography-based optimization (BBO), which employs opposition-based learning (OBL) alongside BBO's migration rates to create oppositional BBO (O B BO). They mathematically prove that it has the highest expected probability of being closer to the problem solution among all OBL methods. In [8],in order to improve BBO, Du et al. incorporate distinctive features from other successful heuristic algorithms into BBO. F-tests and T-tests are used to demonstrate the differences between different implementations of BBOs. In[9], Ma et al. generalize the equilibrium species count in biogeography theory, explores the behavior of six different migration models in BBO. Their study shows that sinusoidal migration curves provide the best performance among the six different models. In[10], Bhattacharya et al. use BBO to solve the problem of economic load dispatch problem. In[11], the BBO is combined with quantum to produce a new kind of hybrid algorithm.BBO is not used for TSP problems since it was presented. In this paper, we use the idea of BBO to solve TSP.

The paper is organized as follows. Section II reviews the ideas of biogeography. Section III introduces the model and algorithm of TSPBMA. Section IV provides some simulation results of TSPBMA compared with other optimization algorithms for TSP. Section V presents some concluding remarks and suggestions for further work.

2 Biogeography and TSPBMA

2.1 Biogeography

In geography, geographical areas that are well suited as residences for biological species are said to have a high habitat suitability index (HSI). Biogeography is nature's way of distributing species, and is analogous to general problem solutions. Suppose that we are presented with a problem and some candidate solutions. A good solution is analogous to an island with a high HSI, and a poor solution represents an island with a low HSI. High HSI solutions resist change more than low HSI solutions. By the same token, high HSI solutions tend to share their features with low HSI solutions. The shared features remain in the high HSI solutions, while at the same time appearing as new features in the low HSI solutions. Poor solutions accept a lot of new features from good solutions. This addition of new features to low HSI solutions may raise the quality of those solutions. The immigration and emigration curves shown in Fig. 1 as straight lines are a simple model. It illustrates a model of species abundance in a single habitat [12]. The immigration rate λ and the emigration rate μ are functions of the number of species in the habitat. The equilibrium number of species is S_0, at which point the immigration and emigration rates are equal [3][13].

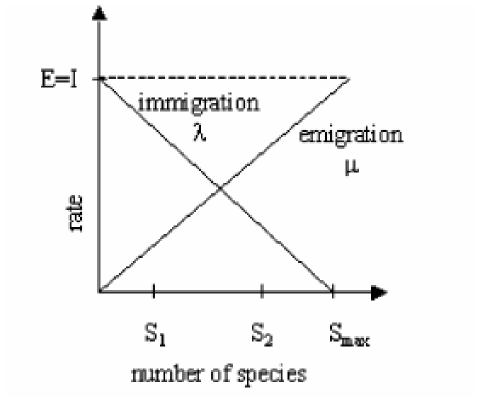


Fig. 1. The model of immigration rate and emigration rate of biology[4]

2.2 TSPBMA Model

Now, consider the probability P_s that the habitat contains exactly S species. P_s changes from time t to time $t+\Delta t$ as follows[1]:

$$P_s(t+\Delta t) = P_s(t)(1-\lambda_s\Delta t-\mu_s\Delta t)+P_{s-1}\lambda_{s-1}\Delta t+P_s\mu_{s+1}\Delta t \tag{1}$$

where λ_s and μ_s are the immigration and emigration rates when there are S species in the habitat. This equation holds because in order to have S species at time , one of the following conditions must hold:

1) There were S species at time t, and no immigration or emigration occurred between t and $t+\Delta t$;
2) There were $S-1$ species at time t, and one species immigrated;
3) There were $S+1$ species at time , and one species emigrated.

Δt is small enough so that the probability of more than one immigration or emigration can be ignored.Taking the limit of (1) as $\Delta t \rightarrow 0$ gives equation (2) shown at the bottom of the page.

Define $n = S_{\max}$ and $P = [P_0, P_1 ... P_n]^T$, for notational simplicity. Now, the $\dot{P}_s$ equations is arranged (for $S = 0,1,...n)$) into the single matrix equation

$$\dot{P} = AP \tag{2}$$

where the matrix A is given as (6). For the straight line curves shown in Fig. 1, we have

$$\mu_k = \frac{Ek}{n} \tag{3}$$

$$\lambda_k = I\left(1 - \frac{k}{n}\right) \tag{4}$$

$$\dot{P} = \begin{cases} -(\lambda_s + \mu_s)P_s + \mu_{s+1}P_{s+1}, S = 0 \\ -(\lambda_s + \mu_s)P_s + \lambda_{s-1}P_{s-1} + \mu_{s+1}P_{s+1}, 1 \le S < S_{\max} - 1 \\ -(\lambda_s + \mu_s)P_s + \lambda_{s-1}P_{s-1}, S = S_{\max} \end{cases} \tag{5}$$

$$A = E\begin{bmatrix} -(\lambda_0+\mu_0) & \mu & 0 & \dots & 0 \\ \lambda_0 & -(\lambda_1+\mu_1) & \mu_2 & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \lambda_{n-2} & -(\lambda_{n-1}+\mu_{n-1}) & \mu_n \\ 0 & \dots & 0 & \lambda_{n-1} & -(\lambda_n+\mu_n) \end{bmatrix} \tag{6}$$

It is easy to prove that the steady-state value for the probability of the number of each species is given by

$$P_k = \begin{cases} P_0 = \dfrac{1}{1 + \sum_{l=1}^{n} \dfrac{\lambda_0\lambda_1\dots\lambda_{l-1}}{\mu_1\mu_2\dots\mu_l}}, k = 0 \\ P_k = \dfrac{\lambda_0\lambda_1\dots\lambda_{k-1}}{\mu_1\mu_2\dots\mu_k\left(1 + \sum_{l=1}^{n} \dfrac{\lambda_0\lambda_1\dots\lambda_{l-1}}{\mu_1\mu_2\dots\mu_l}\right)}, 1 \le k \le n \end{cases} \tag{7}$$

2.3 TSPBMA Algorithm

In BBO,each individual is considered as a "habitat" with a habitat suitability index (HSI), which is similar to the fitness of EAs, to measure the individual. A good solution is analogous to an island with a high HSI, and a poor solution indicates an island with a low HSI. High HSI solutions tend to share their features with low HSI solutions. Low HSI solutions accept a lot of new features from high HSI solutions.In TSPBMA, each individual has its own immigration rate λ and emigration rate μ. A good solution has higher μ and lower λ, vice versa. The immigration rate and the emigration rate are functions of the number of species in the habitat.

They can be calculated as (3) and (4).Note that Eqns. 2 and 3 are just one method for calculating λ and μ. There are other different options to assign them based on different specie models [1].

The main steps of TSPBMA are shown as follows:

```
Initialize $m_{\max}$ , generation number N, generate the
initial path habitats $H^n$ randomly;
Evaluate the HSI for each path individual in $H^n$;
Sort the population from best to worst based on cost;
Initialize the generation counter t = 1;
While the halting criterion is not satisfied do
      For each path individual, map the HSI to the number
      of species;
          Calculate the immigration rate $\lambda_i$ and the
          emigration rate $\mu_i$ for each path $H(i)$;
        Calculate $P_s$ ;
          If  rand<$\lambda_i$ and $\mu_i$<sum($\mu_i$);
            Select a habitat $H(i)$ ;
             Crossover($H(i)$, $H(i+1)$);
            End if;
      End for
Calculate mutation rate $m_i$;
Mutate each habitat in $H^n$ with mutation rate $m_i$ ;
Evaluate $H^n$, update HSI;
Sort $H^n$ according to cost;
Keep the first two best individuals;
t = t + 1;
End while;
```

It begins by computing the immigration and emigration rates of each path habitat Then, path habitat modification is performed on each habitat. The emigration and immigration rates of each solution are used to probabilistically share information between habitats. In TSPBMA, it is carried out by the operation of selection and crossover. According to the model curve of λ and μ in Fig.1, we suppose that the better solution should be with lower λ and higher μ than the other emigration rates μ of the other solutions to decide whether $H(i)$ should be selected or not. If its μ value is bigger than the sum of the other habitats, then it must be better solution than the other ones. Based on the two selection steps, if $H(i)$ is selected, it is used to crossover with its neighbor $H(i+1)$ to share its good information and produce more diverse habitats. The crossover used here is regular cycle crossover. This step is different from that of GA because two parents are selected to crossover to produce offspring in the population of GA. And the crossover operation is separated from selection operation. In TSPBMA, only one habitat will be selected based on its λ and μ . And crossover and selection are in one step, that is, they cannot be separated here. This step mimics the migration phenomena happening in biology. During the migration process, information will be shared among different habitats when animals migrate from one habitat to another. We use inversion mutation on both poor solutions and good solutions.

The mutation rate m_i is the mutation rate that is calculated as

$$m_i = m_{\max}(1 - \frac{p_i}{p_{\max}}) \tag{8}$$

where $m_{\max}$ is a user-defined parameter, and $P_{\max}$ = argmax P_i, $i = 1,...NP$. With the migration operator, TSPBMA can share the information among solutions. Especially, poor solutions tend to accept more useful information from good solutions. This makes TSPBMA be good at exploiting the information of the current population. Additionally, the mutation operator tends to increase the diversity of the population.

3 Benchmark Results

In order to explore the benefits of BBO, we compared its performance on some classical TSPs with five other population-based optimization methods, including ACO,GA,PSO,IA,Fish Swarm(FS)[14]. The benchmarks are Oliver30, Eil50,and Eil75 in TSPLIB.

The parameters of TSPBMA are: habitat modification probability=1, maximum immigration and migration rates=1 for each habitat.

We did some rough tuning on each of the optimization algorithms to get reasonable performance, but we did not make any special efforts to fine-tune the algorithms. For ACO, we used the following parameters: initial pheromone value $\tau_0 = 1E-5$,pheromone update constant $Q = 20$, exploration constant $q_0 = 1$, global pheromone decay rate $\rho_g = 0.9$, local pheromone decay rate $\rho_t = 0.5$, pheromone sensitivity $\alpha = 2$, and visibility sensitivity $\beta = 6$. For the GA, we used roulette wheel selection, single point crossover with a crossover probability of 0.3, and a mutation probability of 0.1. For PSO, we used initial and ending inertia weight 0.9 and 0.3 respectively, and a social constant 0.7 for swarm interaction. For IA, we used single point crossover with a crossover probability of 0.7, and a mutation probability of 0.07. We use try number 100, sense distance 6, crowd factor σ =0.5 for FS.

In our test, the distance is considered only as the cost, to find the best path $T = (t_1, t_2, ..., t_n)$ is the same as to make following targeting function minimum

$$f(T) = \sum_{i=1}^{n-1} d(t_i, t_{i+1}) + d(t_n, t_1) \tag{9}$$

t_i is the numbering of the city which is a natural number between 1 and N , $d(t_i, t_j)$ denotes the distance from city i to city j and for symmetrical TSP $d(t_i, t_j) = d(t_j, t_i)$ .In our TSPBBO, we sort the habitats according to the cost $f(T)$,and recalculate their immigration rate λ and emigration rate μ .

Each algorithm had a population size of 50, an elitism parameter of 2, and ran 1000 generations and 20 times to get average results. Fig. 2 to Fig. 5 show the results of the simulations. For the space limited, we only give the results of TSPBMA on the KroA100 cities and also the comparison results with the other algorithms.

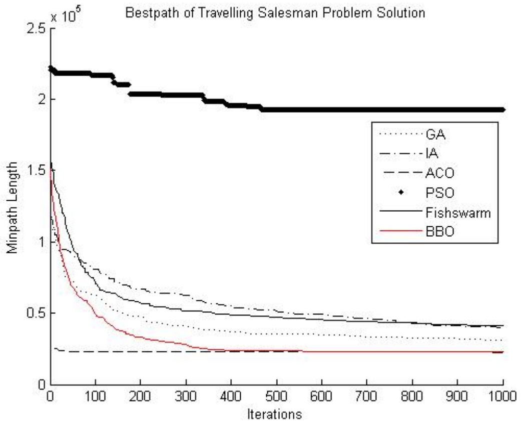


Fig. 2. The comparison results of TSPBMA with the other algorithms on KroA 130 cities

In Fig 2, the comparison results of TSPBMA with the other five algorithms are shown. We can see that TSPBMA is much better than classical GA, IA,PSO and FS both in solution quality and converging speed. It can find the best result at 400 generation. But it is worse than ACO in the converging speed.

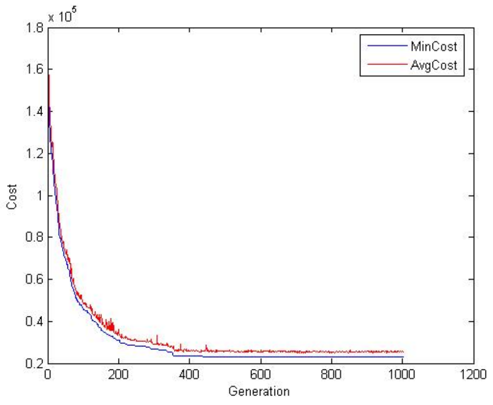


Fig. 3. The average and minimum length of TSPBMA for KroA100

In Fig 3, we give the average and min length gained by TSPBMA for KroA100 cities. In fact, for the other several problems, it has similar performance. The average results changed slowly with generations. It means that TSPBMA is stable in the searching process and can find the relative good path in each generation without too many trials in population.

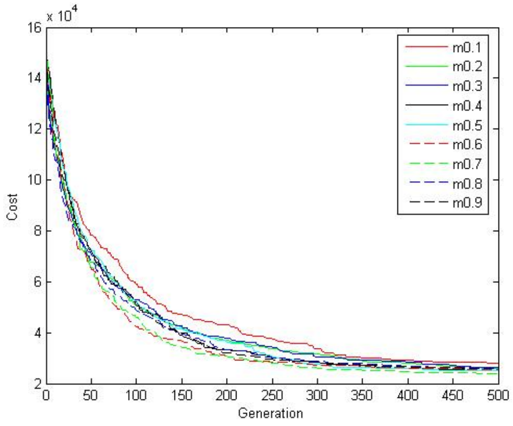


Fig. 4. The performance with the change of mutation rate of TSPBMA

The only parameter defined by user is the mutation rate($m_{\max}$). And we get good results for $m_{\max}$ =0.3-0.9. So it is not sensitive to this parameter. It can be seen in Fig. 4. This is a good performance that many other nature inspired algorithms do not own.

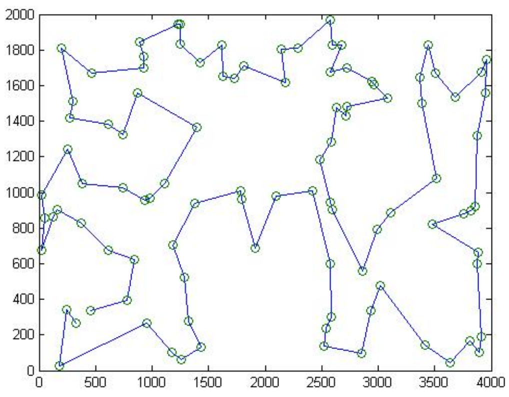


Fig. 5. The best path found by TSPBMA for KroA100 cities

In Fig 5, the best path of KroA100 cities is shown.

Table 1. TSPBMA results compared with other methods

Benchmark	TSPBMA	ACO	GA	IA	FS	PSO	Best Result
Oliver30	420	420	425	442	430	520	420
	422	422	426	453	442	552	
Eil50	425	425	N/A	453	442	530	425
	425	424	428	464	451	554	
Eil75	535	535	N/A	576	561	675	535
	536	535	545	583	572	684	
KroA100	21282	21282	N/A	22322	21923	64919	21282
	21282	21282	21761	22435	22067	66635	

The problems provided in TSPLIB for algorithm testing are in two expression forms. One is integer distance and another is real number distance between two cities. We use integer one here. The results in the parenthesis refer to average generations of 20 times running. N/A means the result cannot be gained.

In Table1, the comparison results of TSPBMA with the ACO, GA, IA,PSO and FS are shown. We report the best integer tour length. Results using GA are from[15]. The best results for these problems are included in TSPLIB. In total, the behavior of TSPBMA with respect to stability, convergence, equilibria, and other issues are better than the other nature inspired algorithms.

4 Conclusion

We have shown how biogeography, the study of the geographical distribution of biological species, can be used to derive algorithms for TSP combination optimization. This new algorithms is called TSPBMA. We have applied TSPBMA to benchmarks oliver30,eil50,eil75, KroA130 cities. And the results showed that it has better performance than most other classical nature-inspired methods, including converging speed, less parameters and robust to parameters. The results show that TSPBMA is a good method and has great potential in solving combination optimization problems including TSP. It provides a new way for researchers to solve similar problems in future. And it also proves that the idea of biogeography can be used to solve real engineering problem, such as TSP. But when compared with the proposed method, ACO was as successful as it in terms of solution quality and significantly better than it in terms of speed of convergence. In future, we will focus on improving our algorithm's performance including converging speed and solution quality on combination optimization problems and solving more complex ones in further.

Acknowledgments. The work is supported by National Nature Science foundation of China , No. 60973075，Nature Science foundation of Heilongjiang Province, China, No.F200937, and Research Fund from National Defence Key Laboratory of Autonomous Underwater Vehicle Technology.

References

1. Wallace, A.: The Geographical Distribution of Animals (Two volumes). Adamant Media Corporation, Boston (2005)
2. Darwin, C.: The Origin of Species. Gramercy, New York (1995)
3. Hanski, I., Gilpin, M.: Metapopulation Biology. Academic, New York (1997)
4. Simon, D.: Biogeography-based optimization. IEEE Transactions on Evolutionary Computation 12, 702–713 (2008)
5. Simon, D.: A Probabilistic Analysis of a Simplified Biogeography-Based Optimization Algorithm, `http://academic.csuohio.edu/simond/bbo/simplified/bbosimplified.pdf`
6. Simon, D., Ergezer, M., Du, D.: Markov Analysis of Biogeography Based Optimization, `http://embeddedlab.csuohio.edu/BBO`
7. Ergezer, M., Simon, D., Du, D.W.: Oppositional Biogeography-Based Optimization. In: IEEE Conference on Systems, Man, and Cybernetics, San Antonio, TX, pp. 1035–1040 (2009)
8. Du, D., Simon, D., Ergezer, M.: Biogeography-Based Optimization Combined with Evolutionary Strategy and Immigration Refusal. In: 2009 IEEE International Conference on Systems, Man, and Cybernetics. San Antonio, TX, pp. 1023–1028 (2009)
9. Ma, H., Chen, X.: Equilibrium Species Counts and Migration Model Tradeoffs for Biogeography-Based Optimization. In: 48th IEEE Conference on Decision and Control (2009)
10. Bhattacharya, A., Chattopadhyay, P.K.: Solving complex economic load dispatch problems using biogeography-based optimization. Expert Systems with Applications 37, 3605–3615 (2010)
11. Li, X., Guo, T.: Quantum and Biogeography based Optimization for a Class of Combinatorial Optimization. In: 2009 World Summit on Genetic and Evolutionary Computation, pp. 969–972 (2009)
12. Wesche, T., Goertler, G., Hubert, W.: Modified habitat suitability index model for brown trout in southeastern Wyoming. North Amer. J. Fisheries Manage. 7, 232–237 (1987)
13. MacArthur, R., Wilson, E.: The Theory of Biogeography. Princeton Univ. Press, Princeton (1967)
14. Li, X.L., Shao, Z.J., Qian, J.X.: An Optimizing Method Based on Autonomous Animats: Fish-Swarm Algorithm. Theory and Practice of System Engineering 11, 33–49 (2002)
15. Wang, Q., Xiong, L., Liu, H.Y.: Improved Particle Swarm Algorithm for TSP Based on the Information Communication and Dynamic Work Allocation. Asian Journal of Information Technology 5(11), 1191–1196 (2006)

An Approach of Redistricting Based on Simple and Compactness

Shanchen Pang[1], Hua He[1], Yicheng Li[2], Tian Zhou[1], and Kangzheng Xing[3]

[1] College of Information Science and Engineering, Shandong University of Science and Technology, Qingdao 266510, P.R. China
shanchenpang@sohu.com
[2] College of Engineering and Applied Sciences, State University of New York at Stony Brook, Stony Brook, NY 11794, U.S.A
[3] College of Sciences Shanghai University, Shanghai 200444, P.R. China

Abstract. Redistricting is a Clustering Problem in optimization. The optimum redistricting is a convincing argument to voters that this solution is fair. In this paper, we set up a kind of model basing on the multi-factor model of clustering of the population pots by adopting the theory of optimization and the tools of stochastic simulation. Through this method, we solve the problem of how to realize the redistribution and the judging problem. Using the statistical data and practical model, we can get the districts of the state of New York satisfied with the rules of all principles.

Keywords: simple, compactness, recursion, partisan swing, stochastic.

1 Introduction

Gerrymandering, which was coin in 1812, is the drawing of election district boundary lines for partisan advantage, to favor the majority party and incumbent politicians of all political parties [1]. In order to avoid the phenomenon and preserve the one voter-one vote fairness principle, districts should be reapportioned after the decennial census. Although the starting point of redistrcting is to protect the principle of fairness, but we cannot ensure that a party's proportion of the seats is according with its proportion of the votes. If there is a systematic tendency for a party's voters to reside in overpopulated districts(malapportionment) or in districts where their votes go "wasted" (gerrymandering), then the party will win less than its "fair" share of seats [13]. The partisan seat division should be symmetrical: that is, if one party wins y seats for *x* percent of the vote, the other party should also win approximately *y* seats for x percent of the vote. The above constraint factors are ideal partitioning, but in really partitioning districts, it is impossible to achieve the ideal phase, any state can be affected by geographical location, climate or some social aspects, things like military, prisons and so on. However reapportionment, made necessary by fidelity to democratic principles, still will bring with it gerrymandering [13].

Over the last four decades, in order to avoid the phenomenon fo gerrymandering and protect electoral fairness, many academics and politicians have put forward some algorithms, methods, models and arguments for redistricting to be automated [2]. Lijphart [3] put the 16 standard principles to regulate the principle of redistricting. In literature [4]-[8], several authors have argued that gerrymandering can eliminated by drawing districts which are maximally compact, and they consider that redistricting can be performed best by automatically optimizing a pre-specified representation function. Gary King buildes a stochastic model of the representation and bias in congressional and state legislative elections, and demonstrates that redistricting has effects in predicted directions in the short run: partisan gerrymandering biases the system in favor of the party in control, and by freeing up seats held by opposition party incumbents, increases the system's responsiveness [9]. An unified statistical method, provided by Andrew Gelman and Gary King, enables one to calculate more efficient estimates, with more trustworthy assessments of their uncertainty, than each of the separate multifarious existing measures of partisan bias [10]. We hope automated redistricting can offer a general-purpose and unbiased method for partitioning districts. However, most of the existing models of redistricting are unable to well satisfy these optimistic expectations. That is , they fail to give satisfactory results on certain types of geographical configurations. Such methods can in principle eliminate Gerrymandering by providing well defined steps and constraints.

In this paper, the Districting Problem is a geographical problem which is present in a number of geographical tasks such as school districting, design of sales territories, etc. The constraints of the Districting Problem are very similar to that of the Clustering Problem in optimization. In order to protect the principle of fairness, we need to deal with these questions below: Each district in the state must contain the same population. Drawing congressional districts with a purely baseline exercise to create the "simplest" shapes. Under the rule of "simple", we need to make a convincing argument to voters in the state that the solution is fair.

This paper is organized as follows. In section two, we put forward the definitions of "compactness" and "simple", and we design some partition models from different aspects. In section three, we design estimate model, including partisan model and compactness model. In section four, we apply partition model to New York State, and get the final figure. In section five, we do a conclusion.

2 Design of Model

Redistricting is a large mathematical problem. We can mathematize redistricting to think of it as a set partitioning problem. In this paper, the rules we complied only include that the method of redistricting is simple and each district in the state contains the same population. Each district elects only one representation, the number of districts which are partitioned is equal to the number of representations.

2.1 Relative Definations

Two basic principals govern all redistricting in the United States: all parts of a district must be contiguous and a district must be reasonably compact in shape. While contiguity is an objective criterion, compactness is subjective, and there are many ways to define it (Niemi et al.1990) [12]. But the existing definition of simple may appeal to our intuition, it does not provide a rigorous or precise standard that can be used to determine whether a districting plan is or is not simple [12], [14]. In order to understand below models and algorithm, we provide the definations of "simple" and constraint of simplest.

Defination 1. We define our "simple" in two different aspects. First, in our model, the shape of districts is the simplest. The more compact of the district is partitioned, the simpler the shape is. Secondly, the manipulation of partition is another aspect of "simple". So, we respect the geographical integrity of any city, county, or geographical region to the best of our abilities to reduce the trouble of partition.

Defination 2. When two figures have the same area, the simpler the shape is, the shorter the perimeter is. To ensure the drawing of the congressional districts to be simple enough, the perimeter of districts is the shortest. A constrained function can be given:

$$\sigma = \sum_i C_i \Big/ \mathrm{A} \text{ , } (i=1,2,3,\ldots) \text{ .} \tag{1}$$

Where C_i = The boundary length of an election district,

A = The area of the district.

We can use the value of σ to control the shape of districts. When the value is small enough, the shape which we got is simple enough.

2.2 Ideal Model

Now, we begin to discuss the model in general conditions, and provide a method of redistricting for any state of the United States. With regard to a certain state, we can get the information of whole population and how many representatives this state will have. For each district elects only one representative, we can compute the population of each district. With the amount of population of each county, the area of each district can be confirmed by us.

2.2.1 The Voter Population of Each District

In real life, it is totally impossible that every people votes to elect representative. We assume that the proportion of voter population in general population is constant in all districts. We can get the conclusion that the voter population of each state is pr/s.

Where p = the whole population of a state,

r = the proportion of voter population,

s = the number of districts.

2.2.2 Partition Districts and Partition Algorithm

Abiding by the principle of simple, we use the original boundary of counties as the boundary of districts as far as possible. It is more reasonable that people vote in their hometown. Of course, it is impossible that all boundaries of districts are the new boundary of county's. Under this condition, when the population of counties is not suitable to the requirements of the districts, we begin to utilize the boundary of villages which are smaller than counties to be the boundary of districts. To respect the integrity of any village as far as possible, people who live in one village should vote together. Otherwise, even if the population is equal in all districts, it will not happen in reality. Hence, when we are partitioning districts, the villages are thought as the smallest unit. At the same time, counties are considered as the basic units to partition districts. Firstly, we combine several counties and compute the amount of their population. Next, we estimate that whether the number of population is satisfied with the requirement of the number of population in one district. That means we change one state into a Graph Structure [11], like in fig. 1, each county is as a node. This method can be thought as a point cluster.

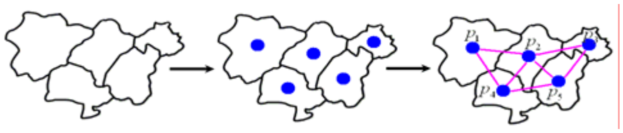


Fig. 1. Change one state into a Graph Structure

Next, we take one state as an example, and give the algorithm of method.

Algorithm 1. partition algorithm

```
step 0  choose one of the downmost counties as the
Current County (C.C) as the starting position of
partitioning, x is the population of the county;
step 1  compare x with pr/s:
  step 1.1  if x<pr/s, search counties which is abutting
the C.C, then goto step2;
  step 1.2  if x>pr/s, choose a county which lies left
underneath of the C.C;
  step 1.3  if x=pr/s, choose this county ,C.C, as a
district;
step 2  it means that the population of the town $y_j$: sum
= sum+ $y_j$;
step 3  compare the sum with pr/s:
   step 3.1  if sum<pr/s, then go to step 1.1;
   step 3.2  if sum=pr/s, then go to step 4;
   step 3.3  if sum>pr/s, choose the nearest town in the
counties which are abutting C.C , it means that the
population of the town $y_k$ : sum = sum+ $y_k$, then goto step 3;
step 4  compare the sum with pr/s , $x_i$ is the population
of the ith district which has been chosen:
   step 4.1  $0.95< sum/x_i <1.05$;
   step 4.2  if not, then goto step 2;
step 5  print parallelism the counties and towns into
array A which surrounded by the boundary, then print the
length of the boundary into array B;
step 6  execute the steps above as recurrences;
step 7  get several methods of redistricting ,the
boundaries and length of that is recorded in the two
kinds of arrays above;
step 8  for any methods of redistricting, sumC = $\sum_i C_i$ can
be calculated;
step 9  choose the Best Method with the smallest sumC in
the ideal model.
```

After the process of our algorithm, we get a result of drawing districts.

3 Design of Estimate Model: Convincing Argument

Next, we will build up Partisan Model. Considering over the competition among different parties in reality, it is obvious that whether our partisan model can show out the principle of fairness, and whether voter population can show out the number of representatives.

For the other, it turns to test the shape of practical districts. When the shape which is given by Partition Model is compact enough, we get a result that the gerrymandering exists less possible in this redistricting. With this rule, we can build up a compactness model to estimate.

3.1 Analysis of Partisan Model

If the election is fair enough, it will reflect the wishes of the most people. That is, the proportion of the seats or the representatives which a party won in the state is equal to the proportion of the voters which a party has. We assume the party called Democratic, of course, it is suitable to any other parties. As a matter of fact, there is a balance between the statistical number of seats and the academic number. This situation can be used by us to define a differences function. The function will be showed if the election is fair. When we consider a particular area:

$$f = \frac{E(s)}{m} - \frac{V}{K} \quad . \tag{2}$$

Where V = The number of voters who favor the Democratic in this area.
K =The whole number of voters in this area.
m =The number of representatives in this area
$E(s)$=The number of seats of Democratic.

We can conclude that the smaller the value of f is, the fairer the election is. It is especially that $E(s)$=g(V, *parameters*). In the ideal condition, there is a systematic tendency for the Democratic voters to reside in districts, that is the partisan seat division should be symmetrical. Hence, we can determine the relative parameter β, and and grants the prescribed error parameter α. From the above, we can come to the conclusion that $E(s)$ is linear :

$$E(s) = \alpha + \beta \bullet V. \tag{3}$$

However, the linear function is not exact in reality. There is no vote preference in this equation. Considering the vote preference, the situation is totally different. The most important factor is that there are always partisan swings in polity. We build up seats-votes coordinates to reflect this problem. There is a new function which called The Uniform Partisan Swing function.

The most widely assumption to derive a seats-votes curve from distract-level electoral votes in the literature is called "uniform partisan swing" and was introduces by Butler in 1951(for recent application, see Niemi and Deegan 1978, Gudgin and Taylor 1979, Niemi 1985, Niemi and Fett 1986,and many others). The assumption is that vote proportions in every distract all move in lockstep, swinging back and forth in response to national or statewide electoral forces and without any random error or local factors to make them behave differently.

Fig. 2 presents an example of a seats-votes curve drawn with the uniform partisan swing. The small square denotes the actual statewide election result, and the rest of the line was drawn with the assumption.

Although this approach does provide a reasonable starting point, permitting some estimates to be made that would otherwise be impossible, it also has several severe weaknesses. First, the assumption is empirically false. Tracking real districts from election to election reveals approximate uniform shifting, but always with some variability. Fig. 3 provides an example of this interelection variability; graphs for other pairs of elections look similar, although some have more and some less variability. If the uniform partisan swing assumption held exactly, all of the points in Fig. 3 would fall on a straight line. They do fall somewhat near a straight line, but the evidence for variability around this line is unambiguous. A more reasonable model should explicitly take this variability into account.

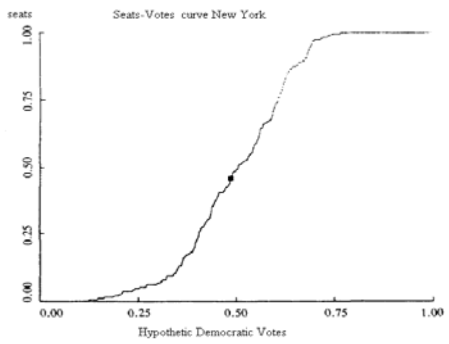


Fig. 2. Seat-Votes curve New York *1972*

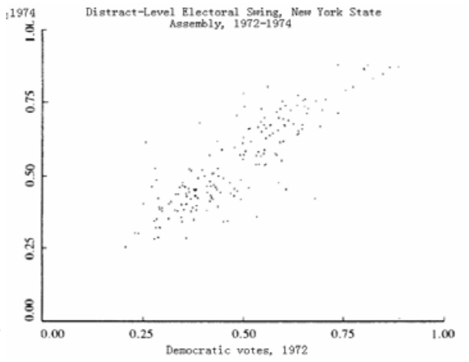


Fig. 3. Distract-Level electoral swing, New York State assembly, *1972-1974*

Furthermore, by assuming the absence of variability, all estimates of variation across repeated hypothetical elections, including standard errors, are zero. Since politics is hardly deterministic, this assumption should obviously be changed. Incorporating this variability will yield estimators with more desirable statistical properties, but it also will enable analysts to distinguish observed changes that are unique from those that reflect significant changes in the underlying electoral system.

Although this approach does provide a reasonable starting point, permitting some estimates to be made that would otherwise be impossible, it also has several severe weaknesses. First, the assumption is empirically false. Tracking real districts from election to election reveals approximate uniform shifting, but always with some variability. Fig. 3 provides an example of this interelection variability; graphs for other pairs of elections look similar, although some have more and some less variability. If the uniform partisan swing assumption held exactly, all of the points in Fig. 3 would fall on a straight line. They do fall somewhat near a straight line, but the evidence for variability around this line is unambiguous. A more reasonable model should explicitly take this variability into account.

Considering partisan swings and according to the theory of Doctor Gary King, we can get the graph of seats-votes.

$$E(s)=\left\{1+\exp\left[-\lambda-\rho\ln\left(\frac{V}{1-V}\right)\right]\right\}^{-1} . \tag{4}$$

Where λ reveals partisan bias. λ=0 means that there is no partisan bias, λ>0 means that the swings favor Democratic, and λ<0 means that the swings favor other parties. ρ indexes the form of Democratic representation.

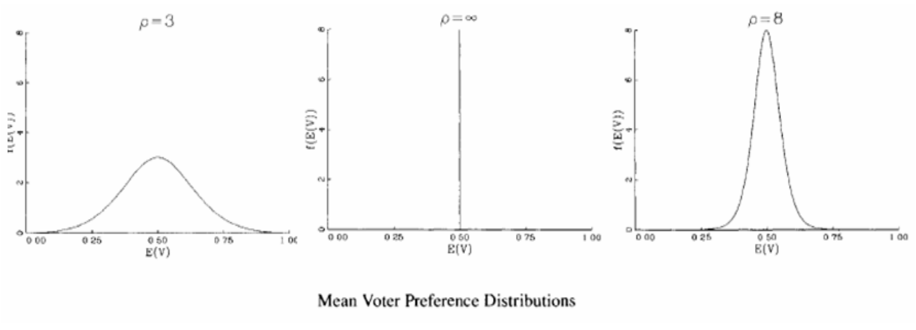


Fig. 4. Mean voter preference distributions

Fig. 4 shows the situation of election existed partisan swings.

3.2 Analysis of Compactness Model

In literature [4]-[8], several authors have argued that gerrymandering can eliminated by drawing districts which are maximally compact, and they consider that redistricting can be performed best by automatically optimizing a pre-specified representation function. Politicians have too many opportunities to manipulate district boundaries by selectively following odd-shaped city, county, and region boundary lines. In common, the most obvious district which is gerrymandered looks unnatural. In other words, an election district lacks compactness and is likely to be gerrymandered. One approach to deriving a Compactness Metric is to compare an election district shape with a compact reference shape. In addition, a useful Compactness Metric should yield the same numeric result for the same shape with different areas. It has the desired property that two districts with the same shape and (approximately) the same number of voters will have the same (or nearly identical) Thus, the Compactness Metric should be normalized to remove differences in population density.

We derive the Compactness Metric formula as follows. The ratio of the boundary to the area (*B/A*) of a square is four times the length of a side (*4s*) divided by the square of the length of a side (s^2), or:

$$\frac{B}{A} = \frac{4*s}{s^2} = \frac{4*sqrt(A)}{A} = \frac{4}{sqrt(A)} \ . \tag{5}$$

Where *sqrt*(*A*) is the square root of *A*, or *s*. Then for any election district, the Compactness Metric (*CM*) is the ratio of the *B/A* ratio of the district to the *B/A* ratio of the equivalent area square, which is

$$CM = (B/A)/(4/sqrt(A)) = (B * {}^{1}\!/_{4})/sqrt(A) \ . \tag{6}$$

For simplicity, we can simply omit the ${}^{1}\!/_{4}$ coefficient and adjust the valid range of *CM* values used for comparison accordingly, by multiplying them by 4:

$$CM = B/sqrt(A) \ . \tag{7}$$

An equivalent but more useful Compactness Metric is obtained by squaring the value of the *B*/*sqrt*(*A*) Compactness Metric to provide a greater range of values and greater differences between the values to be compared:

$$CM = B^2/A \ . \tag{8}$$

4 An Application of Partition Model to Districts of New York State

We use Partition Model to draw the boundary of districts of New York. This state should get 29 seats in the House of Representatives of the Unite States. Put the map of New York into a quadrate table which consists of panes like this. Each pane expresses an element of a matrix. The value of each element indicates the number of population. Obviously, the value of every pane located outside the state map is equal to zero. Therefore, the whole population of New York is the sum of all elements in the quadrate matrix. According to the ideal model of Partition Model and actual data of New York, we partition the whole map. In each time of the partition, we can get a matrix element of a district. After packing them up, the ideal figure can be showed up:

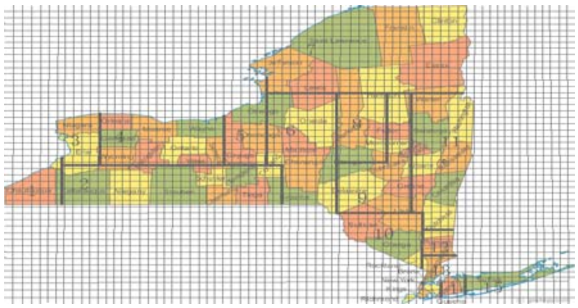


Fig. 5. Result of the ideal model

Arabic numerals (1,2,3,4,……,29) in this map show the order of the districts. Because there are too many people in New York city, it is not partitioned in this map. Of course the constraint of population $0.95<x_i/x_j<1.05$ must be used in each step. We can adjust the boundary of the districts entirely. The practical districts are showed as follow:

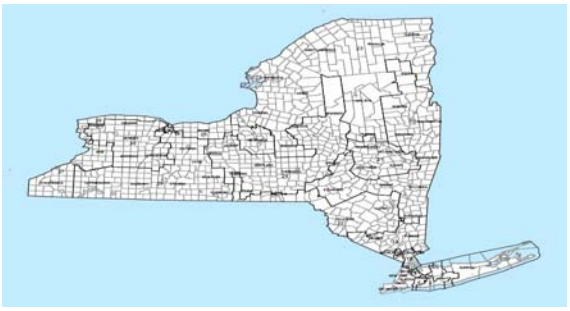


Fig. 6. The final map

5 Conclusion

In order to satisfy with the rules of redistricting and the principle fo fairness at one time, we develop models in different aspects. While achieving the requirement of population and simple, we supply another model to give a convincing argument to the voters. By eliminating the influence of parties in population, we assume that the distribution of each party is stochastic. We define the notion of "simple", and induce the concept of partisan swings which bring a precise result to our model.

However, our models still exist its weaknesses. Although compactness is a reasonable constraint to prevent more extreme cases of election manipulation through gerrymandering, but it is not the sufficient criterion. It is possible to have different redistricting maps with equally compact districts that yield different results in balance of power between the parties and in whether the districts are safe or competitive. Redistricting is a large mathematical problem, the size of the solution set can be enormous, We don't give the proveness of the complex of our algorithm. The above problems are our next advanced work.

References

1. Alan, G.: The American Political Process. Ashgate Publishing Company, New York (1997)
2. Altman, M.: Is Automation the Answer?-The Computational Complexity of Automated Redistricting. California Institute of Technology, California
3. Lijphart, A.: Democracies: Patterns of Majoritarian and Consensus Government in Thirty-One Countries. Yale University Press, New Haven (1984)
4. Harris Jr., C.C.: A Scientific Method of Districting. 9th Behavioral Science 219 (1964)
5. Kaiser, H.: An Objective Method for Establishing Legislative Districts. 10th Midwest Journal of Political Science 200 (1996)
6. Polsby, D., Popper, R.: The Third Criterion: Compactness as a Procedural Safeguard Against Partisan Gerrymandering. 9th Yale Law & Policy Review 301 (1991)
7. Stern, R.S.: Political Gerrymandering: A Statutory Compactness Standard as an Antidote for Judicial Impotence. 41st The University of Chicago Law Review (1974)
8. Wells, D.: Against Affirmative Gerrymandering "Against Affirmative Gerrymandering". Representation and Redistricting Issues (1982)
9. King, G.: Representation through Legislative Redistricting: A Stochastic Model. American Journal of Political Science 33(4), 787–824 (1989)
10. Gelman, A., King, G.: A Unified Method of Evaluating Electoral Systems and Redistricting Plans. American Journal of Political Science 38(2), 514–554 (1994)
11. Wang, Z.G., Wong, Y.S., Rahman, M., et al.: Multi-objective optimization of high-speed milling with parallel genetic simulated annealing. International Journal of Advanced Manufacturing Technology 31(3), 209–218 (2006)
12. Niemi, R.G., Grofman, B., Carlucci, C., Hofeller, T.: Measuring Compactness and the Role of a Compactness Standard in a Test for Partisan and Racial Gerrymandering. Journal of Politics 52(4) (1990)
13. Erikson, R.S.: Malapportionment, Gerrymandering, and Party Fortunes in Congressional Elections. American Political Science Association (1972)
14. Young, H.P.: Measuring the Compactness of Legislative Districts. Legislative Studies Quarterly, XIII (February 1, 1988)

A Rapid Chaos Genetic Algorithm

Jian Gao*, Ming Xiao, and Wei Zhang

School of Computer science and technology, Yantai University, 264005 P.R. China
Tel.: +86 0535 6902 275
hnr-1@163.com

Abstract. Genetic algorithm is an evolutionary algorithm. It is particularly suitable for solving NP-complete optimization problems. In this paper, we propose a rapid genetic algorithm based on chaos mechanism. We introduce the chaos mechanism into the genetic algorithm. Using the ergodic property of the chaos movement, this method can remedy the defect of premature convergence in the genetic algorithm. In the search, this method continuously compresses the searching intervals of the optimization variable for increasing convergence speed. Experiments indicate that this method is a rapid and effective evolutionary algorithm.

Keywords: chaos mechanism, genetic algorithm, NP-complete, premature convergence, optimization problem.

1 Introduction

The genetic algorithm (GA) is a random search method. It simulates the rules of the survival fittest in the process of biological evolution and the random exchange mechanism of the chromosome information within the population. The genetic algorithm possesses outstanding global search ability and a strong robustness; therefore, it is widely used in many fields. The genetic algorithm is an evolutionary algorithm. It is particularly suitable for solving NP-complete optimization problems, which cannot be solved using the traditional optimization methods.

Comparing with the traditional optimization methods, the genetic algorithm has the following characteristics [1,2,3]: 1) The genetic algorithm has strong adaptability. It only requires that the optimization problem is computable, and this requirement has no connection with the essence of the problem. At the same time, GA uses the probability search method, and guides the search direction by the fitness function. So, GA has strong adaptability. 2) The genetic algorithm has global optimization ability. Using multi-point and multi-track search and exchanging information in the search, GA can effectively searches the solution space. 3) The genetic algorithm has implicit

* Corresponding author.

parallelism. It controls n feature strings in the population to respond to o(n3) order patterns. But GA has a defect of premature convergence.

Chaos is a kind of common movement in the nonlinear systems. Chaotic characteristics can be summarized: randomness and ergodic. Taking advantage of these characteristics, chaotic optimization search can avoid falling into local extreme value. But there is an obvious relationship between the optimization result and the search space. When the search space is extremely large, the chaotic optimization search can't obtain satisfying solution. To this end, people introduce the chaos mechanism into the genetic algorithm to remedy the defect of premature convergence [4,5,6,7]. But this advantage is based on a large calculation quantity required by the random relaxation process of the optimization variable. It greatly decreases the convergence speed of the methods; in some optimization problems, it even can't reach the convergence speed of the traditional optimization methods.

In this paper, we propose a rapid chaos genetic algorithm (RCGA). Firstly, we introduce the chaos mechanism into the genetic algorithm. Using the ergodic property of the chaos movement, the defect of premature convergence can be remedied in GA. Then, using the current optimal solution as the center, the search intervals of the optimization variable are continuously compressed for increasing the convergence speed. In this way, RCGA not only can remedy the defect of premature convergence in GA, but also can solve the problem of decelerated convergence in GA after the chaos is introduced. So RCGA can reach the optimal solution (or approximate optimal solution) rapidly and effectively.

2 The Rapid Chaos Genetic Algorithm

2.1 The Chaos Mechanism

The Logisic mapping is a simple and convenient dynamic system that produces chaos variables. The formula is

$$z_{k+1} = 4A\, z_k (1 - z_k). \tag{1}$$

The fork profile of this system is shown in figure 1.

Obviously, when $A \leqslant 0.75$, the system has stable-point attractor; when $0.75 < A \leqslant 0.89$, the system shows some periodicity, when $0.89 < A \leq 1.0$, the system shows a chaos state.

Using the formula (1), we can produce the chaos sequence. Then, the chaos sequence can be introduced into the genetic algorithm for remedying the defect of premature convergence in the genetic algorithm.

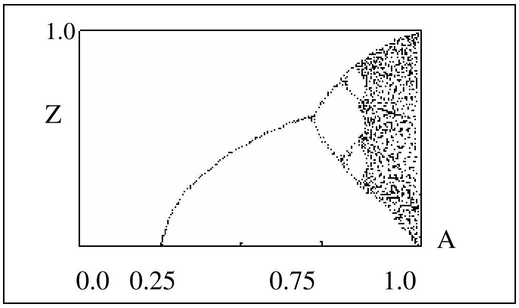


Fig. 1. Fork profile

2.2 The Rapid Chaos Genetic Algorithm

We assume the mathematical model of NP-complete optimization problems as follows:

$$\begin{cases} \max \quad f(p) = f(p_1, p_2, \cdots p_m) \\ s.t. \quad p_j \in [a_j, b_j], \quad j = 1,2,\cdots,m \end{cases} \tag{2}$$

Step 1. Initialization: the population size is defined as L. Let A=1, t=1 (t is the control parameter which is used for adjusting the search interval.). The initial values are randomly given: $z_1, z_2, \cdots, z_m$ ($z_j \in [0,1]$), each of them has small difference with each other. Using the formula (1), chaos variables are produced for each initial value: $z_{j,1}, z_{j,2}, \cdots, z_{j,L}$ ($j = 1,2,\cdots,m$, total: m×L), each of them has different track. All these chaos variables compose the initial population of the evolution: $\{(z_{1,i}, z_{2,i}, \cdots, z_{m,i}) | i = 1,2,\cdots,L\}$. The initial interval of the optimization variables is defined as $[a_j, b_j]$. Let $c_j^{(1)} = a_j, \quad d_j^{(1)} = b_j - a_j, \quad j = 1,2,\cdots,m.$

Step 2. The chaos variables are mapped into the optimization variable intervals. $p_{j,i}^{(t)} = c_j^{(t)} + d_j^{(t)} z_{j,i}^{(t)}, \; p_{j,i}^{(t)} \in [a_j, b_j], \quad (j = 1,2,\cdots,m, i = 1,2,\cdots,L)$.

The evolution population is $p^{(t)} = \{p_i^{(t)} | p_i^{(t)} = (p_{1,i}^{(t)}, p_{2,i}^{(t)}, \cdots, p_{m,i}^{(t)}), i = 1,2,\cdots,L\}$

Step 3. Let $G\left(p_i^{(t)}\right)=f\left(p_i^{(t)}\right)-f_{\min}+\frac{1}{L}\left(f_{\max}-f_{\min}\right)$. $G\left(p_i^{(t)}\right)$ is used as the individual fitness in the current population (in this way, the scope of the fitness is extended and the speed of the convergence will be increased). $f_{\max}$ expresses the maximum of the individual fitness in the current population, $f_{\min}$ expresses the minimum. The population evolves several generations (10~20). The evolution ends if the terminal condition is met, otherwise, goes to step 4.

Step 4. The evolution goes on until the requirement of the initial search is met. The current optimal solution is defined as $p^{*(t)}=\left(p_1^{*(t)},p_2^{*(t)},\cdots,p_m^{*(t)}\right)$.

Step 5. $t\leftarrow t+1$. Using $p_j^{*(t)}$ $(j=1,2,\cdots,m)$ as the centers of the intervals, the search intervals are compressed.

$$[c_j^{(t+1)},c_j^{(t+1)}+d_j^{(t+1)}],\qquad c_j^{(t+1)}=p_j^{*(t)}-\min\left(\left|c_j^{(t)}-p_j^{*(t)}\right|,\ \left|d_j^{(t)}+c_j^{(t)}-p_j^{*(t)}\right|\right),$$

$$d_j^{(t+1)}=2\times\min\left(\left|c_j^{(t)}-p_j^{*(t)}\right|,\ \left|d_j^{(t)}+c_j^{(t)}-p_j^{*(t)}\right|\right),\ (j=1,2,\cdots,m).$$

Step 6. According to the formula: $z_j^{(t+1)}=\frac{p_j^{*(t)}-c_j^{(t+1)}}{d_j^{(t+1)}}(j=1,2,\cdots,m)$, $p_j^{*(t)}$ is mapped into the interval [0,1]. Using formula (1), chaos variables are produced for each $z_j^{(t+1)}$ (total: m×L). Return to step 2.

Algorithm Explication:

(1) We define the average fitness of q-generation population as

$$\overline{G}_q(p)=\frac{1}{L}\sum_{i=1}^{L}G_q(p_i^{(t)})\cdot$$

The requirement of the initial search: $\left|\overline{G}_q(p)-\overline{G}_{q-1}(p)\right|<\delta$. δ is a pre-given small positive number.

(2) We define the maximum of the individual fitness in q-generation population as $G(p_q^*)=\max\{G(p_1^{(t)}),G(p_2^{(t)}),\cdots,G(p_L^{(t)})\}$.

The termination condition: $\left|G\left(p_q^*\right)-G\left(p_{q-1}^*\right)\right|<\varepsilon$. ε is a pre-given small positive number.

2.3 The Convergence of the Rapid Chaos Genetic Algorithm

Theorem: The probability that the rapid chaos genetic algorithm converges to the global optimal solution is one.

Proof: In section 2.2, some closed interval sets have been obtained. $\left[c_j^{(t)}, c_j^{(t)}+d_j^{(t)}\right]$ $(j=1,2,\ldots,m; t=1,2,\ldots\ldots)$,

$$\left[c_j^{(1)}, c_j^{(1)}+d_j^{(1)}\right]\supset\left[c_j^{(2)}, c_j^{(2)}+d_j^{(2)}\right]\supset\ldots\ldots,\quad (j=1,2,\cdots,m).$$

According to *Weierstrass*'s accumulation principle and the theorem of the closed interval sets, in each closed interval sets, there is certainly a point p_j^* that satisfies $p_j^*\in\bigcap_{t=0}^{\infty}\left[c_j^{(t)}, c_j^{(t)}+d_j^{(t)}\right]$ $(j=1,2,\ldots\ldots,m)$. On the other hands, in the measure theory, we assume that N_ε expresses the neighborhood of the global optimal solution p^* . The event is A_t that the vectors $p^*(t)$ produced by RCGA drop in N_ε . Obviously, $P(A_0)\le P(A_1)\le P(A_2)\le\ldots\ldots$, the probability measures of A_t is undiminished and $P(A_t)\le 1$, then $\lim_{t\to\infty} P(A_t)=P\left(\bigcup_{t=0}^{\infty} A_t\right)=1$. That is, the probability that the rapid genetic algorithm converges to the global optimal solution is one.

3 Simulation Experiments

We use four traditional functions to test RCGA performance, and compare RCGA performance with OGA/Q's [3]. F1 is single apex function, F2~F4 are multi-apex functions. In these experiments, $p_c=0.9$(crossover probability), $p_v=0.01$(variance probability), $\delta=10^{-2}$, $\varepsilon=10^{-4}$. For F1~F3, L=80, and L=100 for F4. In Table 1, the results are the average values of twenty times experiments.

F1: min $f(x)=\sum_{i=1}^{n-1}\left[100\left(x_{i+1}-x_i^2\right)^2+\left(x_i-1\right)^2\right]$; $S=[-5,10]^n$; $f_{\min}=0$.

F2: min $f(x)=\sum_{i=1}^{n}(-x_i\sin(\sqrt{|x_i|}))$; $S=[-500,500]^n$; $f_{\min}=-12569.5$.

F3: min $f(x)=\frac{\pi}{n}\{10\sin^2(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2\times[1+10\sin^2(\pi y_{i+1})]+(y_n-1)^2\}$

$$+\sum_{i=1}^{n}u(x_i,10,100,4)$$

$$u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, & x_i>a\\ 0, & -a\le x_i\le a\\ k(-x_i-a)^2, & x_i<-a\end{cases}\quad y_i=1+\frac{1}{4}(x_i+1);\ S=[-50,50]^n;\ f_{\min}=0.$$

F4: min $f(x)=-\sum_{i=1}^{n}\sin(x_i)\sin^2(\frac{i\times x_i^2}{\pi})$; $S=[0,\pi]^n$; $f_{\min}$ less than –99.60.

Table 1. Comparing the performance of RCGA with OGA/Q's

f	Average value		Standard square error	
	RCGA	OGA/Q	RCGA	OGA/Q
F1	4.512E-05	6.896E-03	1.836E-04	1.952E-03
F2	-12569.49	-125769.42	8.746E-8	6.468E-04
F3	1.033E-09	6.012E-06	5.898E-10	1.161E-06
F4	-99.5313	-92.86	2.702E-04	2.628E-02

4 Conclusion

In this paper, we propose a rapid chaos genetic algorithm (RCGA). Using the ergodic property of chaos movement, the defect of the premature convergence can be remedied in GA. Using the current optimal solution as the center, the search intervals of the optimization variable are continuously compressed for increasing convergence speed. In this way, RCGA not only can remedy the defect of premature convergence in GA, but also can solve the problem of decelerated convergence in GA after the chaos is introduced. Experiments indicate that this method is a rapid and effective evolutionary algorithm. It can be seen from table 1, this method is especially suitable for the optimization problem of multi-apex function.

References

1. Cobb, H.G.: An Investigation into the Use of Hyper mutation as an Adaptive Operator in Genetic Algorithms Having Continuous, Time-Dependent Nonstationary Environments, Navy Center for Applied Research in Artificial Intelligence, 1990, 6760 (NLR Memorandum), Washington, D.C. (1990)
2. Ursem, R.K.: When Sharing Fails. In: Proceedings of the 2001 Congress on Evolutionary Computation, CEC 2001, pp. 873–879 (2001)
3. Leung, Y.W., Wang, Y.: An orthogonal genetic algorithm with quantization for global numerical optimization. IEEE Transaction on Evolutionary Computation 5(1), 41–53 (2001)
4. Liao, G.-C., Tsao, T.-P.: Application embedded chaos search immune genetic algorithm for short-term unit commitment. Electric Power Systems Research 7(2), 135–144 (2004)
5. Coelho, L.S., Mariani, V.C.: Combining of chaotic differential evolution and quadratic programming for economic dispatch optimization with valve-point effect. IEEE Transactions on Power Systems 21(2), 989–996 (2006)
6. Juang, C.F.: A hybrid of genetic algorithm and particle swarm optimization for recurrent network design. IEEE Transactions on Systems Man and Cybernetics, Part B 34, 997–1006 (2004)
7. El-Mihoub, T., Hopgood, A., Nolle, L., Battersby, A.: Performance of hybrid genetic algorithms incorporating local search. In: Horton, G. (ed.) 18th European Simulation Multiconference (ESM 2004), Magdeburg, Germany, pp. 154–160 (2004)

Fitness Function of Genetic Algorithm in Structural Constraint Optimization

Xinchi Yan and Xiaohan Wang

School of Environment & Civil Engineering, Jiangnan University,
Wuxi, China
{skywalkeryan,wangxh412}@yahoo.com.cn

Abstract. The mathematics models of Reliability-based Structural Optimization (RBSO) were presented in this paper, then how to handle the constraint become sixty-four-dollar question of establishing the fitness function. Based on exterior penalty function method, penalty gene is made adaptively according to population's evolution, then the fitness function is established, which is mapping formula of objective function and constraints. Subsequently laxity variable is introduced in primary mathematic model, based on Lagrange multiplier method, a new fitness function mapping formula is made, this method can avoid penalty function morbidity by means of adding a Lagrange multiplier, and has a more quick and stable convergence. Then, using GA successfully solved a numerical constrained optimization issue by this two mapping functions. The calculation shows that the two equations are reasonable and efficient, and Lagrange multiplier method has better global optimal capability.

Keywords: genetic algorithm(GA), constraint optimization, fitness function, exterior penalty function method, Lagrange multiplier method.

1 Introduction

Genetic Algorithm is part of evolutionary computation techniques that employ the stochastic search techniques based on principles of natural genetics [1,2]. Import parallel algorithm into GA can be more manageable in structural optimization [3]. In the most case of real structural optimization design, there are constraint conditions, but standard GA only fit for maximum search without constraints. So, how to handle the constraints and objective function, and then establish a tractable fitness mapping function becomes the most important matter of this algorithm.

Contrast the traditional optimization design, reliability-based structural optimization can enhance design quality markedly, and RBSO become an important international scientific research question for discussion. In this paper, system reliability index was treated as design constraints, based on exterior penalty function method and Lagrange multiplier method, fitness functions was made, respectively.

2 Exterior Penalty Function Method

Exterior penalty function method is prevalent in GA, Michalewicz [4] has presented penalty function-based linear constraint optimization method, according to that, this paper established the mapping function of objective function and constraint.

The mathematics model of RBSO is as follows:

$$\begin{cases} \min W = \sum_{i=1}^{n} A_i L_i \rho_i \\ S.t. \quad A^L \le A \le A^U \\ \qquad \beta_s \ge \beta_s^a \end{cases} \tag{1}$$

where A^L, A^U are upper limit and lower limit of design variables, respectively; β_s is system reliability index; $\beta_s{}^a$ is admit system reliability index.

Set up a new function g as follow:

$$g = \begin{cases} 1 - \dfrac{\beta_s}{\beta_s{}^a}, & \beta_s < \beta_s{}^a \\ 0, & else \end{cases} \tag{2}$$

Then the new fitness mapping function is

$$F = C - W(1 + k_1 g) \tag{3}$$

where C is a big enough constant; k_1 is penalty operator, in different optimum issues, the value of k_1 has to be consulted experience, in this paper, it has an adaptive equation value:

$$k_1 = \begin{cases} \dfrac{1.0}{C_{\max} - C_{\min}}, & if\ C_{\max} \ne C_{\min} \\ 1.0, & else \end{cases} \tag{4}$$

where $C_{\max}, C_{\min}$ =maximum and minimum values of the corresponding normalized constraint violations in the current population [4]. So, we get an integrated fitness function like Equation (3).

3 Lagrange Multiplier Method

Lagrange multiplier method is an improvement of penalty function method, and at the present time, it is one of the most efficient constraint optimizations.

Flabby variable z^2 was introduced into the model Equation (1), then inequation constraint turn to equation constraint, and mathematics model turn to:

$$\begin{cases} \min W(A) = \sum_{i=1}^{n} A_i L_i \rho_i \\ s.t. \quad A^L \le A \le A^U \\ \beta_s - z^2 - \beta_s{}^a = 0 \end{cases} \tag{5}$$

Its aggrandized Lagrange function is

$$\Phi(A, z, \lambda, r) = W(A) + \lambda\left(\beta_s - \beta_s{}^a - z^2\right) + \frac{r}{2}\left(\beta_s - \beta_s{}^a - z^2\right)^2 \tag{6}$$

where, r is a constant bigger than 0; λ is Lagrange multiplier, and

$$\lambda_{k+1} = \max\left\{0, \lambda_k + r\left(\beta_s - \beta_s{}^a\right)\right\} \tag{7}$$

where k is iterative times.

And take into account

$$z^2 = \frac{1}{r} \cdot \max\left\{0, -\left[\lambda + r\left(\beta_s - \beta_s{}^a\right)\right]\right\} \tag{8}$$

We obtain new fitness function as follow:

$$F = C - W(A) - \frac{1}{2r} \cdot \left\{\max\left[0, \lambda + r\left(\beta_s - \beta_s{}^a\right)\right]^2 - \lambda^2\right\} \tag{9}$$

where, C is also a big enough constant; $W(A)$ is the weight of system. In Equation (9), flabby variable z^2 has disappeared; the value of Lagrange multiplier λ should be bigger than zero.

4 Numeral Example

A classical 25-bar truss RBSO problem is present as an illustrative example by using these two methods.

The details of the truss geometry and loading are shown in Fig. 1, where $\overline{C}_Y = 27.6kN/cm^2$, $Cov = 0.05$. $E = 6895kN/cm^2$, $\rho = 2.7\times10^{-3}kg/cm^3$. loads $\overline{P}_1 = 88.9kN$, $\overline{P}_2 = 22.6kN$, $Cov = 0.2$. The admit system reliability index is $\beta_s{}^a = 3.5$. bars' cross-sectional areas are A_i (0.5~15) cm^2 .

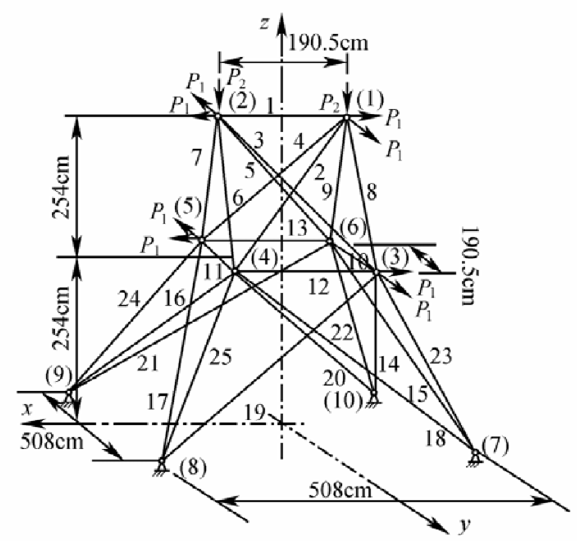


Fig. 1. 25-bar truss structure

Member grouping details are I (A_1) , II ($A_2 = A_5$) , III ($A_3 = A_4$) , IV ($A_6 = A_9$) , V ($A_7 = A_8$) , VI ($A_{10} = A_{11}$) , VII ($A_{12} = A_{13}$) , VIII ($A_{14} = A_{17}$) , IX ($A_{15} = A_{16}$) , X ($A_{18} = A_{21}$) , XI ($A_{19} = A_{20}$) , XII ($A_{22} = A_{25}$) , XIII ($A_{23} = A_{24}$).

The mathematics model of RBSO is as follows:

$$\begin{cases} \min W = \sum_{i=1}^{25} A_i L_i \rho_i \\ S.t. 0.5 \le A \le 15 \\ \beta_s \ge 3.5 \end{cases}$$

Where C in Equation (3) and (9) can not be too big, otherwise the fitness evolution will be insignificant. So, $C = 1.25 \times$ structure system weight when all A_i equal the maximal value, then $C = 416.84$ [5].

1. Exterior penalty function method
   Fitness mapping function is
   $F = 416.84 - W(1 + k_1 g)$

where $g = \begin{cases} 1 - \dfrac{\beta_s}{3.5}, & \beta_s < 3.5 \\ 0, & \beta_s \ge 3.5 \end{cases}$

2. Lagrange multiplier method

Fitness mapping function is

$$F = 416.84 - W - \frac{1}{2r} \cdot \left\{ \max\left[0,\ \lambda + r\left(\beta_s - \beta_s{}^a\right)\right]^2 - \lambda^2 \right\}$$

where λ is Lagrange multiplier, and $\lambda_{k+1} = \max\left\{0, \lambda_k + r\left(\beta_s - \beta_s{}^a\right)\right\}$; r is a constant bigger than 0.

Algorithm adopts binary codes, and the termination criterion is iterative times equal 110. The control parameters are shown in Table 1, and the result is given in Table 2.

Table 1. Control parameters of AIGA

Control parameters of AIGA	
Population size n	30
Bit string length l_s	10
Individual bit string length l	130 (13×10)
Crossover probability p_c	0.8
Mutation probability p_m	0.05

Table 2. Optimization result of 25-bar truss

Element group	**Cross-sectional areas** (cm^2)	**Element group**	**Cross-sectional areas** (cm^2)
I	5.618	II	5.240
III	4.649	IV	4.944
V	9.732	VI	5.236
VII	4.156	VIII	4.156
IX	4.156	X	4.156
XI	4.156	XII	4.643
XIII	4.640		
Weight (kg)	111.158	β_s	3.500

In order to analysis the algorithm's capability and convergence efficiency, the statistic of every generation optimum are shown in Fig. 2. It's clear that both mapping functions can get steady convergence result, but Lagrange multiplier method's iterative process is more quickly, that is to say, adopt this method to handle the constraint and construct fitness function, can improve GA's searching capability availably.

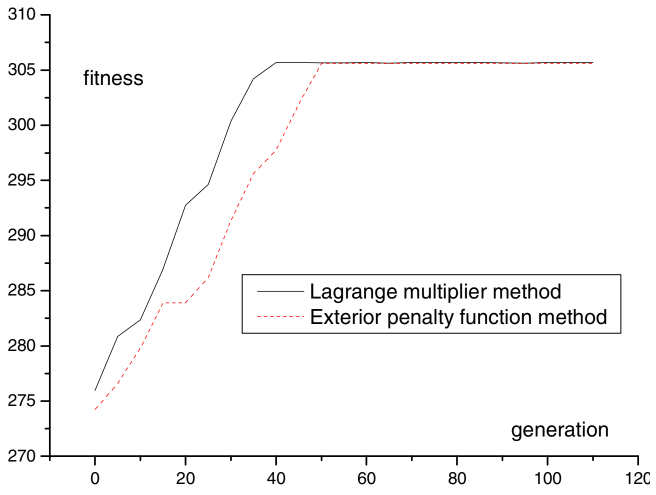


Fig. 2. Maximum fitness of every generation

5 Conclusions

Focus on RBSO issue, this paper adopts exterior penalty function method and Lagrange multiplier method construct fitness functions, respectively. Under the same conditions, 25-bar truss has been calculated. From the maximum fitness of every generation, it is clear that both mapping functions can get the same steady convergence result, the functions are reasonable and efficient. Especially, Lagrange multiplier method's iterative process is more quickly, and has better global optimal capability.

References

1. Holland, J.H.: Adaptation in Natural Artificial Systems. MIT Press, MA (1975)
2. Goldberg, D.E.: Computer-aided Gas Pipeline Operation Using Genetic Algorithms and Rule Learning. Department of Civil Engineering, University of Michigan, No. 8402282 (1983)
3. Ma, Y., Sun, J., Xu, W.: A Parallel Particle Swarm Optimization Algorithm. In: DCABES 2006 Proceedings (2006)
4. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs, 3rd edn. Springer, Heidelberg (1996)

5. Prasad Varma Thampan, C.K., Krishnamoorthy, C.S.: System Reliability-Based Configuration Optimization of Trusses. Journal of structural Engineering 127(8), 947–955 (2001)
6. Yan, X.C., An, W.G., Zhao, W.T.: Adaptive Immunity Genetic Algorithm. Chinese Journal of Applied Mechanics, 445–448 (2005)

Using Genetic Algorithm for Classification in Face Recognition

Xiaochuan Zhao

China north optical-electrical technology co. ltd,
Beijing, China, 100176
sincerezxc@163.com

Abstract. This paper proposed a new algorithm of face recognition using genetic algorithm for classification. After initialization, selection and recombination are executed in a loop until fitness criterion is reached. The fitness function is calculated by an information theoretical measure. Experimental results presented here show that the proposed face recognition algorithm is effective.

Keywords: face recognition, genetic algorithm, region selection algorithm.

1 Introduction

Face recognition compliments face detection. Face detection is the process of finding a 'face' within images or videos and Face Recognition is the process of matching the detected 'face' to one of many the faces known to the file system. Human faces are complex, changeful and high dimensional patterns. Although it is toil for human beings to recognize familiar faces, face recognition is a formidable task for machines. Even so, because of the vast potential applications, face recognition has become an active research area of computer vision and pattern recognition for decades.

Sander Koelstra et al. [1] proposed a dynamic-texture-based approach to the recognition of facial Action Units (AUs, atomic facial gestures) and their temporal models (i.e., sequences of temporal segments: neutral, onset, apex, and offset) in near-frontal-view face videos. Two approaches to model the dynamics and the appearance in the face region of an input video are compared: an extended version of Motion History Images and a novel method based on Non-rigid Registration using Free-Form Deformations (FFDs). The extracted motion representation is used to derive motion orientation histogram descriptors in both the spatial and temporal domain. Per AU, a combination of discriminative, frame-based Gentle Boost ensemble learners and dynamic, generative Hidden Markov Models detects the presence of the AU in question and its temporal segments in an input image sequence. When tested for recognition of all 27 lower and upper face AUs, occurring alone or in combination in 264 sequences from the MMI facial expression database, the proposed method achieved recognition accuracy of 89.2% for the MHI method and of 94.3% for the FFD method. The generalization performance of the FFD method has been tested using the Cohn-Kanade database.

Stefano Berretti et al. [2] presented a novel approach to 3D face matching that shows high effectiveness in distinguishing facial differences between distinct individuals from differences induced by non-neutral expressions within the same individual. The approach takes into account geometrical information of the 3D face and encodes the relevant information into a compact representation in the form of a graph. Nodes of the graph represent equal width facial stripes. Arcs between pairs of nodes are labeled with descriptors, referred to as 3D Weighted Walkthroughs, that capture the mutual relative spatial displacement between all the pairs of points of the corresponding stripes. Face partitioning into iso-geodesic stripes and 3DWWs together provide an approximate representation of local morphology of faces that exhibits smooth variations for changes induced by facial expressions. The graph based representation permits very efficient matching for face recognition and is also suited to be employed for face identification in very large datasets with the support of appropriate index structures. The method obtained the best ranking at the SHREC 2008 contest for 3D face recognition.

Tobias Rehrl et al. [3] presented a real-time capable framework that recognizes traditional visual human communication signals in order to establish a more intuitive human-machine interaction. Humans rely on the interaction partner's face for identification, which helps them to adapt to the interaction partner and utilize context information. Head gestures (head nodding and head shaking) are a convenient way to show agreement or disagreement. Facial expressions give evidence about the interaction partners' emotional state and hand gestures are a fast way of passing simple commands. The recognition of all interaction queues is performed in parallel, enabled by a shared memory implementation.

Yangfeng Ji et al. [4] proposed an improved sparse representation based classification algorithm. Firstly, for a discriminative representation, a non-negative constraint of sparse coefficient is added to sparse representation problem. Secondly, Mahalanobis distance is employed instead of Euclidean distance to measure the similarity between original data and reconstructed data. Their classification algorithm for face recognition has been evaluated under several illuminations and poses using standard face databases. The experimental results demonstrate that the performance of their algorithm is better than that of the up-to-date face recognition algorithm based on sparse representation.

Fulong Wang et al. [5] proposed a method not only extracts nonlinear feature for faces effectively, but also reconstructs between-class and within-class scatter matrix by weighted schemes. So it can modify the kernel maximum scatter difference discriminate criterion function. Considering this method sensitive to the change of illumination, a pretreatment strategy that can reduce image gradation is used. Finally experiments performed on ORL and Yale face database verify the effectiveness of their method.

Pengzhang Liu et al. [6] proposed a novel method, named Modular Locality Preserving Projection (Modular LPP). Their method is derived from the LPP methods, and is designed to handle face images with various illuminations and facial expressions. In their method, the face images are divided into smaller sub-images and the LPP approach is applied to each of these sub-images. As some of the local facial features of an individual do not vary even when the lighting directions and facial expressions vary, the proposed method is expected to cope with these variations. The

Modular LPP and its variant are compared with LPP, based on the Yale face database. Experimental results show the significant improvement of their algorithm.

Fulong Wang et al. [7] proposed a fusion of facial symmetry information method is developed for improving two-dimensional principal component analysis. Their method uses the characteristic of facial symmetry to generate odd-even symmetry images, by weighting odd-even symmetry matrix to replace original image matrix extracting features, and at last minimum distance classifier is used for classification. The predominance of this method is that it takes full advantages of facial symmetry information and considers the impact of odd symmetry matrix which reflects the non-symmetric in Face Recognition. The experiment results on the YALE and ORL face database show that this method has better performance and robustness than the classical PCA and 2DPCA.

Each feature is in fact a region in the face image. For the convenience of analysis and computing, the region is often defined as a rectangle in the image. For instance, eigenface uses the whole face image as global feature, and eigenfeature uses the rectangular region around a certain facial feature, such as eye, nose or mouth, as local feature. However, the extension and position of the local feature are empirically determined by operators. Moreover, there is no reliable criterion to determine which facial feature and how many of them should be used. Thus the performance of those local-feature-based methods greatly depends on the experience of the operators. From this point of view, face recognition can be realized in two steps: the first step is to select one or more features from all possible feature candidates; the second step is to train a classifier for face recognition based on the selected features. It is up to the operators to determine which regions and how many of them should be used. Based on the common sense that facial features, such as eyes, nose and mouth, are crucial in face recognition, most algorithm designers choose to use the image regions around those salient facial features. Similarly, in order to select suitable regions for face recognition, attention should be focused on those accurate and diverse ones.

In this paper, we will adopt region selection algorithm which proposed by Xin Geng et al. [8]. We proposed a new face recognition algorithm using genetic algorithm for classification. After initialization, selection and recombination are executed in a loop until fitness criterion is reached. The fitness function is calculated by an information theoretical measure. Given an unknown face image, its similarities to all the face images stored in the database are computed. Then the most similar one in the database is regarded as matching with the unknown face image.

The rest of the paper is organized as follows. Section 2 is the description of using genetic algorithm for classification. Section 3 focuses on experiments and evaluations. Finally, we end this paper with a conclusion and the future work.

2 Using Genetic Algorithm for Classification

The module of using genetic algorithm for classification is the kernel of our method. Classification has been studied for many decades by machine learning and statistics communities.

Genetic algorithm is a search technique used in computing to find exact or approximate solutions to optimization and search problems. Genetic algorithms are categorized as global search heuristics. Genetic Algorithms have shown their advantages in dealing with the highly non-linear search spaces that result from noisy and multimodal functions.

The genetic algorithm works by randomly selecting pairs of individual chromosomes to reproduce for the next generation. The probability of a chromosome being selected is proportional to its fitness function value relative to the other chromosomes in the same generation. To reproduce, a crossover procedure is defined. In the classical GA, two chromosome strings reproduce by selecting a random bit for the crossing site, the strings are sliced at the site, and the two tail pieces are swapped and rejoined with the head pieces to produce two progenies.

At each steps of such algorithm a set of N potential solutions is chosen in an attempt to describe as good as possible solution of the optimization problem. This population $p = \{I_1, I_2, ..., I_N\}$ is modified according to the natural evolutionary process. After initialization, selection and recombination are executed in a loop until fitness criterion is reached. Each run of the loop is called a generation and P(t) denotes the population at generation t.

Let us see how encoding scheme is used to represent both categorical and continuous attributes present in the dataset. In the categorical case, if a given attribute can take on k-discrete values then we can encode this attribute by using k-bits. The i th value of the attribute's domain is a part of the rule if and only if i th bit is 1.

This encoding scheme can be implemented using bits to represent the value of a continuous attribute in binary notation. For instance the binary string 00001101 represents the value 13 of a given integer-value attributes.

The computation of the degree of interestingness of a rule, in turn, consists of two terms. The degree of interestingness of the rule antecedent is calculated by an information theoretical measure. The information gain is given by these following formulas:

$$InfoGain(A_i) = Info(G) - Info(G \mid A_i) \tag{1}$$

$$Info(G) = -\sum_{i=1}^{m} (p(g) * \log_2(p(g))) \tag{2}$$

$$Info(G \mid A_i) = \sum_{i=1}^{n} (p(v_{ij}) * (-\sum_{j=1}^{m} p(g \mid v_{ij}) * \log_2(p(g \mid v_{ij})))) \tag{3}$$

Where, m is the number of possible values of the goal attribute G, n is the number of possible values of the attribute A, p(g) denotes the probability of g and $p(g \mid v_{ij})$ denotes the conditional probability of g given v_{ij}.

The fitness function is computed as follows:

$$Fitness = 1 - \frac{\dfrac{\sum_{i=1}^{n-1} InfoGain(A_i)}{n-1}}{\log_2(\mid dom(G) \mid)} \tag{4}$$

Where, n is the number of attributes in the antecedent and $dom(G)$ is the domain cardinality of the goal attribute G occurring in the consequent.

The similarities between the corresponding features of two images are calculated according to their projections, and the similarities of all corresponding features are used to form a global similarity score of the two images. Face recognition is then performed based on the similarity score. The similarity between two images is defined as the sum of similarities on all the selected regions. Given an unknown face image, its similarities to all the face images stored in the database are computed. Then the most similar one in the database is regarded as matching with the unknown face image.

3 Experiments and Evaluations

The growing number of face recognition (FR) methods has imposed the development of standard evaluation protocols and the creation of large evaluation databases. Based on that, a series of public tests evaluating face recognition algorithms and examining their limitations have been performed. The most recent one is the Facial Recognition Vendor Test of 2006 (FRVT2006), in which several state of the art commercial FR systems were tested using a database of images of Mexican VISA applicants collected by the US Department of State. The verification performance was measured using the false acceptance rate (FAR) and false rejection rate (FRR). FAR is defined as the percentage of instances that a non-authorized individual is falsely accepted by the system, while FRR is defined as the percentage of instances an authorized individual is falsely rejected by the system.

We used 6000 images to experiment, including 10 clusters, every cluster included 600 images.

We compared three face recognition algorithms: Method1 is a standard PCA algorithm. Method2 is a combination of PCA and LDA algorithm. Method3 is Face Recognition Algorithm Using Genetic Algorithm for Classification.

We compared the results of above three algorithms, the evaluate result of above three algorithms as Table 1 shows:

Table 1. The false acceptance rate and the false rejection rate

	The False Acceptance Rate			**The False Rejection Rate**		
	Method1	**Method2**	**Method3**	**Method1**	**Method2**	**Method3**
Cluster1	17.6%	16.7%	14.5%	24.7%	19.9%	18.6%
Cluster2	19.4%	18.1%	17.7%	23.2%	23.6%	20.3%
Cluster3	18.5%	17.1%	16.4%	22.8%	21.9%	19.3%
Cluster4	16.1%	15.6%	13.4%	23.5%	21.7%	19.8%
Cluster5	17.6%	16.3%	16.7%	21.7%	20.9%	20.5%
Cluster6	16.4%	15.8%	15.2%	19.3%	17.8%	17.2%
Cluster7	15.4%	14.6%	12.8%	23.7%	21.9%	20.3%
Cluster8	17.6%	16.3%	15.9%	20.8%	20.1%	18.7%
Cluster9	16.5%	15.2%	14.6%	21.4%	20.8%	19.5%
Cluster10	18.6%	17.4%	16.9%	22.5%	21.3%	18.7%

The data in table1 shows, face recognition algorithm using genetic algorithm for classification is better than other two algorithms. Face recognition algorithm using genetic algorithm for classification can reduce the false acceptance rate and the false rejection rate.

Experimental result shows that the proposed face recognition algorithm using genetic algorithm for classification is effective.

4 Conclusions and Future Work

We proposed a new face recognition algorithm using genetic algorithm for classification. After initialization, selection and recombination are executed in a loop until fitness criterion is reached. The fitness function is calculated by an information theoretical measure. Given an unknown face image, its similarities to all the face images stored in the database are computed. Then the most similar one in the database is regarded as matching with the unknown face image. Experimental results show that the proposed face recognition algorithm is effective.

Also, there is a lot of room for improvement. For example, we need to speed up genetic algorithm for classification. Many features can be used to face recognition algorithm, we need to find the appropriate features fit different condition.

References

1. Koelstra, S., Pantic, M., Patras, I.: A Dynamic Texture Based Approach to Recognition of Facial Actions and their Temporal Models. IEEE Transactions on Pattern Analysis and Machine Intelligence (2010)
2. Berretti, S., Del Bimbo, A., Pala, P.: 3D Face Recognition Using Iso-geodesic Stripes. IEEE Transactions on Pattern Analysis and Machine Intelligence (2010)
3. Rehrl, T., Bannat, A., Gast, J., Wallhoff, F., Rigoll, G., Mayer, C., Riaz, Z., Radig, B., Sosnowski, S., Kühnlenz, K.: Multiple Parallel Vision-Based Recognition in a Real-Time Framework for Human-Robot-Interaction Scenarios. In: 2010 Third International Conference on Advances in Computer-Human Interactions, pp. 50–55. IEEE Press, New York (2010)
4. Ji, Y., Lin, T., Zha, H.: Mahalanobis Distance Based Non-negative Sparse Representation for Face Recognition. In: 2009 International Conference on Machine Learning and Applications, pp. 41–46. IEEE Press, New York (2009)
5. Wang, F., Liu, X., Huang, C.: An Improved Kernel Fisher Discriminant Analysis for Face Recognition. In: 2009 International Conference on Computational Intelligence and Security, pp. 353–357. IEEE Press, New York (2009)
6. Liu, P., Shen, T., Hu, Y., Zhao, S.: Face Recognition Using Modular Locality Preserving Projections. In: 2009 International Conference on Computational Intelligence and Security, pp. 320–324. IEEE Press, New York (2009)
7. Wang, F., Huang, C., Liu, X.: A Fusion of Face Symmetry of Two-Dimensional Principal Component Analysis and Face Recognition. In: 2009 International Conference on Computational Intelligence and Security, pp. 368–371. IEEE Press, New York (2009)
8. Geng, X., Zhou, Z.-H.: Image Region Selection and Ensemble for Face Recognition. Journal of Computer Science and Technology 21, 67–73 (2006)

Dynamic Path Optimization of Emergency Transport Based on Hierarchical Genetic Algorithm

Yongjie Ma[*], Ye Tian, and Wenjing Hou

College of Physics and Electronic Engineering, NorthWest Normal University, 730070 Lanzhou, China
{myjmyj,lovetianye,WenjingHou}@163.com

Abstract. An efficient search algorithm, which have to take into account the time and cost on emergency transport, the hierarchical Genetic Algorithm was proposed. Local optimization is achieved by Bottom GA in Subnet, global optimization is achieved by top GA in the whole network. The Contradiction of Global search capability and search efficiency is solved by maintaining a balance between GA Random search and this method is narrow search. The results of calculation show that this method can satisfy the demands of practical engineering of dynamic path selection in a wide range and quick fix of failure path on Emergency transport.

Keywords: Genetic Algorithm, Emergency Transport, Constrained Multi-objective Optimization, Hierarchical.

1 Introduction

The emergency logistics system is a special logistic activity in order to provide the emergency supplies to deal with incidents. The main features of Emergency transportation, which is the core issue of emergency logistics are: 1) The goal of Emergency transport is to achieve the minimizing transportation cost under the premise of the relief supplies and personnel transport to the destination as soon as possible. 2) Each sudden of natural disaster and public emergencies has uncertainty dimensions and type, there are many types of relief supplies, and the supply points of each materials may be inconsistent with the demand points. 3) Emergency transportation needs to use different modes of transport.

In recent years, various sudden-onset disasters becomes an important research subject for the scholars at home and aboard that how to use advanced technology means to solve these diverse crises quickly and effectively, and improve the ability of the transportation in the emergency.

According to its characteristics, the problems of the emergency transportation and distribution can be modeled and processed under the constraints of the multi-objectives. If only consider one vehicle which transports emergency supplies, how to

[*] Gansu Province Natural Science Foundation (096RJZA115). The scientific research project of the Education Department of Gansu Province (0901B-08).

pass the necessary emergency supplies for loading and unloading points from the starting point and arrive at the distribution points as quickly as possible, it will be the problem of the shortest path.

For solving the problem of the shortest path for the vehicles, there are already many scholars conducting a large number of deep research. A new real time dynamic optimum route has been presented which approach the true optimum route and put forward improved Dijkstra algorithm and improved A*algorithm to calculate this route, the former suit multi-vehicles navigation and the latter suit solo-vehicle navigation in paper [2]. A directional multi-phase labeling algorithm is proposed to deal with the shortest path cost problem in container motorrail multimodal transportation, with reference to the Dijkstra labeling algorithm in paper [3]. An improved genetic algorithm for path planning is designed. It provides a new approach for solving the optimum path planning problems in practical vehicle guidance systems in paper [4]. A hybrid genetic algorithm with 2-OPT sub-routes optimization was presented for Capacitated Vehicle Routing Problem (CVRP) in the logistics distribution optimization area. In this method, a Double Layers Chromosome (DLC) coding scheme was proposed in paper [5].

Aiming at the characteristics of emergency transports, a new hierarchical genetic algorithms is applied in this paper which has reduced the scale of the solving of the problems. For the fault section, mutation operator is used to repair the local network, to avoid repeated counting; the crossover and mutation of the hierarchical genetic algorithm is limited in different layers, to avoid a wide range of useless search and the impact of the hanging and dead points on the computational efficiency.

2 Analysis of the Shortest Path Problem of the Transportation of Emergency Supplies

The shortest path of the transportation of the emergency supplies, it could be described as follows: for the road network G=(N,E), N is the set of nodes and E is the set of directional edges between nodes, with the condition $|N|=n$,$|E|=m$. for any directional edge $e_{ij}^{k}\in E$, there are h_e kinds of transport modes, $h_e\le H$, H stands for the number of all transport modes. And $k\le K$, k stands for the number of paths between node i and node j. If supposed $c(i,j,e_{ij}^{k},u,h)$ is the transportation cost that the transport modes h consumes starting to through the path e_{ij}^{k} between node i and node j at time u ,thus $c(i,j,e_{ij}^{k},u,h)$ is a non-negative real number; $t(i,j,e_{ij}^{k},h)$ is the time that the transport modes h pass the path e_{ij}^{k} between node i and node j; $p(i,h_1,h_2)$ is the transportation cost that the transport modes h_1 and h_2 transit between node i and node j; $w(i,h_1,h_2)$ is the transit time that the transport modes h_1 and h_2 exhaust.

The problem of the shortest path of the emergency transportation which is at time T through the path $R=\{ e_{ij}^{k} \mid i,j\in N, e_{ij}^{k}\in E \}$ from the starting point O to the ending point D could be converted into the following functions:

$$\min T = \sum_{i,j\in N, e_{ij}^{k}\in E} t(i, j, e_{ij}^{k}, h) + \sum_{i\in N} w(i, h_1, h_2). \tag{1}$$

$$\min C = \sum_{i,j\in N, e_{ij}^{k}\in E} c(i, j, e_{ij}^{k}, u, h) + \sum_{i\in N} p(i, h_1, h_2) \tag{2}$$

The grades of the sudden-onset disasters are generally divided into four: grade I (particularly severe), grade II(severe), grade III(serious) and grade IV(common). Thus, the objective function of the emergency transportation could be set into grade 1, 2, 3 and 4 for a priority. The priority grade 1 is the primary objective, under the same conditions, the objective function (2) will be considered. But another priority grades of emergency supplies transportation should be an appropriate balance between minimize of the cost.

If u is set to be the quantity of the current emergency supplies, q_{ij} is the capability through the path e_{ij}^{k} between node i and node j, then the transportation must satisfy the following constraints:

$$\sum_{k} q_{ij}^{k} \geq u \tag{3}$$

If supposed $Z=\{ e_{ij}^{k} \mid i,j\in N, e_{ij}^{k}\in E \}$ is the set of the loading and unloading points of the emergency supplies, then the set Z is contained in the transit path R. At the same time, the changes of the paths will deal with the following constraints (4): The transit path R must pass from the starting point O to the ending point D, and the path can not circulate.

$$Z \subseteq R \quad . \tag{4}$$

3 The Two-Layer Multi-objective Genetic Algorithm

For the road network shown in figure 1,when the road network G=(N,E) is partially in trouble, it will take a lot of computing time to re-globally search when the road network is great using the means in paper [2,3,4,5].

3.1 Partition of the Road Network

The road network G could be divided into many sub-network G_i which has the same size subnet. When face to the different problems, the ways of network segmentation will change as well. For instance, in Figure 1, if it is required to start from the starting point 1 to the destination node 15, the path could be divided into three sub-networks 1-3,3-10,10-15. This kind of segmentation way does not only satisfy the constraints (4), but also avoid the appearance of hanging points when the upper genetic algorithm optimizes the results in the entire network.

3.2 The Design of The Bottom Genetic Algorithm

The effect of the bottom genetic algorithms is to improve the efficiency with the genetic algorithm parallel computing to optimize in partial network.

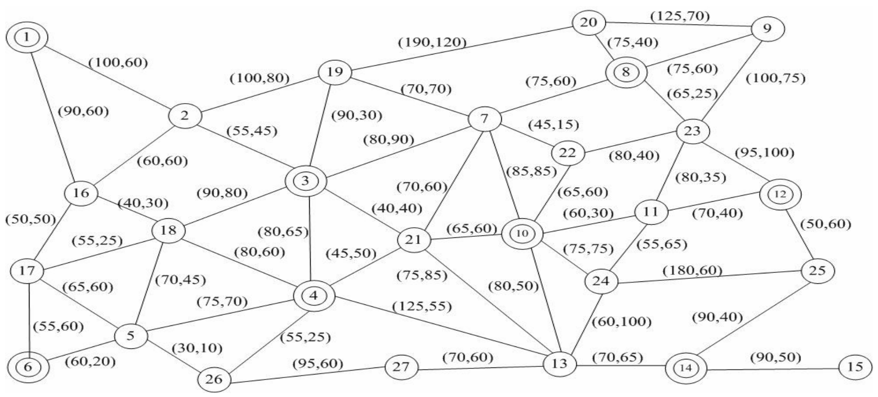


Fig. 1. Structure of the emergency logistics network

1) Genetic Encoding: As to the problem of the shortest path, its feasible solution is generally in the form of the number strings connected by the subscript of the nodes. Thus, the way of the individual coding in genetic algorithm is generally symbol coding. The natural encoding scheme is adopted in this paper. Taking the existence of variable length of the path into account, the natural coding is designed with the method of the variable-length chromosomes way. If the needed path is from the node 2 to the node 4 , the encoding string is (2,16,18,4), (2,3,4) and so on.

2) Initial Population*:* Our path growth method is following: a node directs connecting the starting point is randomly selected as the next node from the starting point, and this is repeated to the ending node. This ensures there will not be any open circuit. At the same time, in order to avoid the loops, in the generating process of the path, it is provided that in a path, if a path node is selected, then this node will be marked. Thus, another nodes which have not been marked will be selected as the new nodes. After every path being selected, all the marks will be refreshed. The encoded individuals will be stored in the collection of the initial individual.

3) Crossover Operator*:* When the crossover operator is applied to the parent individual V_1 and V_2, the procedure is following: 1) The assemble N_c which is the assemble of nodes both in V_1 and V_2 (excepting the starting node and the objective node) is selected as the position of crossover; 2) The node $i \in Nc$ is selected randomly from Nc as the intersection point; 3) The content of V_1 and V_2 before the node i or after is checked , if it is same ,the crossover is given up; otherwise, the next step is continued; 4) All the nodes after crossover points are exchanged and then there will be new offspring. And the length of the offspring may be inconsistent with the parent. For example, if supposed a pair of parent individuals are V_1 (1,2,3,4,13) and V_2 (1,16,18,3,4,21,13), there will be two potential crossing nodes 3,4, it is Nc = {3 , 4 }. And then if the node 4 is selected as the crossing position ,the offspring individuals V_1 and V_2 are respectively V_1 (1,2,3, 4,21,13) and V_2 (1,16,18,3,4, 13).

4) Mutation Operator*:* The re-election of the path mutation is applied. That is two mutated nodes i and j are randomly selected from the parent individuals; a new path is selected to take place of the former from the node i to the node j, and form a new offspring individuals. If the mutation generate the loop, the mutation is abolished.

5) Individuals Ranking: Owing to the problems is multi-objectives constrained optimization, the target (1), and (2) are generally conflicted each other. Therefore, the method of Pareto is adopted which is based on the minimum in paper [6].

Table 1. Results of The Network

TASK	constraints	the best path of subnet 1-21	21-25	The global best path	Object function time	cost
1-25	21must be past	1-2-3-21	21-13-14-25	1-2-3-21-13-14-25	410	325
		1-2-19-3-21	21-10-11-12-25	1-2-3-21-10-11-12-25	440	335
		1-16-2-3-21	21-10-13-14-25	1-16-2-3-21-13-14-25	460	385
		1-16-18-4-21	21-4-13-14-25	1-2-3-21-10-24-25	515	340
		1-16-18-3-21	21-10-24-25	1-16-18-3-21-13-14-25	475	390
1-25	21must be past and 2-3 is invalid	1-16-18-4-21	21-13-14-25	1-16-18-4-21-13-14-25	470	380
		1-16-18-3-21	21-10-11-12- 25	1-16-18-3-21-13-14-25	475	390
		1-16-18-5-26-4-21	21-10-13-14-25	1-16-18-5-26-4-21-13-14-25	545	400
		1-16-18-5-4-21	21-4-13-14-25	1-16-18-4-21-10-13-14-25	540	415
		1-2-19-7-21	21-10-24-25	1-16-18-4-21-4-13-14-25	565	410

If supposed $s_k = \max\{ g_k(X), 0\}\ (k = 1,2,..., r)$,for the rector with n-dimensional f(X)= (f_1(X), f_2(X), … , f_n(X)), in the Solution Space R^n, where X =(x_1 , …,x_m)∈D is a variate with m-dimensional in decision space R^m, D⊆R^m is search space, f: $R^m \rightarrow R^n$ is the objective function, g_i(X) is the Constraints conditions. Function is defined as follows:

$$\text{Better}(X_1, X_2) = \begin{cases} \text{true}, & \text{if } s_k(X_1) < s_k(X_2) \\ \text{true}, & \text{if } f_i(X_1) < f_i(X_2) \text{ and } s_k(X_1) \le s_k(X_2). \\ \text{false}, & \text{otherwise} \end{cases} \quad (5)$$

where, i=1,2,…,n; k=1,2,…,r. The individual populations of the generation t ranked according to the following rules:

$$\text{Rank}(X_i) = N\{X_j \mid X_j \prec X_i; i \ne j; X_j, X_i \in D; i, j = 1,..., pop\} \cdot \quad (6)$$

Where, N{·} is the number of elements in the set, $X_1 \prec X_2$ is the X_1 dominant constraint on the X_2. In this sense, the value of Rank of the optimal Pareto solution is necessarily 0, and the value of Rank the worst individual in each generation is the certain maximum.

6) Selection: In order to avoid offspring atavism, for the offspring individual after crossover and mutation, if it already exists in the parent, then it is abandoned. In order to guarantee fast convergence of the algorithm, ($\mu+\lambda$) evolution strategy is applied to achieve the maintaining of the elite, and to avoid the degradation of offspring. The parent and offspring individuals are combined, the individuals with a smaller Rank value are selected to be into the next generation.

7) Stopping Rule: If the algorithm reaches its maximum GMAX -the most maximal Evolution Algebra, or without any new individuals appear in consecutive generations, then the algorithm is stop. The obtained best solution set is set as the optimal solution set of sub-network.

3.3 Design of the Upper Genetic Algorithm

The upper genetic algorithm in the global optimization mainly base on the bottom genetic algorithm which optimizes in the local, and generate an optimal path and a number of alternative paths: 1) Initial population: The way of arbitrary choices of paths to connected together in sub-networks generates initial population. 2) Crossover encoding: Since each individual is the shortest path, it does not need to mutate, but crossover. Simultaneously, it need not to judge the handing points as with the same points between subnets. 3) Selection and ranking: The approach of chapter 3.2 is applied to select and rank. 4) Partial failure of a path re-election: The shortest path obtained from above calculation due to the follow-up impact of the unexpected events in the course of the usage might lead to a partial failure. when the fault enlarge in scope, the subnet could be started by the corresponding underlying genetic algorithm for local search; when the fault range is smaller, the mutation operator of chapter 3.2.4 can be used for the re-election of paths, to achieve repairing of the partial failure paths .

4 The Shortest Path of the Transport of Emergency Supplies

4.1 The Steps of the Shortest Paths Based on Hierarchical Genetic Algorithm

1) Steps of the bottom Genetic Algorithm a) To take population size pop=30, crossover probability p_c=0.8, mutation probability p_m = 0.4, maximum number of iterations GMAX=200. In order to simplify the calculation, assumed that Figure 1 is an undirected graph, and emergency transportation priority of grade 1, and there is only a mode of transport, the path e_{ij}^k marked on the ($t(i,j,e_{ij}^k,h)$, $c(i,j, e_{ij}^k,u,h)$). b) For the road network shown in fig .1,the Adjacency matrix $A_{n\times n}$ of the road network G=(N,E) as follows (7): Then, initial population is formatted under the method in chapter 3.2.2.

$$a_{ij} = \begin{cases} 1, & \text{w hen}(i,\ j) \in E, \quad \text{and} \quad i,\ j \in N \\ 0, & \text{w hen}(i,\ j) \notin E, \quad \text{and} \quad i,\ j \in N \end{cases} \tag{7}$$

c) According to the method the chapter 3.2.3-3.2.7 , by mutating, crossovering, selecting, the local optimal path of the bottom genetic algorithm can be obtained.

2) Steps of the Upper Genetic Algorithm: Pick up Population size pop=50, Crossover probability p_c = 0.8, Maximum number of iterations GMAX=100, Using the Section 3.3 method for global optimization.

4.2 The Shortest Path Based on Hierarchical Genetic Algorithm

The Steps of The Solving of The Underlying Genetic Algorithm is following: For example, in Figure 1, if starting from the starting node 1 to the destination node 25, which must pass the node 21, then the paths can be divided into two sub-networks 1-21 and 21-25, which does not only satisfy the constraints (4), but also avoid the appearance of hanging points of the upper genetic algorithm searching the global to get the optimum. Table 1 shows the optimum paths from node 1 to node 25, as the constraints are "node 21 must be past" and "21 must be past and Node 3 is invalid ", four suboptimal paths. Fig. 2 shows the curve of the target time and cost on tasks 1-25.

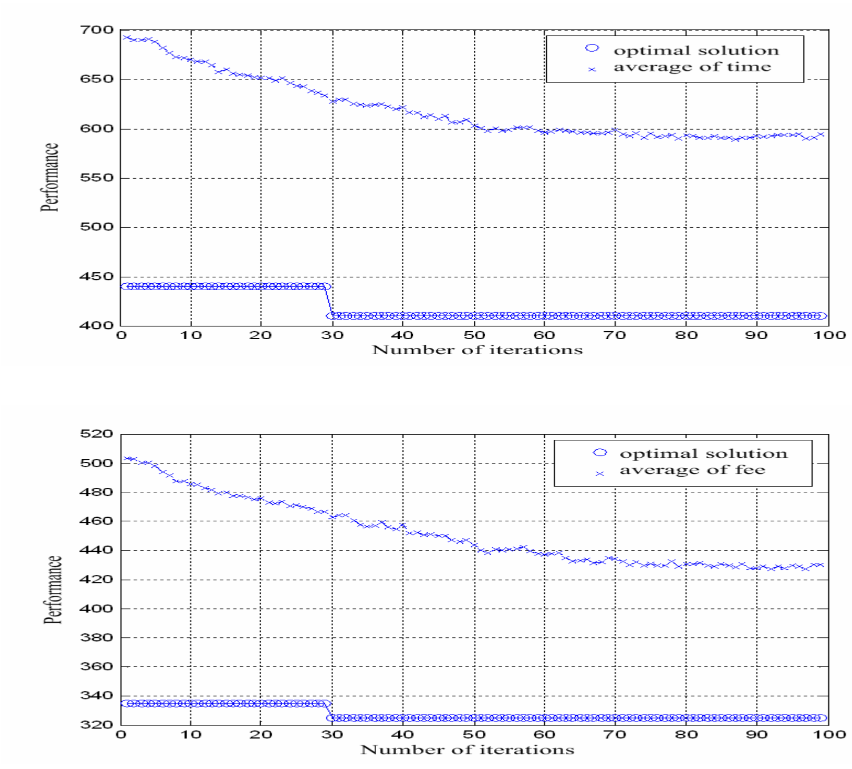


Fig. 2. Performance curve

5 Results and Discussion

In this paper, Hierarchical genetic algorithm is used to solve constrained multi-objective optimization problems, from Table 1 and Figure 2 can be seen: 1) Subnet segmentation method is used for handling of partial constraints. Since the genetic algorithm method limits the global random search into the subnet, compared with the global search in the paper [2-5], it greatly reduces the scope and strength of the search, and improves the search efficiency. 2) The elitism strategy is used to make the individuals which have entered the optimal solution set participate in genetic until the existence of a far better one to replace the individual, thus avoiding the loss of the optimal solution. The results of repeated calculation show that the optimal solution can always be found each time, indicating a strong stability of the algorithm. 3) Since it is independent among subnets, several sub-networks can take the parallel search with the underlying genetic algorithm. When the local node fails, just start the bottom genetic algorithm of the subnet where the invalid node is and repair it through the local search without the need for global search. So it largely improves the speed of the formation and the restoration of the dynamic path of the emergency transport. 4) Lower search can only get the local optimal path, but not necessarily the global optimum. Using the upper search, the global optimization can be achieved based on the lower optimization.

The results show the good performance of hierarchical genetic algorithms in the aspect of selecting dynamic path of emergency transportation in larger road network.

References

1. Ou, Z., Wang, H., Jiang, D., et al.: Emergency Logistics. Journal of Chongqing University (Natural Science Edition) 27(3), 164–167 (2004)
2. Su, Y., Yan, K., Huang, X., Zhu, P.: Study of The Method to Search Dynamic Optimum Route for Vehicle Navigation System. Systems Engineering 18(4), 32–37 (2000)
3. He, G.: Computer-aided Algorithm of the Cost Weighted Shortest Path Problem in Container Motor-rail Multimodal Transportation. Journal of the china Railway Society 28(1), 1–5 (2006)
4. Li, Q., Zhang, W., Yin, Y., Wang, Z.: An Improved Genetic Algorithm for Optimal Path Planning. Information and Control 35(4), 444–447 (2006)
5. Jiang, C., Dai, S., Hu, Y.: Hybrid genetic algorithm for capacitated vehicle routing problem. Computer Integrated Manufacturing Systems 13(10), 2047–2052 (2007)
6. Ma, Y., Bai, Y., Jiang, Z.: Fast Multi-objective Constrained Evolutionary Algorithm and its Convergence. Systems Engineering-Theroy & Practcie 29(5), 149–157 (2009)
7. Mitsuo, G., Runwei, C.: Genetic Algorithms and Engineering Optimization. Tsinghua University Press, Beijing (2004)

Fault Diagnosis of Analog Circuits Using Extension Genetic Algorithm

Meng-Hui Wang*, Kuei-Hsiang Chao, and Yu-Kuo Chung

Department of Electrical Engineering National Chin-Yi University of Technology
35 Lane 215 Chung-Shan Rd. Sec. 1 Taiping City, Taichung County 411 Taiwan, R.O.C.
wangmh@ncut.edu.tw

Abstract. This paper proposed a new fault diagnosis method based on the extension genetic algorithm (EGA) for analog circuits. Analog circuits were difference at some node with the normal and failure conditions. However, the identification of the faulted location was not easily task due to the variability of circuit components. So this paper presented a novel EGA method for fault diagnosis of analog circuits, EGA is a combination of extension theory (ET) and genetic algorithm (GA). In the past, ET had to depend on experiences to set the classical domain and weight, but setting classical domain and weight were tedious and complicated steps in classified process. In order to improve this defect, this paper proposes an EGA to find the best parameter of classical domain and increase accuracy of the classification. The proposed method has been tested on a practical analog circuit, and compared with other classified method. The application of this new method to some testing cases has given promising results.

Keywords: Analog circuit, Fault diagnosis, Extension genetic algorithm (EGA).

1 Introduction

With the more complex of analog circuits and developed, test in analog circuits becoming an urgent task. The faults of Analog circuit are usually divided into two categories: hard faults and soft faults [1]. The hard faults are due to break of circuit component, and soft faults are due to a variation of one (or more) circuit component values over the tolerance range and deviation of about 50% of the faulty element from their nominal value [2-3]. The deviation of the component condition is complex, the poor fault models, component tolerances, nonlinear effects make the automatic fault diagnosis of analog circuits very complex[4-5], because soft faults do not change the circuit topology, but cause the circuit to operate outside its specifications. Hence, there were many fault diagnosis methods using artificial intelligence (AI) techniques to solve the problems in the past [6-7].

Cai originally created the concept of ET to solve contradictions and incompatibility problems in 1983 [8]. In recent years, ET has proposed practical applications on different applications for fault diagnosis [9-12]. The concept of an extension set is to

* Corresponding author.

extend the fuzzy logic value from [0,1] to (-∞,∞), which allows us to define any data in the domain and has given promising results in many fields. Extension clustering methods had depend on experiences to set the classical domain and weight, but setting classical domain and weight were tedious and complicated steps in clustering process. In order to overcome the defect of extension classified method, this paper using EGA searching characteristic of genetic algorithms to find the best parameter of classical domain in the Extension classified method. The fault diagnosis problem of an analog circuit is used to show the accuracy of the proposed method.

2 Extension Genetic Algorithm (EGA)

This paper proposed classifying method involves the combination of extension theory and genetic algorithm. Extension theory provides a means of distance measurement for classification process, and genetic algorithm has the ability to search for an optimal solution in a wide space. EGA is a kind of supervised learning that finds out the best classical domain and gets better accuracy without adjusting weight [11].

2.1 Outline of Extension Theory

In the standard set, an element either belongs to or, so the range of the standard set is {0, 1}, which can be used to solve a two-valued problem. In contrast to the standard set, the fuzzy set allows for the description of concepts in which the boundary is not explicit. It concerns not only whether an element belongs to the set but also to what degree it belongs to. The range of a fuzzy set is [0, 1]. The extension set extends the fuzzy set from [0, 1] to (-∞, ∞). As a result, it allows us to define a set that includes any data in the domain. Extension theory tries to solve the incompatibility or contradiction problems by the transformation of the matter element.

2.2 Basic of Genetic Algorithm

The genetic algorithm (GA) is the most well-known, and it is always to combine with other algorithm for optimized problems. Genetic algorithm is transposed the notions of evolution in nature to computers and imitate natural evolution [13]. Basically, they find solution to a problem by maintaining a population of possible solutions according to the "survival of the fittest" principle. Genetic algorithm constitutes a class of search algorithms especially suited to solving complex optimization problems. In addition to parameter optimization, genetic algorithm is also suggested for solving problems in creative design, such as combining components in a novel creative way. In general, the major advantage of using the GA is that the optimal solution is obtained globally [14]. The genetic algorithm generally includes the following five parts: (1) gene coding, (2) fitness function, (3) selection mechanism, and (4) crossover and mutation mechanism.

2.3 The Computing Method of EGA

This section will present the mathematical description of EGA. The extension method would be found out at the paper of the author [9], so it isn't necessary to be explained here. Before using the algorithm, we define several variables. First, the training patterns are set to be patterns = $[X_1,X_2,\ldots,X_n]$, where the total number of training patterns is N_n. The *i-th* pattern is $X_i^k = \{X_1,X_2,\ldots,X_{Nc}\}$, where the total number of features is N_c, and k is the category of *i-th* pattern. To evaluate the objectives of convergence, N_m is the total mistake number, then the total mistake rate E_T, it can be defined by:

$$E_T = \frac{N_m}{N_n} \tag{1}$$

The learning algorithm of the proposed EGA is shown as follows:

Step1: Choose values of the classical domains. The range of classical domains can be directly obtained from previous experience, or determined from training data as follows:

$$V_{kj}^L = min\{ X_j^k \} \tag{2}$$

$$V_{kj}^U = max\{ X_j^k \} \tag{3}$$

$$j = 1,2,\ldots,N_c \tag{4}$$

The k is the category of patterns.

Step2: Calculate the initial cluster centers for every classical domain.

$$V_{kj}^c = \frac{V_{kj}^L + V_{kj}^U}{2} \tag{5}$$

$$V_k^c = \{ V_{k1}^c, V_{k2}^c, \ldots, V_{kNc}^c \} \tag{6}$$

In which $V_k = \left\langle V_{kj}^L, V_{kj}^U \right\rangle$, j is the total number of a matter element model's features. The V_{kj}^L is the upper limits of V_P , and V_{kj}^L is the lower limits of V_P.

Step3: Classical domain of the upper and lower limits of the length of the gene is

$$2^{n1} < \left(V_k^c - V_k^L\right) \times d \le 2^{m1} \tag{7}$$

$$2^{n2} < \left(V_k^U - V_k^c\right) \times d \le 2^{m2} \tag{8}$$

$$Gene_k^L = m1 \text{ (bits)} \tag{9}$$

$$Gene_k^U = m2 \text{ (bits)} \tag{10}$$

In the Eq. (15) and (16), d is the resolution by user defined, and is set to 10000 in this paper. The *m1* and *m2* is the length of the gene.

Step4: Combine all the genes to the chromosome of sequence 0 and 1.

Step5: Result in a population from binary coding randomly to be the parents that include N groups of chromosome and then turn it into decimal.

$$chrom_{10} = chromosome_{2} \times (\frac{s_k - r_k}{2^m - 1}) + r_k \tag{11}$$

In which N is the number by user defined, and it is set to 20 in this paper.

Step6: Build the pro-generation model.

$$R_k = \begin{bmatrix} N, & c_1, & \langle v_1^L, v_1^U \rangle \\ & c_2, & \langle v_2^L, v_2^U \rangle \\ & \vdots & \vdots \\ & c_{Nc}, & \langle v_{Nc}^L, v_{Nc}^U \rangle \end{bmatrix} \tag{12}$$

Step7: Read the first training patterns.

Step8: Calculate the correlation of matter-element.

$$K_{ij}(v_{tj}) = \begin{cases} \dfrac{-2\rho(v_{tj}, V_{ij})}{|V_{ij}|}, & if \ v_{tj} \in V_{ij} \\ \dfrac{\rho(v_{tj}, V_{ij})}{\rho(v_{tj}, V'_{pj}) - \rho(v_{tj}, V_{ij})}, & if \ v_{tj} \notin V_{ij} \end{cases} \tag{13}$$

Where $|V_{ij}|$ is obtained from s_p-r_p.

$$\lambda_i = \sum_{j=1}^{N} W_{ij} K_{ij} \quad , \ i = 1,2,...,m \tag{14}$$

$$\lambda'_i = \frac{2\lambda - \lambda_{min} - \lambda_{max}}{\lambda_{max} - \lambda_{min}} \ , \ i = 1,2,...,m \tag{15}$$

$$\text{If } (\lambda'_k = 1) \text{ THEN (the fault type is k)} \tag{16}$$

Step9: Repeat step7-step8 until the entire pattern is completed and then goes to next step.

Step10: Calculate the accuracy rate of the chromosome (individual).

Step11: Repeat step7-step10 until the all chromosomes have been processed, and then go to next step.

Step12: Calculate the fitness of every chromosome.

$$\text{Fitness} = \frac{S}{H} \tag{17}$$

The S is the total correct number. H is the total number of training patterns.

Step13: Choose parental chromosome to execute crossover by roulette wheel selection. The rate of the chosen parents is:

$$P_k = \frac{F_k}{\sum_{i=1}^{N} F_i} \tag{18}$$

The F_k is the fitness of an individual. P_k is the percentage of the selected

Step14: Set the chosen filial generation which is obtained from step13 into the mating pool, and makes mutation.

Step15: Repeat step7-step14 until the propagation of the last generation or convergence, then the training epoch is finished.

3 Experiment Results and Discuss

The circuit under test (CUT) was used to illustrate our method as shown in Fig. 1, it is a low-pass filter that passes low-signals but attenuates signals with frequencies higher than the cutoff frequency. In this paper, the DC gain of the filter is set to 1 and the cutoff frequency is set to 1 kHz. Node 1 and 2 are the testing node where voltage can be measured or simulated. The nominal values and the tolerances of the components are summarized in Table 1.

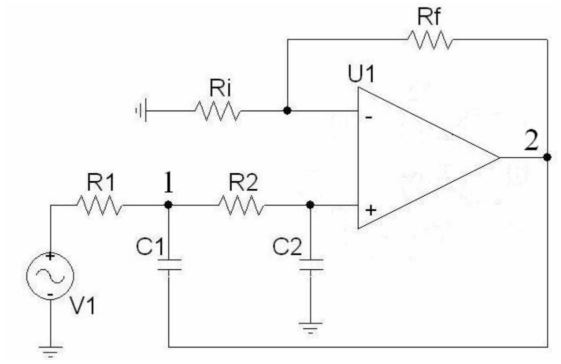


Fig. 1. The second-order low-pass filter

Table 1. The nominal value of low-pass filter

Components	Nominal values	Tolerance (%)
R1	100KΩ	±5
R2	85KΩ	±5
Ri	2.2MΩ	±5
Rf	0.1KΩ	±5
C1	5nF	±5
C2	2nF	±5

3.1 Testing the Ability to Training

The CUT was simulated both at normal and all faulty conditions by using PSPICE software, the tolerance rang of normal was selected deviation of about ±5% of the nominal values, and soft fault was deviation of about ±20%. By analyzing sensitivity of the CUT, R1, R2, C1 and C2 are selected to be the fault components, consequently, 13 fault classes are definite, and Table 2 shows the typical circuit signal for 9 kinds of faults. In this research was discovered that the transient voltage (V1, V2) and phase spectrum (P1, P2) were different between normal and other faults condition at nodes, the RMS value of transient voltage and phase spectrum can made the pattern of CUT in the training stage, every fault is produced 50 sets with the Monte-Carlo method, there are total 450 sets of training data. In addition, this paper is produced another 30 sets for each fault to test the stability of the fault diagnostic system, hence there are total 270 sets of testing data.

In the paper, the parameter of EGA is setting the tolerance of error rate to be 0.01, the crossover rate is 0.8, and the mutation rate is 0.1. The learning times are set to 100 epochs that can give convergence result. In the training stage, the rate of highest accuracy is 98.4%. Fig. 2 shows the training curve of the rate of highest accuracy for each epoch. When the training stage of the EGA has been completed, then the identifying stages of the proposed method can be started for fault diagnosis, so this paper design a user interface of fault diagnosis window for this analog circuit by the Microsoft Visual Basic (VB), it can diagnose fault quickly by analyzing 4 circuit signals and to show what kind of fault in the analog circuit. For example, Fig. 3 shows a diagnostic fault in R2 and also shown in the fault displayed window by a twinkling red light to alert user which one circuit element was happening fault.

Table 2. The typical circuit data with different fault types (Partial samples)

Cases	V_1	V_2	P_1	P_2	Faults no.	Actual fault type
20	7.749	14.434	-93.204	-151.301	1	Normal
70	9.923	16.691	-97.915	-156.404	2	R1 over nominal
120	9.924	16.691	-88.829	-147.318	3	R1 below nominal
187	7.531	16.560	-91.174	-156.134	4	R2 over nominal
236	7.685	11.709	-96.250	-143.379	5	R2 below nominal
285	7.143	13.658	-101.683	-159.625	6	C1 over nominal
335	7.5005	13.853	-84.7055	-142.6476	7	C1 below nominal
383	7.527	14.353	-87.139	-152.612	8	C2 over nominal
429	7.983	13.980	-102.364	-149.876	9	C2 below nominal

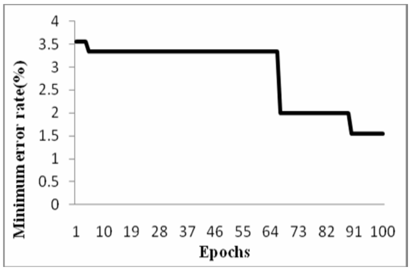
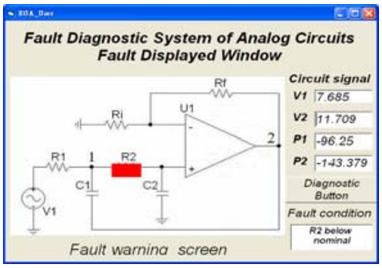


Fig. 2. The minimum error rate of training curve



Circuit signal
V1 7.685
V2 11.709
P1 -96.25
P2 -143.379
Fault condition
R2 below nominal

Fig. 3. User interface of fault diagnosis system

3.2 Testing Results

Table 3 shows the accuracy by using the multilayer neural network (MNN), k-means, ET, and the proposed EGA based method to diagnosis the soft fault of tested circuit. The maximum testing accuracy is 92.3% in the MNN, 84.67% in the k-means based method and 92% in the traditional extension method. The testing accuracy of proposed method is 96.2%. It is clearly to show the proposed EGA-based diagnosis method to be better than other methods in the both training and testing stages. Moreover, the training times of proposed method is also less than the MNN.

Table 3. Diagnosis performances of different methods

Methods	Training times (Epochs)	Training accuracy (%)	Testing accuracy (%)
Proposed method	100	98.6%	96.2%
Extension method	N/A	N/A	92%
K-means	N/A	N/A	84.67%
MNN(4-7-9)	1000	86.2%	84.6%
MNN(4-8-9)	1000	87.6%	85.7%
MNN(4-9-9)	1000	95.8%	92.3%

4 Conclusions

This paper presents a novel fault diagnosis method based on EGA for analog circuits. Compared with other traditional AI methods, the proposed EGA-based method can achieve the higher accuracy. The calculation of the proposed diagnosis algorithm is also fast and simple. It is feasible to implement the proposed method in a Microcomputer for portable fault detecting devices. This new approach merits more attention, because EGA deserves serious consideration as a tool in diagnosis or classified problems. We hope this paper will lead to further investigation for industrial applications.

Acknowledgments

The author gratefully acknowledges the support of the National Science Council, Taiwan, ROC, under the grant no. NSC-98-2221-E-167-028.

References

1. Yongkui, S., Guangju, C., Hui, L.: Analog Circuits Fault Diagnosis Using Support Vector Machine. In: International Conf. Communications, Circuits and Systems, ICCCAS 2007, pp. 1003–1006 (2007)
2. Catelani, M., Fort, A., Alippi, C.: A Fuzzy Approach for Soft Fault Detection in Analog Circuits. Meas. 32, 73–83 (2002)
3. Slamani, M., Kaminska, B.: Fault Observability Analysis of Analog Circuits in Frequency Domain. IEEE Trans. Circuits Systems II, Analog and Digital Signal Proc., 134–139 (1996)
4. Tadeusiewicz, M., Halgas, S., Korzybski, M.: An Algorithm for Soft-Fault Diagnosis of Linear and Nonlinear Circuits. IEEE Trans. Circuits Systems, Fundamental Theory and Applications 49(11), 1648–1653 (2002)
5. Catelani, M., Fort, A.: Soft Fault Detection and Isolation in Analog Circuits: some Results and a Comparison between a Fuzzy Approach and Radial Basis Function Networks. IEEE Trans. Instrumentation and Meas. 51, 196–202 (2002)
6. Yanghong, T., Yigang, H., Chun, C., Guanyuan, Q.: A Novel Method for Analog Fault Diagnosis Based on Neural Networks and Genetic Algorithms. IEEE Trans. Nstrumentation and Meas. 57(11), 2631–2639 (2008)
7. Spina, R., Upadhyaya, S.: Linear Circuit Fault Diagnosis Using Neuron Morphed Analyzers. IEEE Trans. Circuits Systems II, Analog Digital Signal Proc. 44(3), 188–196 (1997)
8. Cai, W.: The Extension Set and Incompatibility Problem. J. of Scientific Exploration 1, 81–93 (1983)
9. Wang, M.H.: A Novel Extension Method for Transformer Fault Diagnosis. IEEE Trans. Power Delivery 18(1), 164–169 (2002)
10. Wang, M.H.: Application of Extension Theory to Vibration Fault Diagnosis of Generator Sets. IEE Proc. Generation, Transmission and Distribution 151(4), 503–508 (2004)
11. Wang, M.H., Tseng, Y.F., Chen, H.C., Chao, K.H.: A novel clustering algorithm based on the extension theory and genetic algorithm. Expert Systems With Applications 36(4), 8269–8276 (2009)
12. Wang, M.H., Ho, C.Y.: Application of Extension Theory to PD Pattern Recognition of High Voltage Current Transformers. IEEE Trans. Power Delivery 20(3), 1939–1946 (2005)
13. Renner, G., Ekárt, A.: Genetic algorithms in computer aided design. Computer-Aided Design 35(8), 709–726 (2003)
14. Hwang, S.F., He, R.S.: A hybrid real-parameter genetic algorithm for function optimization. Advanced Engineering Informatics 20(1), 7–21 (2006)

A Collision Detection Algorithm Based on Self-adaptive Genetic Method in Virtual Environment

Jue Wu[1,2], Lixue Chen[1], Lei Yang[2], Qunyan Zhang[1], and Lingxi Peng[3,*]

[1] College of Computer Science and Technology, South West Petroleum University, Chengdu, Sichuan, 610500, China
wujue826@yahoo.com.cn
[2] College of Computer science and technology, South West University of Science and Technology, MianYang, SiChuan, 621010, China
[3] College of Computer science and education software, Guangzhou university, Guangzhou, Guangdong, 510006, China
scu.peng@gmail.com

Abstract. Collision detection is very important to enhance the sense of reality and immersion in virtual environment. Most of the traditional collision detection algorithms have been analyzed, but there is no algorithm that is applicable to all situations, and with the scene complexity increases, the efficiency of the algorithm tends to decline rapidly. In this paper, a new method is proposed to solve the problems: converting the problem of collision detection to the nonlinear programming problem with constraint conditions, and then using the adaptive genetic algorithm to solve it. The experiment results show that this method is efficient, especially in large-scale scenes.

Keywords: collision detection, genetic algorithm, non-linear programming.

1 Introduction

Collision detection is a very important problem in Virtual Reality (VR), robotics, computer-animation, geometry algorithm, CAD/CAM, and so on[1]. Collision detection is based on a ubiquitous factor in real life, which is that two objects can not share the same space [2].

The hierarchical bounding box is a very popular method in collision detection. The method is presented by Clark in 1976. The method uses boxes with a bigger and simpler volume bounds the object. We use the bounding boxes to detect the collision when the collision detection is needed in a scene. This method is very effective when the objects in the scene are not intersected with each other. However we should think about the simplicity and tightness when we choose the bounding box, because the intersection of the bounding box does not mean that the objects are intersected with each other. The simplicity and tightness of the bounding box is a pair of contradiction. The simpler the bounding box is, the worse the tightness of bounding

[*] Corresponding author.

box is. The bounding box can be divided into AABB[4], OBB[5], DOPs[6] and so on by the. These algorithms are used in different situation respectively and no algorithm is applicable to all situations. On the other side, the efficiency of most collision detection algorithms is declining when the scenes become complicated and massive, which makes the real-time and stability of collision detection uncertain.

In allusion to the problems mentioned above, the paper presents a collision detection algorithm based on adaptive genetic method (CDAG) and the result of the experiments shows that this algorithm is feasible, and it is much more effective in the larger scene than the bounding box.

2 Algorithmic Model

The CDAG algorithm converts the problem of the collision detection to the non-linear programming problem with constraint conditions. And use the adaptive genetic algorithm to solve the problem. There are many traditional methods to resolve the non-linear programming problem, such as quadratic programming, sequence quadratic programming, penalty function method, gradient projection method and so on. But all these methods have limitations and they have they own scope of application. There is no method which can be used to all situations up to now. However, the robustness of genetic algorithm and the ability of adaptive search are very good. The algorithm can find optimal solution in space which is complex, uncertain and with multi-extreme point. Firstly, the following definition is given.

2.1 Related Definitions

Definition 1: give m points $x_1,x_2,\ldots x_m \in R^m$ and real number $\lambda_1,\lambda_2,\ldots\lambda_m$, we call $\lambda_1 x_1+\lambda_2 x_2+\ldots\lambda_m x_m$ is the linear combination of points $x_1,x_2,\ldots x_m$, especially when $\lambda_1+\lambda_2+\ldots+\lambda_m=1$, and $\lambda_1,\lambda_2\ldots\lambda_m\geq 0$. We call the $\lambda_1 x_1+\lambda_2 x_2+\ldots\lambda_m x_m$ is the convex combination of points $x_1, x_2, \ldots x_m$. The R^m means n dimensions.

Definition 2: Supposed that $s \in R^m$, a convex hull which is combined with random finite points in s signed as $H(s)$, it is formula (1).

$$H(s)=\left\{\lambda_i \geq 0, i=1,2,\ldots m, \sum_{i=1}^{m}\lambda_i = 1, m\in N^+\right\} \tag{1}$$

The N^+ means positive integer set.

Definition 3: defined that $Min\ d_{A,B}= \| x-y \|$ presents the shortest distance between two convex polyhedrons. x is a point on object A. y is a point on object B. On account to the theory described above, we can formulate the shortest distance between two objects as formula (2).

$$Min\ d_{A,B} = \left\| \sum_{i=1}^{m}\lambda_i x_i - \sum_{j=1}^{n}\sigma_j y_j \right\| \tag{2}$$

The $\sum_{i=1}^{m} \lambda_i x_i$ means a point x on object A, the $\sum_{j=1}^{n} \sigma_j y_j$ means a point y on object B. λ_i and σ_j satisfies the conditions as formula (3) and (4).

$$\sum_{i=1}^{m} \lambda_i = 1, and \lambda_i \geq 0, i = 1, 2...m \tag{3}$$

$$\sum_{i=1}^{n} \sigma_i = 1, and \sigma_i \geq 0, i = 1, 2...n \tag{4}$$

So by using the convex hull of the vertex to represent the convex polyhedron, the shortest distance problem between objects with the constraint condition is transformed into a nonlinear programming problem. Find the points with the minimum value in Mind$_{A,B}$ which meet the conditions of inequality and equality constraints. Supposed that the points is λ_i and σ_j . *If* Mind$_{A,B}$*=0,* the object A and *B* are collided with each other; if Mind$_{A,B}$ >0 , object A and B are not collided. This problem can be converted to the optimization problem with constraint conditions. We use Collision (A, B) to present the collision between two objects. 0 means that A and B are collided. 1 means that A and B are separated. We use an equation (5) , (6) and (7) to describe this situation.

$$Collision(A,B) = \begin{cases} 0 \; iff \; Min \, d_{A,B} = Min \left\| \sum_{i=1}^{m} \lambda_i x_i - \sum_{j=1}^{n} \sigma_j y_j \right\| = 0 \\ 1 \; else \; Min \, d_{A,B} = Min \left\| \sum_{i=1}^{m} \lambda_i x_i - \sum_{j=1}^{n} \sigma_j y_j \right\| > 0 \end{cases} \tag{5}$$

$$\sum_{i=1}^{n} \lambda_i = 1 \; and \; \lambda_i \geq 0 \tag{6}$$

$$\sum_{i=1}^{n} \sigma_i = 1 \; and \; \sigma_i \geq 0 \tag{7}$$

The planning problem has two equations independently. Therefore, we can eliminate two variables which are λ_n and σ_m. According to the equality constraint elimination method, the problem can be converted to the problem as formula (8),(9) and (10) described.

$$Collision(A,B) = \begin{cases} 0 \; iff \; Min \, d_{A,B} = Min \left\| \sum_{i=1}^{m-1} \lambda_i x_i - \sum_{j=1}^{n-1} \sigma_j y_j \right\| = 0 \\ 1 \; else \; Min \, d_{A,B} = Min \left\| \sum_{i=1}^{m-1} \lambda_i x_i - \sum_{j=1}^{n-1} \sigma_j y_j \right\| > 0 \end{cases} \tag{8}$$

$$0 \le \sum_{i=1}^{i=m-1} \lambda_i \le 1 \, and \, 0 \le \lambda_i \le 1 \tag{9}$$

$$0 \le \sum_{i=1}^{i=n-1} \sigma_i \le 1 \, and \, 0 \le \sigma_i \le 1 \tag{10}$$

2.2 Algorithm Description

In this paper, the problem of collision detection is converted to the non-linear programming problem with restricted conditions. The adaptive genetic algorithm is adopted to solve this problem.

2.2.1 Initial Population Generation

The algorithm uses real-number encoding, supposed that the number of the entity in the population is n, and the x_t^i means the entity i in generation t , i={1, 2,…, n}. The number of gene of each entity L=m, which was composed with m real-number. The entity $x_t^i \in R^m$, x_t^i can presets m dimension row vector, that is x_t^i =(x_t^i (1), x_t^i (2)…x_t^i (m)), thus the t generation can be presents as n×m matrix, X_t= (x_t^1, x_t^2 …x_t^n)^T. Supposed that the number of the entities in the generation is Pop_Size. Generate Pop_Size entities freely in the filed of the feasible solution.

2.2.2 Selection Strategy

The first step of the selection procedure is count the fitness. The fitness function is very important to the genetic algorithm search. The inappropriate fitness function may leads to premature convergence or generates the locally optimal solution but not the globally optimal solution. For the problem of collision detection the target function is the problem of minimum value, hence the fitness function is formula (11).

$$Fit(fit(x)) = \begin{cases} 2 - f(x) & iff \; f(x) < 2 \\ 0 & else \end{cases} \tag{11}$$

2 is the estimation of the maximum value of f(x).

Each entity in the population has a selective probability, which is determined by the fitness and the distribution of the fitness.

The CDAG algorithm uses proportional fitness assignment, in order that the entity with higher fitness has higher survival probability.

The specific steps are as follows.

1) Calculate the fitness of each entity as the fitness function (11) and supposed that the fitness of entity i is singed as f_i.
2) Calculate the selective probability and cumulative odds of every entity.
3) Divided the extent of [0, 1] according to the cumulative odds.

4) Generate a random number between 0 and 1. If the number meets the condition of formula (12), choose entity i to copy.

$$\sum_{j=1}^{i-1} f_j / \sum_{j=1}^{Pop\_size} f_j < \zeta \le \sum_{j=1}^{i} f_j / \sum_{j=1}^{Pop\_size} f_j \tag{12}$$

5) Repeat the step (4) until the number of entity meets the demand.

2.2.3 Crossover and Mutation Strategy

The crossover strategy crosses the number bit-by-bit according to the crossover probability named P_c. First, choose two entities in the population randomly to combine into a pair. Second, choose a position randomly. And then generate a random number in [0, 1]. If the random number is greater than P_c, cross the number at the selected position, otherwise keep the original state.

Mutation is a local random search method. Combined with the crossover, it can satisfy the efficiency of the genetic algorithm It also makes genetic algorithm have the ability of random search. At the same time, mutation can make the genetic algorithm maintain the diversity to prevent premature convergence. First of all, select an entity and the position to mutate. Secondly, generate a random number in [0. 1]. If the random number is greater than P_m make the operation of mutation at the selected position, otherwise keep the original state.

The operation of mutation is very simple because of the real-number encoding. Checking a point $X(x_1, x_2)$ in the search space, keep the x_2 unchanged, and then change the value of x_1 , the variation range of x_1 is $[0, 1-x_2]$. On considering of crossover, supposed that there are two points $x=(x_1,x_2)$ and $x'=(x_1',x_2')$. The random linear combination of the $\lambda x+(1-\lambda)x'$ $(0\le\lambda\le1)$ is also a point of the search space, and which meets the constraint condition. Thus we finish the operation of crossover. Because the search space is convex, this genetic manipulation makes the solution vector in the set of the feasible solution.

The crossover probability P_c and the mutation probability P_m can be adjusted as formulas (13) and (14).

$$P_c = \begin{cases} P_{c1} - (P_{c1} - P_{c2})(f' - f_{avg})/(f_{\max} - f_{avg}) & f \ge f_{avg} \\ P_{c1} & f < f_{avg} \end{cases} \tag{13}$$

$$P_m = \begin{cases} P_{m1} - (P_{m1} - P_{m2})(f' - f_{avg})/(f_{\max} - f_{avg}) & f \ge f_{avg} \\ P_{m1} & f < f_{avg} \end{cases} \tag{14}$$

The $P_{c1}=0.9$, $P_{c2}=0.6$, $P_{m1}=0.1$, $P_{m2}=0.001$. f' means the larger fitness of the two entity crossed. f_{max} means the largest fitness in the population. f_{avg} means the average fitness in the population.

3 Simulation Experiment

The experiment uses two three-dimensional scenes. The scene 1 includes two ring cycles. It is composed of 5000 triangles. The scene 2 includes two ring cycles too, but it is composed of 22000 triangles. According to the distance function between objects, the problem-solving model is as equation (15). The x_i, y_i are the coordinate information of the points which have been known.

$$f(x,y)=\sqrt{(\sum_{i=1}^{n}\lambda_i x_{i1}-\sum_{j=1}^{m}\sigma_j y_{j1})^2+(\sum_{i=1}^{n}\lambda_i x_{i2}-\sum_{j=1}^{m}\sigma_j y_{j2})^2(\sum_{i=1}^{n}\lambda_i x_{i3}-\sum_{j=1}^{m}\sigma_j y_{j3})^2} \tag{15}$$

For the CDAG algorithm, the point coordinate information of object A and B are known. Hence the main problem to resolve the optimization problem is the time complexity. In the algorithm simulation, the number of the entity in the population is set as 50, the max generation set as 500, the fitness function uses the formula (11), the selection strategy uses the formula (12), the crossover probability and mutation probability are calculated as formula (13) and (14).

The traditional collision detection algorithms are bounding box, the CDAG algorithm model converts the collision detection problem to the non-linear programming problem. And there are many traditional methods to resolve the non-linear programming problem, such as the penalty function method, gradient projection method quadratic programming method and so on. In order to check the efficiency of the CDAG algorithm, the CDAG algorithm is compared with the AABB algorithm and gradient projection method which are used broadly in the experiment. The experiment environments are described as following: the basic frequency of the PC is P4.6G, the tool is Visual C++6.0. Every algorithm has been run 10 times. The average time of the algorithms is compared to each other. The result is presented in figure 1 and figure 2.

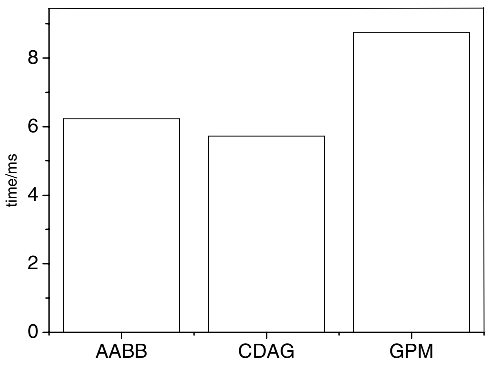


Fig. 1. The performance comparison in a simpler scene

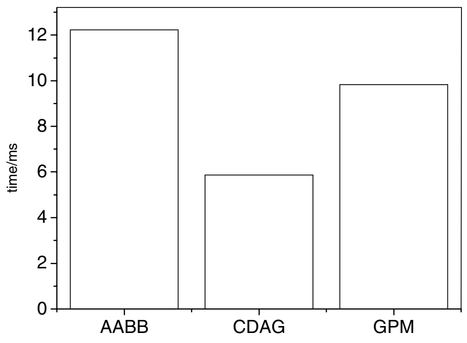


Fig. 2. The performance comparison in a more complex scene

From the two figures we can draw a conclusion that in the simpler scene 1 the efficiency of CDAG algorithm increased by 8.1% and 34.8% compared to the AABB algorithm and the gradient projection method respectively. In the more complex scene 2, CDAG algorithm increased by 40.3% and 52% compared to the AABB algorithm and the gradient projection method respectively. The result of the experiment shows that the adaptive genetic algorithm is superior to the gradient projection method and the AABB algorithm. The superiority is not so distinct in simple scene, but when the scene is becoming more and more complex, the advantage is much more distinct.

4 Conclusion

In this paper, we convert the collision detection problem to the problem of finding the shortest distance between two objects and use the adaptive genetic algorithm to solve this problem. From the simulation model we can draw a conclusion that the speed of the CDAG is faster than AABB method and the GPM. The result of the experiment shows that the CDAG is feasible, and this model is much more efficient when used in complex scenes than traditional algorithms.

References

1. Shi, J.-Y.: The Basic Technology and the Practical Algorithm of Virtual Reality. Scientific Press, Beijing (2002)
2. Wei, Y.-M., Wu, Y.-Q., Shi, J.-Y.: Research on Fixed Direction Hull Bounding Volume in Collision Detection. Journal of Software 12(7), 1056–1063 (2001)
3. Ma, Y.-J., Ma, Y.-D., Jiang, Z.-Y., Sun, Q.-G.: Fast genetic Algorithm and its Convergence. Systems Engineering and Electronic Technology 3(3), 714–718 (2009)
4. Hubbud, P.M.: Approximation Ployhedra with Sphere for Time-critical Collision Detection. ACM Trans. Graph. 15(3), 179–210 (1996)
5. Palmer, I.J., Grimsdale, R.L.: Collision Detection for Animation Using Shpere-Trees. Computer Graphics Forum 14(2), 105–116 (1995)

6. Gottschalk, S., Lin, M., Manocha, D.: OBBTrees: A Hierarchical Structure for Rapid Interference Detection. In: Proceedings of ACM Siggraph 1996, pp. 171–180 (1996)
7. Liu, C.-a., Wang, Y.-P.: Evolutionary algorithm for constrained multi-objective optimization problems and its convergence. Systems Engineering and Electronic Technology 29(2), 277–280 (2007)
8. Lin, M.C., Manocha, D.: Fast interference detection between geometric models. The visual Computer 11(10), 542–561 (1995)
9. Gilbert, E.G., Johnson, D.W., Keerthi, S.S.: A fast procedure for computing the distance between complex objects in three dimensional space. IEEE Trans. on Robotics and Automation 4(2), 193–203 (2007)

A Non-dominated Sorting Bit Matrix Genetic Algorithm for P2P Relay Optimization

Qian He[1,2], Junliang Chen[1], Xiangwu Meng[1], and Yanlei Shang[1]

[1] State Key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications, Beijing 100876, China
[2] Network Information Center, Guilin University of Electronic Technology,
Guilin, 541004 China
treeqian@gmail.com, {chjl,mengxw,shangyl}@bupt.edu.cn

Abstract. Cooperative caching and relaying content in ISPs can decrease the bandwidth costs and distribution time. The relay resources installed at ISP are limited and the upload rates of relay servers are various. After formulating the optimization problem, we design a Non-dominated Sorting Bit matrix Genetic Algorithm (NSBGA) to solve it. Constraint-satisfied population is initialized according to resource ratio dynamically; improved alone point crossover and symmetric mutation is designed; population is non-dominated sorted. The experiments show that NSBGA is better than NSGAII and it can support P2P relay optimization very well. The relations between performances and parameters as the numbers of ISPs, source channels and relay servers are analyzed. As a general optimization algorithm, NSBGA also can be used in other application fields.

Keywords: Genetic Algorithm, Peer-to-Peer, Non-dominated Sorting, Bit Matrix, Cooperative Relay.

1 Introduction

The fundamental advantage of peer-to-peer (P2P) content distribution is to allow peers to contribute their upload bandwidth, such that bandwidth costs may be saved on dedicated streaming servers. Server bandwidth cost savings and the speedup of distribution are more substantial when the data deliver among the end peers and their Internet Service Providers (ISPs). However, the heavy P2P traffic has affected the whole network performance—congestion, for example [1]. How to decrease the costs of P2P distribution becomes a very important problem for ISPs, and some ISPs have started to restrict the P2P traffic.

Genetic Algorithm (GA) is a powerful tool to solve combinatorial optimizing problems [2]. GA is based on the concepts of natural selection and reproduction and do not guarantee to obtain the optimal solution, however they provide appropriate solutions to a wide range of optimization problems which other deterministic methods find difficult or impossible to solve. Furthermore, GA, at least in most implementations, has the advantage that they do not require any gradient information and have intrinsic parallelism in searching the design space, thus making them robust and scalable optimization techniques.

The caching and relaying strategy can affect the performance of P2P content distribution significantly [3-5]. In real networks, the relay recourses are limited and the upload rates of ISPs' relaying servers are various. As an ISP, it is useful to ensure high quality of services of global networks through an appropriate relay strategy. Because the problem of how to allocate resources effectively is highly nonlinear, the methods as Gauss Newton and Simplex are not suitable. This paper proposes the application of GA for the optimal relay strategy to solve the limitation of relay recourses and to reach the P2P distribution growth requirements. The original and corporate servers' bandwidth costs are decreased as little as possible by using a special peers' fetching mechanism in which peers assist to distribute and fetch the content as near as possible. The optimal model of relaying strategy aims to minimize the average distribution time and the distribution time of the slowest content channel, which can represent networks' global and fair distribution performance. To solve this multi-objective optimization involves two types of difficulties: multiple conflicting objectives and a highly complex search space. In these cases instead of a single optimal solution competing goals give rise to a set of compromise solutions, generally denoted as Pareto-optimal [6]. NSGAII [6], which is an improved non-dominated sorting genetic algorithm, has been successfully applied to solving many real problems [7-9]. In this paper, we developed a Non-dominated Sorting Bit matrix Genetic Algorithm (NSBGA) based on NSGA-II to obtain optimal relaying strategy.

The rest of this paper is organized as follows. Section 2 presents the relaying architecture and the constrained model. In Section 3, NSBGA is designed. Section 4 evaluates NSBGA. Finally, we conclude the whole works in Section 5.

2 Approximate Relaying Model

The architecture we considered is built on relay nodes deployed at ISPs. There are a lot of original channel sources providing media for customers. Each ISP has a finite number of relay servers, and each relay server (RS) can be used to relay one steaming channel to an arbitrary number of peers in a p2p fashion. The control server collects the architecture's information including the number of relaying servers, the upload rate of different RSes with respect to various ISPs, and the popularity of channels; then, it allocates the relaying strategy of RSes periodically. In this architecture, the end nodes can download from the local, neighbors or source. If a requested object is not available in the local RS, the customer will try to fetch from the neighboring RSes. When the neighboring RSes also don't have the object, the peer will access the original servers directly. The special peers' fetching mechanism, in which peers access the surrogate server as nearer as possible, can decrease the original and corporate servers' bandwidth costs effectively. The relaying networks architecture is shown in Fig. 1.

Because the relay resources are limited, we should design a method to allocate them in ISPs for caching and relaying channels more effectively. Our method

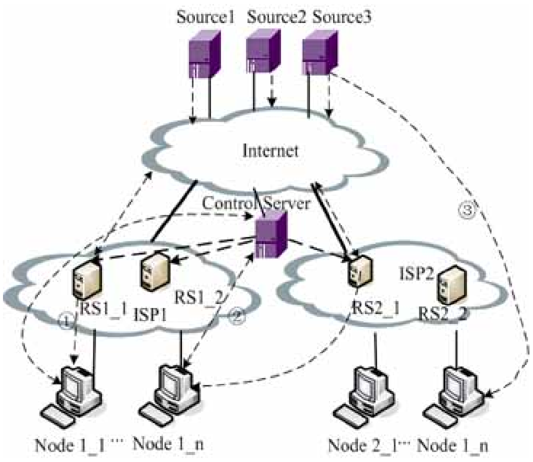


Fig. 1. Relaying networks architecture

aims to minimize the average distribution time and the distribution time of the slowest channel. The allocation strategy can be updated timely through running an optimization algorithm periodically on the control server. If the allocation policy should be changed in a RS, it will be notified by the control server.

Each RS of ISP i has an upload bandwidth $u_{i,j}$ to ISP j and each leecher has a download bandwidth d_i. T^j_{min} is the minimum distribution time of channel j. The minimum d_i is denoted as d_{min}, the sum of upload bandwidth of seeds is denoted as $U(S)$, and the sum of upload bandwidth of seeds and leechers is denoted as $U(I)$.

Theorem 1. The minimum distribution time for the general heterogeneous peer-assisted file distribution system is [10]

$$T^j_{\min} = F/\min\{d_{\min}, U(I)/L, U(S)\} \tag{1}$$

Table 1. Main notations

Variable	Description
P	The set of ISPs
$\|P\|$	The number of ISPs
H	The set of channels
$\|H\|$	The number of channels
r_j	The rank of channel j
T_j	The distribution time of channel j to the end nodes
$T^{(i,j)}_{min}$	The minimum distribution time of channel j in ISP i
$u_{i,j}$	The element of upload rate matrix U
N_i	Number of relaying resources in ISP i
$k_{i,j}$	The element of relaying matrix

In local networks the upload rate and download rate are normally so fast that d_{min} is greater than $U(S)$ obviously. In addition, as a networks designer, the upload capacity of seeds is our main considered problem. We can get:

$$T^{j}_{\min} = F/U(S) \tag{2}$$

Since the main objectives are to minimize the average distribution time (g_1) and the distribution time of the slowest channel's (g_2), the formal constrained optimization problem can be described as:

$$\begin{aligned}
&\min : g = (g_1, g_2)\\
&g_1 = \frac{\sum_j \sum_i r_j * T^{(i,j)}_{\min}}{|H|\cdot|P|}\\
&g_2 = \max(\{\forall i, \forall j, r_j * T^{(i,j)}_{\min}\})
\end{aligned} \tag{3}$$

According to Eq.2, the distribution time is decreasing with the increasing of the upload bandwidth of seeds and $T^{(i,j)}_{min}$ should be minimized when the sum of upload bandwidth of seeds for channel j in ISP i is maximized. Thus, the main objectives become to maximize the average upload bandwidth and the slowest upload bandwidth. If the matrix of the upload rate between various ISPs (U) is known, the sum of one cannel's upload bandwidth of seeds in one ISP is decided by the relaying matrix (K). Since u_0 is the sum of upload rate of source channel which is not affected by the relaying matrix, it can be ignored. The new constrained optimization model can be described as:

$$\begin{aligned}
&\min : f(k) = (f_1(k), f_2(k))\\
&f_1(k) = -\frac{\sum_j \sum_p r_j \cdot V_s^{(p,j)}}{|H|\cdot|P|}\\
&f_2(k) = -\min(\{\forall j, \forall p, r_j * V_s^{(p,j)}\})\\
&V_s^{(p,j)} = \sum_{i \neq p} \delta \cdot k_{i,j} \cdot u_{i,p} + k_{p,j} \cdot u_{p,p}\\
&s.t.\\
&\sum_j k_{i,j} \leq N_i\\
&\delta = 0, \text{if } k_{p,j} > 0\\
&k_{i,j} = 0 \text{ or } k_{i,j} = 1
\end{aligned} \tag{4}$$

In the Eq.4, δ implies that when there is local RS, the leecher will not fetch content from the neighboring RSes. If the ranks r_j are equal for all channels, it can be erased in the objective function. We assume that the number of RSes in each ISP is less than the number of channels. Otherwise, it is best obviously to relay every channel in ISPs.

To find the optimal $k_{i,j}$ is very difficult for the problem is multi-objective, nonlinear and non-differentiable with constraints. To deal with this problem, we propose a heuristic search algorithm using Genetic Algorithm based on non-dominated sorting.

3 Non-dominated Sorting Bit Matrix Genetic Algorithm

The capabilities of multi-objective genetic algorithms (MOGAs) to explore and discover Pareto-optimal fronts on multi-objective optimization problems have been well recognized. Since the element of relaying matrix $k_{i,j}$ is zero or one so that it can be represented by bit directly. We implement a multi-objective optimization algorithm for bit matrix based on NSGAII [6]: NSBGA. The flowchart of which is shown in Fig 2.

Step 1: Generate a uniformly distributed parent population of size in the initial generation; individuals are generated with respect to the number of resources.

Step 2: Evaluate the individuals and obtain the fitness of each individual.

Step 3: Sort the population based on non-domination and crowding-distance. This returns two columns for each individual which are the rank and the crowding distance corresponding to their position in the front they belong.

Step 4: Fill new population of size with the individuals from the sorted fronts starting from the best after non-domination sorting. If a front can only partially fill the next generation, the crowding-distance method to ensure diversity is invoked. The crowding-distance method maintains diversity in the population and prevents convergence in one direction. This procedure is shown in Fig 3.

Step 5: Check the stopping criteria. If the stopping criterion is reached, we will stop the program and then go to Step 9.

Step 6: Generate offspring. Use the usual binary tournament selection. The individual with better fitness is selected as a parent. Tournament selection is carried out until the pool size is filled.

Step 7: Combine the offspring and parents to form new population.

Step 8: Update the number of generations and repeat the steps 2-8 until a stopping criterion is met.

Step 9: The best individual will be sent to the control server. If the refresh cycle is reached, the relaying allocation will be updated timely.

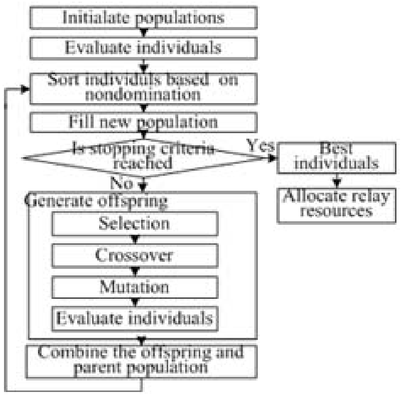


Fig. 2. Flowchart of NSBGA

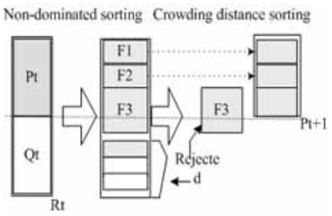


Fig. 3. Filling new population procedure

3.1 Initial Population Policy

During the evolution, the initial population is the basis for genetic algorithm to obtain better fitness. It has an important influence on the final fitness. Hence, an optimized initial population can effectively overcome the situation of "prematurity" and accelerate convergence. The following is the proposed policy.

Firstly, the constraint of optimization should be satisfied when the population is generated randomly in the initialization. Some rubbish individuals which will be weeded out soon are avoided.

Secondly, the distribution of relaying matrix should be sensitive to the summation relaying resources in every ISP.

The relaying matrix element is created by a dynamic probability with respect to the ratio between resources and channels. The special initial population becomes the basis to get higher fitness during the later evolution. The pseudo-code of initialization is illustrated in Algorithm 1.

Algorithm 1. Initialize (pop, popSize,relayResources)

```
//pop is the initial population
//popSize is the size of population
//relayResources is the vector of relaying resources in ISPs
(1) For i=1:popSize
(2)   For j=1 : $|P|$
(3)     tSum=0;
(4)     For jj=1:$|H|$
(5)       initProb=relayResources (j)/ $|H|$ ;
(6)       if(rand(1)¡ initProb && tSum¡relayResources (j))
(7)         pop[i][j,jj]=1;
(8)         tSum++;
(9)       else
(10)        pop[i][j,jj]=0;
(11)      endif
(12)    End For
(13)  End For
(14) End For
```

3.2 Special Genetic Operator

In NSBGA, special crossover and mutation operators are designed to improve the performance of GA. The crossover is designed based on the classical alone point crossover (APC) algorithm. In order to avoid breaking constraints, we choose entire row of relay matrix to exchange between two parents. Since considering constrained condition is unnecessary, it can simplify the non-dominated sorting and ensure the offspring has good properties. The improved crossover is described in Algorithm 2.

In the mutation operator, we always choose two symmetric positions to change simultaneously. Firstly, a position in each row is selected randomly, and then a position where the value is opposite to that in the first position is selected. This special mutation operator can ensure the offspring will not break the constraint: $\sum_j k_{i,j} \leq N_i$. Algorithm 3 gives the pseudo-code of symmetric mutation.

Algorithm 2. Crossover (p1, p2,c1, c2)

```
//p1, p2 are the selected in tournament
//c1, c2 are the offspring individuals
(1) c1=p1;
(2) c2=p2;
(3) V=$|P|$*$|H|$;
(4) Choose site1 and site2 randomly in [1,V]; site1<site2;
```

```
(5)row1=floor(site1/$|H|$)+1;
(6) row2=floor(site2/$|H|$)+1; (7) For i=row1:row2 (8) For j=1:$|H|$
(9) c1[row1,j]=p2[row1,j]; (10) c2[row2,j]=p1[row2,j];
(11)End For (12) End For
```

Algorithm 3. SymMutation (p, c)

```
//p is the selected individuals using tournament
//c is the offspring individuals after crossover.
(1) c=p; (2) For i=1 : $|P|$ (3) Choose j randomly
(4) Choose j1 randomly, where $p[i,j1] = \overline{p[i,j]}$;
//Update the value of position (i,j) and (i,j1) (6)$c[i,j] = \overline{p[i,j]}$ ;
(7) $c[i,j1] = \overline{p[i,j1]}$;
(8) End For
```

4 Performance Evaluations

To verify the efficient of the proposed cooperative relay strategy, numerous simulations had been performed on several relaying networks topologies. All experiments were taken on same software and hardware, which were Pentium Dual 1.8 GHz processor (E2160), 2 GB of RAM, Windows XP Pro SP2, and MATLAB 2008a. We construct the network topologies for ISPs where there are links between each others. If ISP i can't communicate with j directly, $u_{i,j}$ and $u_{j,i}$ are zero in the matrix of the upload rate. We assume the numbers of relaying resources in every ISP are same (N) and N can be different in various experiments.

4.1 Process of NSBGA-Based Solution

In the first experiment, we set $|P| = 8, |H| = 6$ and $N = 3$, then generate the upload matrix U randomly in the range $[0, 0.1, 0.2, ..., 1]$ Mbps. Because the upload rate of local RS is faster than others, $u_{i,i}$ always equals to 1. Generally, the upload rates are symmetrical between two ISPs, so U is equal to its transpose U^T. In the experiment, the crossover probability is set to $P_c = 0.9$ and the mutation probability P_m equals to $2/|H|$. In this experiment P_c is the same as that suggested in [6], but P_m is bigger than that in [6] (where $P_m = 1/(|P|*|H|)$). Fig 4 shows the running process of NSBGA. The program converges in the 61st generation and gets the individual.

4.2 Experiments on Initial Population and Mutation Policies

The special initialization and genetic operations are tested in our experiments. The parameters of NSBGA are similar to the first experiment, and we assume that the local upload rates are 0.5 Mbps, the neighbor upload rates are 0.1 Mbps and the relaying resources on each ISP are N. We found that the original algorithm as NSGAII [6] may often break constraints and bring some bad individuals

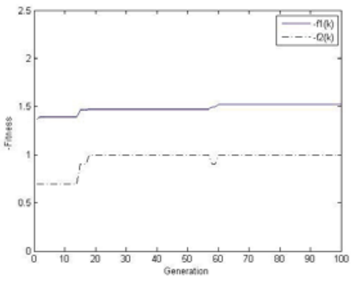


Fig .4. Running process of NSBGA

but NSBGA doesn't. If N is small, the original initialization generates so much bad individuals that GA often can't obtain solutions successfully. In the case of $|P| = 16, |H| = 54$ and $N = 18$, the original NSGAII can't get effective schema, the average speed $(-f1(k))$ and the slowest speed $(-f2(k))$ are always zero. Since the elitism is used in NSGAII which can preserve the excellent individuals, the affection of bad offspring caused by original genetic operations is limited. Fig. 5 compares the performance of NSGAII and NSBGA. The original NSGAII randomly generate initial individuals and the finesses are normally zero (which means too poor), so we replace its initiate policy by ours in experiments. We use our improvement initialization algorithm in NSGAII (NSGAII+CI) to compare NSBGA. The experiments show that NSBGA is better than NSGAII and the special initial and genetic algorithms are effective.

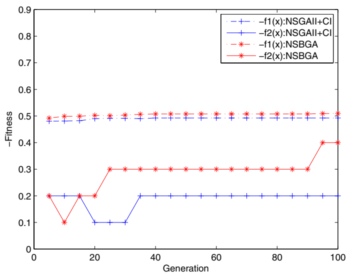


Fig. 5. NSBGA VS NSGAII

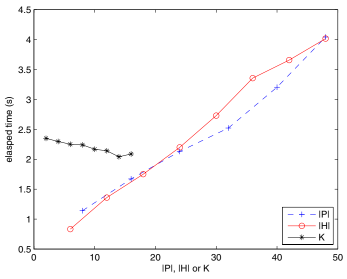
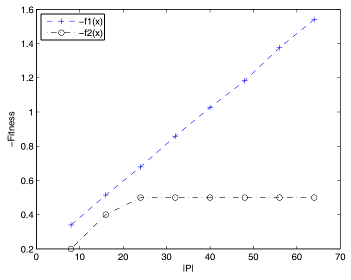


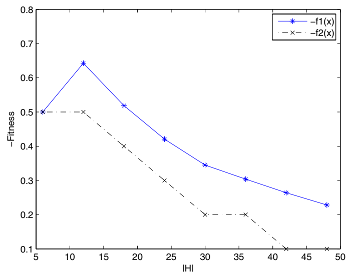
Fig. 6. Elapsed time of NSBGA

4.3 Experiments on Initial Population and Mutation Policies

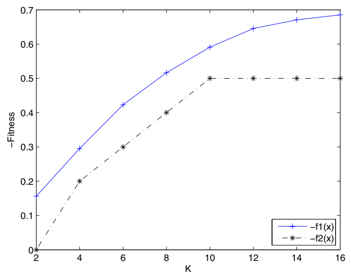
The relations between these parameters of network topology and the performance of NSBGA are studied. Fig. 6 gives the elapsed time of NSBGA in various ISPs' number ($|P|$), the channels' number ($|H|$) and N. Under various experiments, the optimization value is obtained and the elapsed time is acceptable.



(a) $|H| = 12$ and $N = 4$



(b) $|P| = 16$ and $N = 6$



(c) $|P| = 16$ and $H = 24$

Fig. 7. Fitness value for various $|P|, |H|$ and N

When N is increased, the elapsed time decreases little but the effect is not obvious. The elapsed time increases rapidly with increased $|P|$ and $|H|$, and the effects of these two factors on elapsed time are nearly similar. It shows that larger network and more source channels means longer processing time but increasing the relaying resources on ISPs will not prolong the completing time.

The objective fitness value under different network topology is illustrated in Fig. 7. Firstly, $|H|$ and K are fixed, and then $|P|$ is varied from 8 to 64. The average speed $(-f1(k))$ almost increases linearly with the increasing of the number of ISPs. The slowest speed $(-f2(k))$ increases at the beginning and then converges in the local upload rate (0.5). It means that the NSBGA can be effective to utilize the cooperative resources to speed up distribution. This result is shown in Fig. 7a. Secondly, $|P|$ and K are fixed, and then $|H|$ is varied from 6 to 48. The average speed and the slowest speed normally decrease with the increasing of the number of source channels. Notably, when $|H| = 12$, the average speed is larger than before. It is because that in the case of $|H| = 6$, we can get a schema by which all the data can be fetch from local relaying server. Thirdly, $|P|$ and $|H|$ are fixed, and then N is varied from 2 to 16. The average speed increases lower than linear with the increasing of N. The slowest speed increases at first and then converges in the local upload rate (0.5). Fig. 7b and Fig. 7c tell us if we want to ensure the system performance after $|H|$ is increased, we must increase the relaying resources correspondingly.

5 Conclusion

In this paper, we design NSBGA to solve the optimization of utilizing relaying resources to speedup p2p content distribution. Experiments tell us the number of ISPs and relayed channels can affect the elapse time obviously and the number relaying resources should be proper to the number of relayed channels. Our results show that the average time and slowest time can be optimized effectively by the resource allocation based on NSBGA. There is important actually guide meaning on this NSBGA relay strategy for ISP-aided P2P designers and NSBGA can also be used in lots of other multi-objective optimization fields.

Acknowledgments. This paper was supported by the National Natural Science Foundation of China under Grant No.60432010, 60872051 and the National Grand Fundamental Research 973 Program of China under Grant No.2007CB307103.

References

[1] Karagiannis, T., Broido, A., Brownlee, N., Claffy, K., Faloutsos, M.: Is P2P dying or just hiding? In: IEEE Globecomm 2004, Dallas (2004)

[2] Srinivas, M., Patnaik, L.M.: Genetic algorithms: a survey. Computer 27, 17–26 (1994)

[3] Shen, G., Wang, Y., Xiong, Y., Zhao, B.Y., Zhang, Z.: HPTP:Relieving the tension between ISPs and P2P. In: The 6th International Workshop on Peer-to-Peer Systems (IPTPS 2007), Bellevue, WA, USA (2007)

[4] Aggarwal, V., Akonjang, O., Feldmann, A.: Improving user and ISP experience through ISP-aided P2P locality. In: IEEE Infocomm 2008, Phoenix, AZ, USA (2008)

[5] Dan, G.: Cooperative Caching and Relaying Strategies for Peer-to-peer Content Delivery. In: The 7th International Workshop on Peer-to-Peer Systems (IPTPS 2008), Tampa Bay, Florida (2008)

[6] 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)

[7] Deb, K., Mitra, K., Dewri, R., Majumdar, S.: Towards a better understanding of the epoxy-polymerization process using multiobjective evolutionary computation. Chemical engineering science 59, 4261–4277 (2004)

[8] Favuzza, S., Ippolito, M.G., Sanseverino, E.R.: Crowded comparison operators for constraints handling in NSGA-II for optimal design of the compensation system in electrical distribution networks. Advanced Engineering Informatics 20, 201–211 (2006)

[9] Kumar, Y., Das, B., Sharma, J.: Service restoration in distribution system using non-dominated sorting genetic algorithm. Electric ower Syst. Res. 76, 768–777 (2006)

[10] Kumar, R.: Measurement, Modeling and Economics of P2P Systems, PHD. Polytechnic University, New York (2006)

Fast Parallel Memetic Algorithm for Vector Quantization Based for Reconfigurable Hardware and Softcore Processor

Tsung-Yi Yu, Wen-Jyi Hwang*, and Tsung-Che Chiang

Department of Computer Science and Information Engineering,
National Taiwan Normal University, Taipei, 117, Taiwan
{g97470731,whwang,tcchiang}@csie.ntnu.edu.tw

Abstract. A novel parallel memetic algorithm (MA) architecture for the design of vector quantizers is presented in this paper. The architecture contains a number of modules operating memetic optimization concurrently. Each module uses steady-state genetic algorithm (GA) for global search, and K-means algorithm for local refinement. A shift register based circuit for accelerating mutation and crossover operations for steady state GA operations is adopted in the design. A pipeline architecture for the hardware implementation of K-means algorithm is also used. The proposed architecture is embedded in a softcore CPU, and implemented on a field programmable logic array (FPGA) device for physical performance measurement.

1 Introduction

Memetic algorithms (MA) have been found to be effective for evolutionary computation. It can be viewed as the hybrid genetic algorithms (GA) consisting of local refinement to genetic search results. Because the algorithms involve both the global and local searches, one challenging issue of the MAs is to reduce the computational complexity. One way for lowering the computational time is to employ the parallel MA. There are multiple populations in the parallel MA. Each population evolves independently. With smaller population size, the parallel MA may be able to converge at faster rate while finding good solutions. However, long computational time may still be required when computational intensive problems are considered for optimization.

The objective of this paper is to present a VLSI architecture for parallel MA. The architecture is able to accelerate the MA for both global and local searches. The application considered in this paper is vector quantization (VQ) [1]. The VQ codeword training is computational intensive. It requires long training time even for small number of codewords. Therefore, the VQ design is a good example for verifying the effectiveness of the proposed MA architecture.

* Corresponding author.

In the proposed architecture, each population of the parallel MA is associated with a hardware module for independent memetic evolutions. Each hardware module consists of population memory unit, mutation and crossover unit, K-means [1] unit, and survival test and update unit. The mutation and crossover unit is used for global search, while the K-means unit is used for local refinement. Note that each hardware module contains only one population memory for reducing the area cost [5]. Both the mutation and crossover operations are performed concurrently for accelerating the MA. In addition, a pipeline architecture with direct memory access (DMA) operation is adopted for the K-means operation. A hardware sorting structure [3] is adopted for survival test.

The K-means unit of different modules will request for the delivery of training vectors for local refinement of MA. When the delivery for each K-means unit is performed independently, the resulting memory bandwidth will be high, limiting the speed of the proposed architecture. To solve this problem, the K-means unit of all the modules operate synchronously. The training vectors delivered by he DMA operation will be broadcasted to all the modules. The reduction in bandwidth for transmission of training vectors is beneficial for further accelerating the speed of the proposed architecture for parallel MA.

The proposed architecture has been implemented on field programmable gate array (FPGA) devices [2] so that it can operate in conjunction with a softcore CPU [6]. Using the reconfigurable hardware, we are then able to construct a system on programmable chip (SOPC) system for the genetic VQ design. As compared with its hardware and software counterparts, numerical results reveal that the proposed FPGA-based MA architecture attains higher performance with significantly lower training time for VQ design.

2 The Proposed MA Architecture

Before presenting the proposed architecture, a brief description of the VQ is presented. The goal of a VQ is to partition a large data set $\mathcal{X} = \{x_1, ..., x_t\}$ into N non-overlapping clusters $\mathcal{C}_1, ..., \mathcal{C}_N$, where $N \ll t$. The partitioning process is based on a set of codewords $\{y_1, ..., y_N\}$, where the codewords and the vectors in $\mathcal{X}$ are of the same dimension w. Given a vector $x \in \mathcal{X}$, the x will be assigned to the cluster $\mathcal{C}_i$ when $i = \alpha(x) = \arg\min_{1 \leq j \leq N} d(x, y_j)$,where $d(u, v)$ denotes a distance measure between two vectors u and v. In this paper, the squared distance is adopted as the distance measure. When applied for data reduction applications such as data compression, a vector x will be represented by the codeword y_i when $i = \alpha(x)$. One cost function for the data reduction is the average distortion for representing x by y_i, as shown below

$$D = \frac{1}{wt} \sum_{i=1}^{t} d(x_i, y_{\alpha(x_i)}). \tag{1}$$

Given a data set $\mathcal{X}$, the objective of the VQ design is to find a set of codewords $\{y_1, ..., y_N\}$ minimizing D in eq.(1).

In this paper, a parallel MA is proposed for the VQ design. There are M islands in the parallel MA. Each island evolves independently. In each island, the steady-state GA is used for global search, and the K-means algorithm is adopted for local refinement. As compared with the usual generational GA, the steady-state GA has been found to be more suitable for hardware implementation because of consuming less hardware resource. In each island, there are P genetic strings for the memetic operations. Each string r represents a set of N codewords $\{y_1, ..., y_N\}_r$.

Let S_i be the set of P genetic strings in the island i, which are called the parent strings. Initially, the P strings in S_i are randomly generated. Two strings (denoted by r_1 and r_2) in S will be selected for mutation and crossover for creating a new child string (denoted by c). The new string then is used as the initial codebook to the K-means algorithm for local refinement. As the MA of all the islands are terminated, the fitness of best strings from every island are compared. The string with highest fitness is selected as the codebook of the VQ.

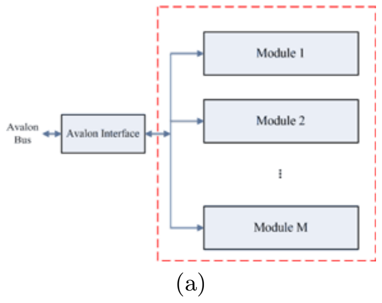
Note that because each string for the VQ design is actually a codebook, the memory access time for string retrieval may be long. Consequently, the retrieval process for r_1 and r_2 may be time-consuming. To reduce the memory access time, in the algorithm, the previous r_1 becomes the new r_2 and then the new r_1 is chosen randomly from S. This selection scheme reduces the memory access time by half.

The hardware architecture of the parallel MA can be viewed as a user logic in the NIOS-based SOPC system [6]. Because the proposed architecture is used for the VQ design, the training data is required. The goal of using the SOPC is to provide the training data for the hardware architecture. The training data can be stored in a SDRAM, and delivered to our architecture via the Avalon bus. The DMA can be used to speed up the delivery of training vectors.

Figure 1 shows the hardware architecture of the parallel MA. It contains M modules, where each module is used for the memetic operations of one island. Each module consists of population memory, crossover & mutation unit, K-means unit, and survival test & update unit. Each unit will be described in detailed as shown below.

Population Memory Unit. The population memory contains a 2-port RAM and a RNG unit. The 2-port RAM contains S, the set of P genetic strings. In our design, the implementation of the RAM is based on the embedded memory, which is provided by some FPGA devices such as Altera Stratix II. The goal of RNG unit in the population memory unit is to select randomly a string r_1 for the subsequent crossover and mutation operations. In our design, the cellular automata (CA) is adopted for the VLSI implementation of random number generator due to its simplicity and regularity of the design.

Mutation and Crossover Unit. Figure 2 shows the basic structure of the mutation and crossover unit, which contains three shift registers for storing the strings r_1, r_2 and c, respectively. A number of RNGs, comparators, multiplexers and counters are then used for crossover and mutation. The major advantage of



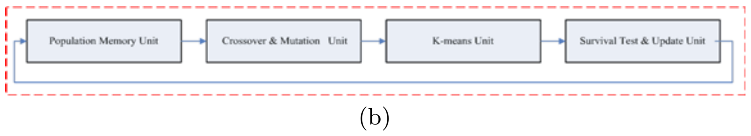


Fig. 1. The architecture of the parallel MA: (a)The overall architecture, (b)The architecture of each module

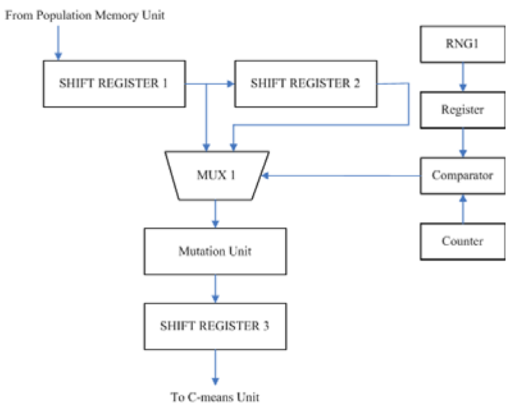


Fig. 2. The architecture of crossover and mutation unit

this architecture is that the crossover and mutation can be performed concurrently with low area cost.

As shown in Figure 2, SHIFT REGISTER 1 and SHIFT REGISTER 2 contain strings r_1 and r_2, respectively. Note that the architecture does not randomly select new r_1 and r_2 from the population memory. In fact, only new r_1 is chosen from population memory. The new r_2 is actually the previous r_1. The memory access time and routing overhead can then be significantly reduced. In the architecture, The SHIFT REGISTER 1 obtains r_1 from the population memory unit. The SHIFT REGISTER 2 obtains r_2 from SHIFT REGISTER 1.

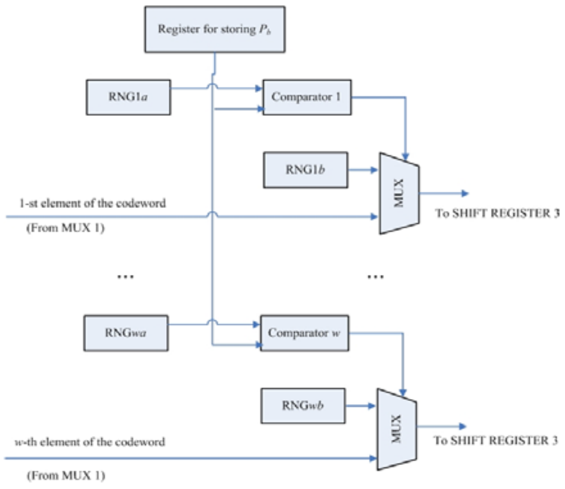


Fig. 3. The architecture of mutation unit

The crossover operations are accomplished by concurrently shifting the strings in SHIFT REGISTER 1 and SHIFT REGISTER 2 to MUX 1. Each shift register will shift one codeword at a time. As shown in Figure 3, MUX 1 is a switch selecting the codewords of either r_1 or r_2, and route them to SHIFT REGISTER 3, which contains the resulting child string c. The control line of MUX 1 is connected to a comparator, which compares the value of RNG 1 to that of a counter. The counter records the number of shifts made by the shift registers. The value of RNG 1 serves as a threshold here. When the counter value is less than the threshold, codewords of SHIFT REGISTER 1 (i.e., r_1) goes to SHIFT REGISTER 3. Otherwise, codewords of r_2 will be selected. Consequently, the value of RNG 1 determines the crossover point. The value will be randomly generated prior to the shifting operations.

We also observe from Figure 2 that the output codeword of MUX 1 will pass through the mutation unit before arriving the SHIFT REGISTER 3. Figure 3 shows the architecture of the mutation unit. As shown in the figure, all w components of the output codeword mutate concurrently. The mutation circuit for each component i consists of 2 RNGs (termed RNG ia and RNG ib), one register (termed register i), one comparator (termed comparator i), one multiplexer (termed mux i).

The probability for mutation P_b is stored in a separate register, and is broadcasted to all the mutation circuits. In the mutation circuit for each component i, the value of RNG ia is first compared with the P_b. The component i will be mutated when the value of RNG ia is less than P_b. The mutated value is then determined by RNG ib.

K-Means Unit. The goal of the K-means unit is to locally refine the mutated child string stored in SHIFT REGISTER 3 using the K-means algorithm.

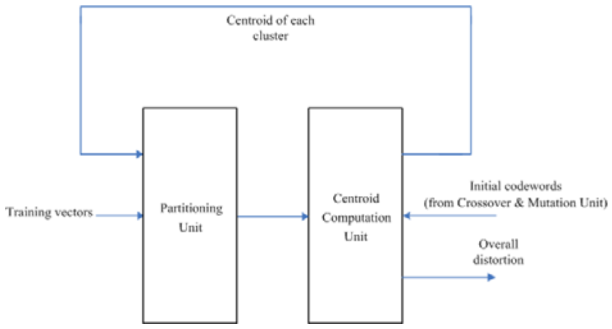

Centroid of each cluster
Training vectors
Partitioning Unit
Centroid Computation Unit
Initial codewords (from Crossover & Mutation Unit)
Overall distortion

Fig. 4. The architecture of the C-means unit

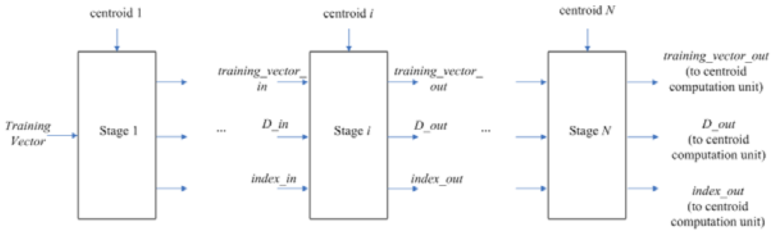


Fig. 5. The architecture of the partitioning unit

As shown in Figure 4, the K-means architecture can be decomposed into two units: the partitioning unit and the centroid computation unit. These two units will operate concurrently for the local refinement process. The partitioning unit uses the codewords stored in the register to partition the training vectors into N clusters. The centroid computation unit concurrently updates the centroid of clusters. Note that, both the partitioning process and centroid computation process should operate iteratively in software. However, by adopting a novel pipeline architecture, our hardware design allows these two processes operate in parallel for reducing the computational time. In fact, our design allows the concurrent computation of N+2 training vectors for the K-means operations.

Figure 5 shows the architecture of the partitioning unit, which is a N-stage pipeline, where N is the number of codewords (i.e., clusters). The pipeline fetch one training vector per clock from the input port. The i-th stage of the pipeline compute the squared distance between the training vector at that stage and the i-th codeword of the codebook. The squared distance is then compared with the current minimum distance up to the $(i-1)$-th stage (denoted by D_{in} in Figure 5). If distance is smaller than the current minimum distance, then the i-th codeword becomes the new current optimal codeword, and the corresponding distance becomes the new current minimum distance, denoted by D_{out}. After the computation at the N-th stage is completed, the current optimal codeword and current minimum distance are the actual optimal codeword and the actual minimum distance, respectively. The index of the actual optimal codeword and

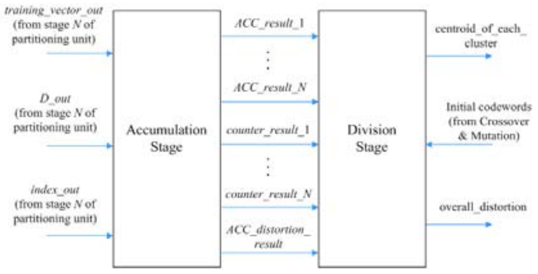

training_vector_out (from stage N of partitioning unit)
D_out (from stage N of partitioning unit)
index_out (from stage N of partitioning unit)
Accumulation Stage
ACC_result_1
ACC_result_N
counter_result_1
counter_result_N
ACC_distortion_result
Division Stage
centroid_of_each_cluster
Initial codewords (from Crossover & Mutation)
overall_distortion

Fig. 6. The architecture of the centroid computation unit

its distance will be delivered to the centroid computation unit for computing the centroid and overall distortion.

Figure 6 depicts the architecture of the centroid computation unit, which can be viewed as a two-stage pipeline. In this paper, we call these two stages, the accumulation stage and division stage, respectively. Therefore, there are $N + 2$ pipeline stages in the K-means unit. The concurrent computation of $N+2$ training vectors therefore is allowed for the local refinement operations.

There are N accumulators and N counters in the accumulation stage. The i-th accumulator records the *current* sum of the training vectors assigned to cluster i. The i-th counter contains the current number of training vectors mapped to cluster i. The *training_vector_out, D_out* and *index_out* in Figure 6 are actually the outputs of the N-th pipeline stage of the partitioning unit. The *index_out* is used as control line for assigning the training vector (i.e. *training_vector_out*) to the optimal cluster found by the partitioning unit.

The goal of division stage is to compute the centroid. There is only one divider in the unit because one centroid computation is necessary at a time. Assume the j-th training vector is now entering the stage, and the i-th codeword is the best matching codeword for the j-th training vector. The output of the i-th accumulator and i-th counter of the accumulation stage will then be delivered to the divider for centroid computation. The divider is implemented by a table lookup based technique so that it can be embedded in a pipeline. Detailed discussion of the K-Means architecture can be found in our earlier work [4].

Survival Test and Update Unit. This unit contains a hardware sorting circuit [3], which sorts the N parent strings in a descending order according to their fitness values. After the fitness evaluation operation is completed, the fitness value of the child string is used as the input to the sorting circuit. When the distortion of the string is larger than the parent string with lowest fitness value, the child string is not survival, and no updating operation is necessary. Otherwise, the parent string with highest distortion is replaced by the child string. The sorting circuit is then activated to determine the new parent string with the highest distortion.

Table 1. Area cost of the SOPC system for parallel MA

Items	Values
ALMs of parallel MA Circuit ($M = 3$)	19281
Embedded Memory Bits of parallel MA Circuit	12288
DSP Blocks of parallel MA Circuit	288
ALMs of Entire SOPC System	22895
Embedded Memory Bits of Entire SOPC System	609024
DSP Blocks of Entire SOPC System	288

Synchronous Memetic Operations Among Islands. Although the genetic strings in different islands evolve independently, they all need the same set of training vectors for K-means algorithm and fitness evaluation. Independent requests for training vector delivery from different islands demand very high memory bandwidth. This may become the bottleneck of the architecture. To solve the problem, the K-means operation of all the architectures operate synchronously. Therefore, training vectors from main memory can be broadcasted to all the islands for K-means training. In addition, the DMA is used for further accelerating the data delivery. To implement synchronous K-means operations among different modules, we first note that all the string selection, mutation & crossover, and survival test operations take fixed number of clock cycles. The same operation will take the same number of clocks in different modules. Consequently, when all the modules start MA operations at the same time, the synchronization among the modules can be achieved.

3 Experimental Results

This section presents some physical performance measurements of the proposed FPGA implementation. In the experiments, there are 16 codewords (i.e., $N = 16$) in the VQ. The vector dimension of codewords is $w = 2 \times 2$. The target FPGA device for the hardware design is Altera Stratix II 2S60ES. Table 1 shows the hardware resource consumption of the NIOS-based SOPC system [6] using the proposed MA architecture as the custom user logic. There are 3 islands in the architectures ($M = 3$). The number of genetic strings in each island is $P = 8$. Therefore, the total number of genetic strings of the architecture is 24 strings.

As shown in the Table 1, the proposed MA circuit consumes only 19281 adaptive logic modules (ALMs). The population memory of the architecture is implemented by the embedded memory of the FPGA. The consumption of the population memory bits for the parallel MA circuit is 12288 bits. Moreover, our circuit also uses 288 digital signal processing (DSP) blocks of the FPGA device for the implementation of squared distance computation in the K-means unit. The NIOS softcore CPU [6] itself also consumes hardware resources. The whole SOPC system uses 22895 ALMs, which is 95 % of the ALMs of the target FPGA device. The operating speed of the system is 50 MHz.

Table 2. Comparisons of the performance of various MA implementations

Algorithms	ALMs	Embedded Memory Bits	DSP Blocks	Average CPU Time	Average Distortion
Parallel MA ($M = 3$)	19281	12288	288	0.47 (sec)	54.62
Parallel MA ($M = 1$)	7203	12288	96	1.07 (sec)	55.27
Software				34.55(sec)	55.23

Table 2 compares the performance of the proposed architecture with $M = 1$ and $M = 3$, and their software counterparts with $M = 1$. The CPU time and average distortion measurements are based on the average values of 100 independent runs. The area cost of the hardware implementations are also included in the table for comparison purpose. The proposed architecture with $M = 1$ can be viewed as the implementation of basic memetic algorithm. To achieve meaningful comparisons, both the architectures have the same number of total genetic strings $P = 24$. They are also implemented on the same target FPGA device. The software counterpart executes on the processor 4GHz Intel I7.

It can be observed from Table 2 that the proposed architecture with $M = 3$ has lowest average CPU time and lowest average distortion. The average CPU time of the proposed architecture with $M = 3$ is 0.47 second, which is only 43.93% and 1.36% of the CPU time of the architecture with $M = 1$ and the software counterpart with $M = 3$, respectively. Note that, each island for $M = 3$ has only 8 genetic strings. By contrast, the island for $M = 1$ has 24 genetic strings. Therefore, because each island for $M = 3$ has smaller population, its memetic operations are able to achieve faster convergence. In addition, the best string is selected from multiple islands. Its average distortion is lower than that of the basic memetic algorithm containing only one island, as shown in Table 2.

Although each island has smaller population for $M = 3$, the total number of genetic strings of the proposed architecture with $M = 3$ is identical to that of the architecture with $M = 1$. As a result, we can see from Table 2 that both the hardware architectures consume the same number of embedded memory bits. On the other hand, the architecture with $M = 3$ uses more ALMs and DSP blocks for hardware implementation. This is because the architecture with $M = 3$ consists of 3 islands. Therefore, when both the area cost and speed are the important concerns, the proposed architecture with $M = 1$ can be used.

Figure 7 shows the distribution of distortion for 100 independent executions of the proposed architectures with $M = 1$ and $M = 3$, and the K-means architecture [4]. It can be observed from the figure that the proposed architectures with $M = 1$ and $M = 3$ have similar distributions. In addition, their distributions have higher concentration as compared with that of the K-Means architecture [4]. Therefore, the performance of the proposed architecture is insensitive to the initial populations. All these facts demonstrate the effectiveness of the proposed architecture.

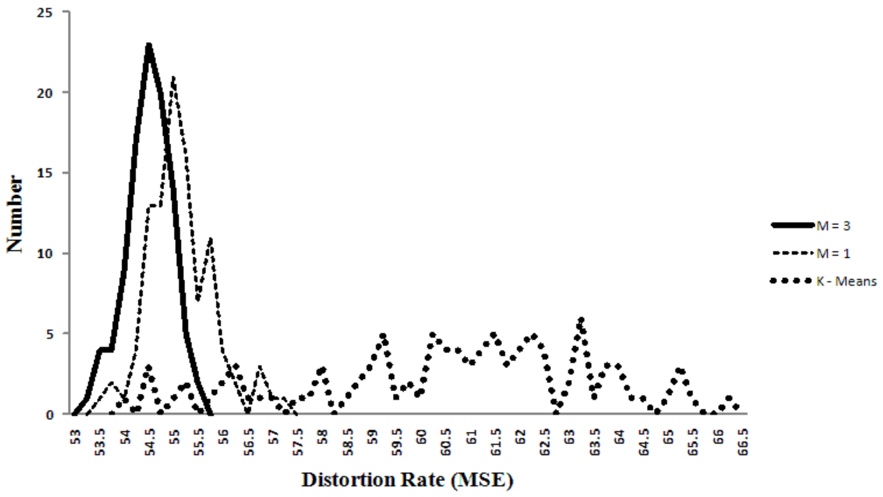


Fig. 7. The distortion distribution of various VQ implementations

4 Concluding Remarks

The proposed parallel MA architecture has been found to be effective for VQ design. As compared with the K-means architecture, the proposed MA architectures are less sensitive to the selection of initial codewords. In addition, by the employment of parallel MA architectures, the average distortion of the VQ design can be further improved while accelerating the computational speed. The proposed architecture therefore is an effective alternative for applications where both the performance of speed are important concerns.

References

1. Gersho, A., Gray, R.M.: Vector Quantization and Signal Compression. Kluwer, Norwood (1992)
2. Hauck, S., Dehon, A.: Reconfigurable Computing. Morgan Kaufmann, San Francisco (2008)
3. Hwang, W.J., Li, H.Y., Yeh, Y.J., Chan, K.F.: FPGA Implementation of Competitive Learning with Partial Distance Search in the Wavelet Domain. In: Kang, G.B. (ed.) Progress in Neurocomputing Research, ch. 8, pp. 203–221. NOVA Science Publisher (2008)
4. Li, H.Y., Hwang, W.J., Hsu, C.C., Hong, C.L.: Efficient K-Means VLSI Architecture for Vector Quantization. In: Salberg, A.-B., Hardeberg, J.Y., Jenssen, R. (eds.) SCIA 2009. LNCS, vol. 5575, pp. 440–449. Springer, Heidelberg (2009)
5. Nedjah, N., Mourelle, L.: Hardware Architecture for Genetic Algorithms. In: Ali, M., Esposito, F. (eds.) IEA/AIE 2005. LNCS (LNAI), vol. 3533, pp. 554–556. Springer, Heidelberg (2005)
6. NIOS II Processor Reference Handbook, Altera Corporation (2008), `http://www.altera.com/literature/lit-nio2.jsp`

Optimization of Minimum Completion Time MTSP Based on the Improved DE

Huiren Zhou[1] and Yinghui Wei[2]

[1] Institute of Systems Engineering, Tianjin University, Tianjin 300072, China
[2] Department of Management, Liaoning Institute of Science and Technology, Benxi 117022, China
{huirenzhou,mary_wei2003}@126.com

Abstract. In order to solve traveling salesman problems that employed completion- time- shortest as the evaluating rule, an encoding method and improved differential evolution algorithm were proposed. In these methods, real number encoding and roulette wheel selection were adopted for improved differential evolution and neighborhood search operator was devised. It was fit for solving symmetric and asymmetric multiple traveling salesman problem. Asymmetric multiple traveling salesman problems were simulated. By comparison with the results of genetic algorithm and standard differential evolution, it is shown that the improved differential evolution algorithm proposed in this paper is efficient to solve the discrete combinatorial problem, such as optimization of multiple traveling salesman problems.

Keywords: differential evolution algorithm, multiple traveling salesman problem, discrete combinatorial problem, optimization.

1 Introduction

As a typical NP-hard combinatorial optimization, TSP (Traveling Salesman Problem) is widely used in various application areas [1]. Usually, MTSP (Multiple Traveling Salesman Problems) involves *M* salesmen departing from the same city (or different cities) by their specific routes. For each of the cities, exactly one salesman will pay a visit (with an exception of departure cities) with minimum total travel distance. The TSP-related research has a great significance in the real world in that many applications such as traffic, pipe laying, route selection, computer network topology, and traveling postman can be formulated as TSP/MTSP models. A number of Bionic Evolution Algorithms including GA [2], AC [3], SA [4] and TS [5] have been devised to solve the TSP and MTSP models.

The study of the total completion time of all salesmen is of operational significance because the speed of each traveling salesman may vary in practical problems. Reference [6] proposed a genetic algorithm to minimize the total traveling time of the MTSP model. However, little research has been done in this area.

DEA (Differential Evolution Algorithm), a real-coded evolutionary computing technique, was proposed by Storn and Price in 1995 [7]. Initially, it was devised to compute Chebyshev polynomial. Afterwards, it proved to be an effective method in optimizing complex problems. As a new intelligent optimization based on evolved population, DEA has the merits of desirable effectiveness, convergence, and robustness. Although DEA has drawn a wider attention of research, little intensive and systematic research findings have been published compared with those of other evolutionary algorithms. Theoretical breakthrough of DEA is expected [8]. In particular, little efforts have been made in using DEA to solve discrete combinatorial optimizations. In this paper, the completion time of MTSP is optimized with an improved DEA.

To optimize the symmetric or asymmetric MTSP with minimum completion time, an improved DEA together with a new coding method is proposed in this paper. The optimization consists of defining the cities visited by and the route of a specific salesman, namely, defining the allocation of salesmen and the routes. At the completion of travel, minimize the maximum traveling time consumed by certain salesman. The simulation indicates that the proposed Improved DEA is valid.

2 Mathematical Model of MTSP

Define point 0 (origin point) as the departure city of a salesman; points 1, …, l denotes the cities to be visited by m traveling salesmen.

Define variables:

$$x_{ijk} = \begin{cases} 1 & salesman \quad k \quad passes \quad arc(\mathrm{i,j}) \\ 0 & otherwise \end{cases}$$

$$y_{ki} = \begin{cases} 1 & \text{salesman k visits city i} \\ 0 & otherwise \end{cases}$$

c_{ij} —the distance of arc (i, j) passed by salesman

v_k —the speed of the k th salesman

h_k —the completion time of the k th salesman

The following model is formulated:

Objective function is

$$H = \min(\max(h_1, h_2, \cdots, h_m)) \tag{1}$$

Of which,

$$h_k = \sum_{i=0}^{l}\sum_{j=0}^{l} c_{ij} x_{ijk} / v_k \quad (k = 1,2,\cdots m) \tag{2}$$

Subject to

$$\sum_{k=1}^{m} y_{ki} = \begin{cases} m & i = 0 \\ 1 & i = 1,2,\cdots,l \end{cases} \tag{3}$$

$$\sum_{i=0}^{l} x_{ijk} = y_{kj} \tag{4}$$

Of which, $j = 0,1,\cdots,l;\ k = 1,2,\cdots,m$

$$\sum_{j=0}^{l} x_{ijk} = y_{ki} \tag{5}$$

Of which, $i = 0,1,\cdots,l;\ k = 1,2,\cdots,m$

$$X = (x_{ijk}) \in S \tag{6}$$

Of which, *S* is branch eliminating constraint. Namely, discard the solution of branch tracks unable to form a complete route. See [9] for details of *S*.

In the model, Equation (1) minimizes the completion time of the one among *m* salesmen consuming the maximum traveling time. Equation (2) computes the traveling time of each salesman. Equation (3) means that departing from city 0, each of the cities will have exactly one salesman passed. Exactly one city is connected with the terminal city of any arc, as described in equation (4). For the departure city of any arc, exactly one terminal city shall be connected with it (see Equation (5)). Finally, Equation (6) indicates that any solutions of branch tracks unable to form a complete route shall be rejected.

3 Design of Improved DEA

DEA, short for differential evolution algorithm, was proposed by Storn and Price in 1995 to solve continuous global optimization problems [10]. Similar to PSO (Particle Swarm Optimization) and GA (Genetic Algorithm), DEA is an optimization method utilizing swarm intelligence theory, in which swarm intelligence derived from the cooperation and competition between individuals within the swarm is used to guide the searching process. To improve the standard DEA, we propose new selection operation and neighborhood operation methods according to features of the combinatorial optimization problem.

3.1 Individual Encoding

In respect that the DEA is fit for solving continuous function optimization, rational individual encoding method shall be devised for the discrete function optimization of TSP. Each individual generated shall correspond to a feasible route and no infeasible solutions may produce during the mutation and crossover processes.

Consider a MTSP involving *N* + 1 cities (including the departure city 0) and *m* traveling salesmen. The individual generated is represented by $[x_1, x_2, x_3, \cdots, x_N]$.

The individual consists of *N* real numbers, and each real number x_i respects the following condition: $1 \le x_i < m+1,\ i = 1,2,\cdots N$.

3.2 Individual Decoding

For decoding, rounding operation for each real number in the code is conducted. Suppose x_i represents that the traveling salesman $Int(x_i)$ visits city *i*, where $Int(x_i)$ is the rounding operation for real number x . It is possible that for $j \neq k$, $Int(x_j) = Int(x_k)$, which means a salesman visits more than one cities. In this case, sequence real numbers x_j and x_k in an ascending order so as to determine the sequence of cities to be visited by the salesman. In specific, the salesman visits the city with the smallest number first. In case real numbers $x_j = x_k$, the city that to be visited first is determined by the appearance sequence of the number x_j , x_k , and the first visited city appears first in the individual.

For instance, consider the MTSP containing 10 cities (including the departure city 0) and 3 traveling salesmen. One possible individual could be described as [1.2 2.3 3.9 2.1 1.8 3.3 1.2 2.4 3.7]. From the individual we see, the sequences of cities to be visited shall be Salesman 1: 0—1—7—5—0; Salesman 2: 0—4—2—8—0;Salesman 3: 0—6—9—3—0.

To eliminate trivial sub-routes, it is important to assume C_{00}=*M*, where *M* is an infinite positive number and C_{00} is the distance between departure cities. Consider the following individual:[1.2 1.3 3.9 1.1 1.8 3.3 1.2 3.4 3.7]. From the above we see, the sequences of cities to be visited shall be Salesman 1: 0—4—1—7—2—5—0; Salesman 2: 0—0—0; Salesman 3: 0—6—8—9—3—0.

Taking the assumption C_{00}=*M* into consideration, the traveling distance of salesman 2 is infinite. Accordingly, the traveling time is infinite. No trivial sub-routes may produce because such an individual will be rejected in the iteration process of DEA.

3.3 Mutation Operation

The basic mutation element of DEA is the differential vector of the parent generation. Each vector consists of two different individuals (x_{r1}^t, x_{r2}^t) of parent. Depending on the generation methods of mutated individuals, diversified DEA schemes will be obtained. Among them, one possible individual mutation is given as follows:

$$x_{mi} = x_i^t + \lambda(x_{best}^t - x_i^t) + \beta(x_{r2}^t - x_{r1}^t) \tag{7}$$

In which, $i = 1, 2, \cdots, N$; x_{best}^t is the individual with the optimal fitness value of the current population; x_{r1}^t and x_{r2}^t are random individuals different from x_i^t ; λ is an additional control variable used to improve the expectation by introducing the optimal individual; and β is a scaling factor.

In the paper, parameters λ and β are assigned the same value: and set $\lambda = \beta$ =0.5.

3.4 Boundary Conditions Dealing

The value of new elements generated from mutation may exceed the boundary. Therefore, it is necessary to replace the element values failing to meet the boundary conditions with values randomly-generated from the feasible domain.

3.5 Crossover Operation

DEA uses crossover operation to maintain the diversity of population. Crossover will conduct between the i^{th} individual x_i^t and x_{mi} to generate a trial individual x_{Ti}. To guarantee the evolution of x_i^t, random selection enables that at least one bit of x_{Ti} comes from x_{mi}. For other bits of x_{Ti}, a crossover factor determines the bits coming from x_{mi} and x_i^t, respectively. The crossover equation is described as follows:

$$x_{Tij} = \begin{cases} x_{mij} & rand_j \le C_r \\ x_{ij}^t & rand_j > C_r \end{cases} \quad j = 1,2,\cdots,D. \tag{8}$$

In which, $i = 1,2,\cdots,N$; rand is a random number at the interval of [0,1]; and $C_r \in$ [0,1].

3.6 Selection Operation

Using "greedy search" strategy, the standard DEA generates trial individual x_T through mutation and crossover. By competing with x_i^t, x_T will be selected as the next generation only if the fitness value of it outperforms that of x_i^t. Otherwise, x_i^t is selected as the next generation. The selection equation is described as follows:

$$x_i^{t+1} = \begin{cases} x_{Ti} & f(x_{Ti}) \le f(x_i^t) \\ x_i^t & f(x_{Ti}) > f(x_i^t) \end{cases} \tag{9}$$

To avoid possible stagnation during the standard DEA evolution process, the proportional selection method on the roulette wheel basis of GA is borrowed [11]. Select *N* individuals from parent and trial populations containing *N* individuals respectively to generate the next generation population. According to the roulette wheel selection in GA, the individual with the largest fitness value remains competitive. Therefore, the reciprocal of the fitness value obtained in Equation (1) will be used as the fitness value of roulette wheel selection. To ensure that the individual with the largest fitness value remains in next generation population, the optimal reservation strategy in GA will be used [11].

3.7 Neighborhood Operation Method

Neighborhood operation involves randomly selecting two bits of an individual, exchanging the values of the two bits. Namely, exchange randomly two nodes in feasible route. Considering the following parent individuals V: [2.2 1.4 3.9 0.6 5.7].

By exchanging 1.4 and 5.7, the following individual of the next generation will be obtained V': [2.2 5.7 3.9 0.6 1.4].

3.8 Procedures of Improved DEA

The procedures of Improved DEA are as follows:

Step 1: As per the optimization problem to define the range of values; and as per the population scale to generate the initial population X with a pop-size N.

Step 2: Perform mutation operation as per Equation (7) and apply the boundary conditions.

Step 3: Generate new population X^T by crossover as per Equation (8).

Step 4: For populations X and X^T, apply roulette wheel and optimal conservation strategy to generate population X^S containing N individuals.

Step 5: Perform neighborhood operation for population X^S to generate X^n.

Step 6: The individuals in population X^S compete with those in X^n. Namely, select individuals as per Equation (9) to generate new generation of population X^t.

Step 7: Repeat Step 2 to Step 7 until the specified termination condition is satisfied. In the paper, the termination condition refers to the maximum number of iterations.

4 Instance Simulations

The above introduced Procedures of using Improved DEA to optimize the minimum completion time multiple traveling salesman problems. In this section, simulations are conducted using the data of asymmetric TSPLIB as a benchmark (http://www.iwr.uni-heidelberg.de/groups/comopt/software/ TSPLIB95/). The simulation results will then be compared with those of GA in [6], and standard DEA.

4.1 MTSP Problems in 17 Cities (br17.atsp)

The simulation benchmark br17.atsp involves 17 cities and the distance between each two cities are asymmetric. The distances in the matrix is in Km. Suppose the number of salesmen is 3 with the traveling speed 4 km/h, 5 km/h and 6 km/h, respectively.

Reference [6] performed CX crossover, OX crossover and PMX crossover at a rate of 0.55 for chromosomes. Adopt exchange mutation and the exchange mutation rate is 0.25; the pop-size is 100; maximum iterations are 100. Program with Matlab and run randomly 10 times. The results are given in Table 1.

In the paper, let $F = \lambda = 0.5$ and $C_r = 0.05$; pop-size is 100; number of iterations is 50. For the Improved DEA, program it with Matlab. Run randomly 10 times and the results are given in Table 1.

And the simulation with Improved DEA is given in Figures 1.

Table 1. Simulation results of br17.atsp

	Optimal	Worst	Average
OX	4.6667	7	5.3833
CX	4.6667	7.75	5.9400
PMX	4.6667	7.75	5.9117
Improved DEA	4.6667	5.6	4.86

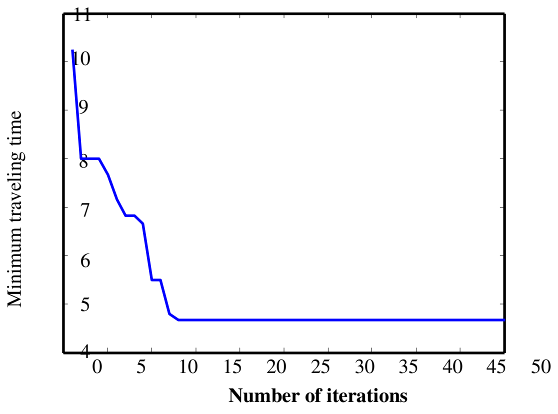


Fig. 1. Simulation of br17.atsp with Improved DEA

The instance is a small-scale MTSP. As can be seen from Table 1, the results with Improved DEA devised in the paper outperform those obtained with GA, and less iteration is needed.

4.2 MTSP Problems in 100 Cities (kro124p)

The simulation benchmark br124p involves 100 cities and the distance between each two cities are asymmetric. The distances in the matrix is in Km. Suppose the number of salesmen is 4 and 8. The traveling speed is 24, 35, 52, and 28, respectively. For MTSP with 8 salesmen, the speeds are 24, 35, 52, 28, 32, 41, 39 and 50.

Reference [6] performed CX crossover, OX crossover and PMX crossover at a rate of 0.55 for chromosomes. Adopt exchange mutation and the exchange mutation rate is 0.25; the pop-size is 100; the number of maximum iteration is 800. Program with Matlab and run randomly 10 times for the number of salesmen of 4 and 8, respectively. The results are given in Table 2.

Table 2. Simulation results of GA

No. of salesmen	Crossover operator	Optimal	Worst	Average
	CX	641.3714	732.2143	667.1155
4	OX	611.6000	700.5714	656.3961
	PMX	584.4167	644.500	617.1515
	CX	372.4000	410.2439	390.3927
8	OX	369.6410	416.2571	393.3918
	PMX	342.2800	389.9268	372.0015

Table 3. Simulation results of Improved DEA

No. of salesmen	Optimal	Worst	Average
4	553.1714	572.2143	563.4521
8	292.5641	328.5938	314.2036

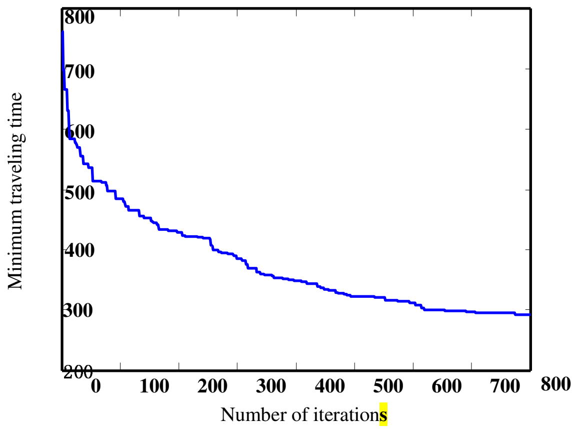


Fig. 2. Simulation of kro124p with Improved DEA

In the paper, let $F = \lambda = 0.5$ and $C_r = 0.05$; pop-size is 100; number of iterations is 800. Program the Improved DEA with Matlab. Run randomly 10 times for the number of salesmen 4 and 8, respectively. The results are given in Table 3.

In which, the simulation figures of one of optimal results using Improved DEA by 8 travelling salesmen are shown in Figures 2.

The instance is a medium-scale MTSP. As can be seen from Tables 2 and 3, the results with Improved DEA devised in the paper outperforms those obtained with GA in the aspects of optimal, worst, and average.

4.3 MTSP Problems in 403 Cities (rbg403.atsp)

The benchmark rbg403 involves 403 cities and the distance between each two cities is asymmetric. Suppose the distance unit in the matrix is km. Suppose the number of salesmen is 8 and 12. For 8-salesman, the traveling speeds are 24, 35, 52, 28, 32, 41, 39 and 50km/h, respectively. For 12- salesman, the speeds are 20, 35, 42, 18, 32, 11, 39, 50, 62, 28, 12 and 8km/h.

Simulate it with standard DEA and Improved DEA 10 times, respectively. Here, let $F = \lambda = 0.5$, $C_r = 0.1$; pop-size =100; maximum iterations = 1000. For 8-salesman simulation, the results at the iteration number of 500, 800, and 1000 are given in Table 4. For 12-salesman simulation, the results are given in Table 5.

The simulation of 8 salesmen at the iteration number of 1000 with standard DEA and Improved DEA is given in Figures 3 and 4, respectively.

Table 4. Simulation results of rbg403 with 8 salesmen

Iterations	Standard DEA			Improved DEA		
	Optimal	Worst	Average	Optimal	Worst	Average
500	35.9091	36	35.764	21.8974	23.5	22.709
800	34.2619	35.0625	34.715	20.5	23.5	21.8
1000	33.7692	34.2619	33.99	19.9444	22.2	20.964

Table 5. Simulation results of rbg403 with 12 salesmen

Iterations	Standard DEA			Improved DEA		
	Optimal	Worst	Average	Optimal	Worst	Average
500	26.6563	28.4545	27.6184	16.5	17	16.6964
800	25.875	27.25	26.4757	15.5	16.5	15.8742
1000	25.2727	26.6667	25.7349	15	16.5	15.5289

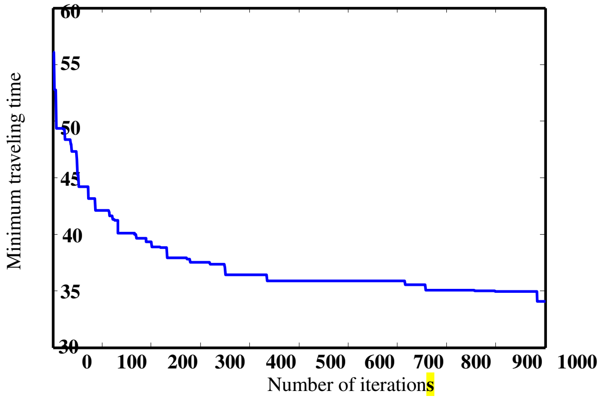


Fig. 3. Simulation with standard DEA

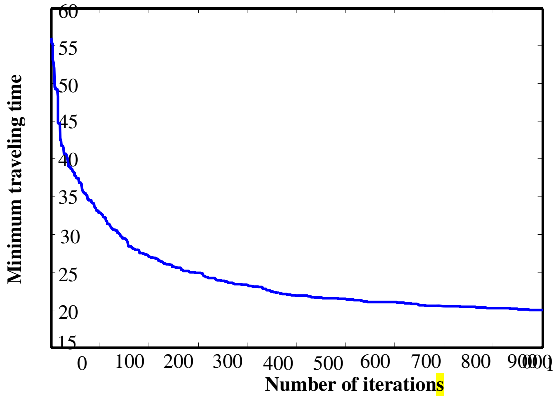


Fig. 4. Simulations with Improved DEA

The instance is a large-scale MTSP. As can be seen from Tables 4 and 5, the results with Improved DEA devised in the paper outperforms those obtained with standard DEA in the aspects of optimal, worst, and average, which shows that the rate of convergence of the Improved DEA is much fast.

5 Conclusions

For discrete combinatorial optimization problems such as MTSP with minimum completion time, a encoding method and improved differential evolution algorithm are proposed in this paper. Through simulation and comparison of examples, the instance simulation indicates that the proposed methods are fit for the optimization of symmetric and asymmetric MTSP and have gained better effects.

Acknowledgements. Sponsored by China Postdoctoral Science Foundation funded project (20090450759).

References

1. Greco F.: Travelling salesman problem. In-Teh., Croatia (2008)
2. Wang, H.-l., Zhou, H.-r., Wei, Y.-h.: Study on multiple traveling salesman problem based on genetic algorithm. Journal of Computer Applications 29(1), 119–122 (2009)
3. Hung, K.S., Su, S.F., Lee, Z.J.: Improving ant colony optimization algorithms for solving traveling salesman problem. Journal of Advanced Computational Intelligence and Intelligent Informatics 11(4), 433–436 (2007)
4. Ohlmann, J.W., Thomas, B.W.: A compressed-annealing heuristic for the traveling salesman problem with time windows. Journal on Computing 19(1), 80–90 (2007)
5. Thamilselvan, R., Balasubramanie, D.P.: A genetic algorithm with a tabu search for traveling salesman problem. International Journal of Recent Trends in Engineering 1(1), 607–610 (2009)
6. Zhou, H.-r., Tang, W.-s., Wei, Y.-h.: Study on minimum completion time multiple traveling salesman problem based on genetic algorithm. Application Research of Computers 26(7), 2526–2529 (2009)
7. Storn, R., Price, K.: Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. International Computer Science Institute, Berkley (1995)
8. Liu, B., Wang, L., Jin, Y.-h.: Advances in differential evolution. Control and Decision 22(7), 721–729 (2007)
9. Li, J., Guo, R.-h.: Dispatch optimize theory and methods for logistic distribution vehicle, pp. 63–73. Chinese Logistics Publishing House, Beijing, (2001)
10. Xu, Z.-b.: Computational intelligence-simulated evolutionary computation, pp. 116–117. Higher Education Press, Beijing (2004)
11. Liu, M., Wu, C.: Intelligent optimization scheduling algorithms for manufacturing process and their applications, pp. 49–55. National Defense Industry Press, Beijing (2008)

Differential Evolution for Optimization of Land Use

Yanjie Zhu[1] and Zhihui Feng[2]

[1] School of Economics and Management, Xuchang University, 461000 XuChang, China
[2] College of Information and Management Science, Henan Agricultural University, 450002 Zhengzhou, China

Abstract. With rapid economic growth of Henan province, it is necessary to optimize all types of land resource allocation. Differential evolution algorithm is an evolutionary algorithm based on groups, which utilizes the differential information of individuals in the current population of solutions to guide its further search. There are three operators in Differential Evolution, mutation, crossover and selection. It applies that improved method to the optimization of land resource allocation. A model of land use allocation is put forward. The model can attain the optimal solution under multi-constraints such as economic benefit, coordinated and balanced program of development, the total population, agricultural acreage and environment. Results show that differential Evolution is effective and has robust character in dealing with multi-constraint and multi-dimensional optimization problems by cooperation and evolution of swarm.

Keywords: Differential Evolution, Constraints, Optimization.

1 Introduction

High-speed economic growth brings enormous pressure of land resources in Henan province. Land for construction is increasing, agricultural land is decreasing. With the economy continuing increase, supply of land resources gradually becomes the important factor that restricts the growth and economic sustainable development of the national economy. The limitation of land resources supply and unlimitedness of the resources demand determine the land resource allocation should be optimized.

Differential Evolution (DE) Algorithm [1] is put forward by Storn R and Price K in 1995; it is a kind random search algorithm with real vector coded. Its principle is simple, and few control parameters and easy coding. Recently, with its usability, robustness and real powerful optimal capacity, successful application in several fields causes wide attention of scholars [2-3].

The optimized allocation of land resource is constrained by several factors, such as the land area, economic benefit, coordinated and balanced program of development, the total population, agricultural acreage, environment and actual situation. The allocation of land resources in Henan province is put forward and optimized.

2 Differential Evolution Algorithm

Differential Evolution Algorithm is based on group with the optimal solution and individual memory within populations of information sharing. Optimization problem solving is achieved by cooperation and competition of populations among individual. Its essence is a greed genetic algorithm with a real number coding. Random initialized population is $X^0 = \left[x_1^0, x_2^0, \cdots, x_{N_P}^0 \right]$, N_P is the population scale. Individual $x_i^0 = \left[x_{i,1}^0, x_{i,2}^0, \cdots, x_{i,D}^0 \right]$ is used to characterize solution for the optimization problem, D is the dimension. Another new population is produced by Mutation and crossover operations of current population. The new generation of population is generated by selecting operation from the two populations based on the greed genetic algorithm.

2.1 Initialization

DE uses the dimension of D real-valued vector parameters as each generation of population, each individual: $x_{i,G}\left(i = 1, 2, \cdots, N_p\right)$, i am sequence of individuals in a population; G is evolutionary algebra, N_P is population scale and remains unchanged at during the minimizing process.

In order to establish initial points of optimization search, population must be initialized. Usually a method for the initial population is random selection from the constraints of the given boundary value [4]. Generally DE assumption of all randomly initialized population with uniform probability distribution. The limit of parameter variable is $x_j^L < x_j < x_j^U$,

$$x_{ij,0} = rand\left[0,1\right] \cdot \left(x_j^U - x_j^L\right) + x_j^L \quad (i = 1, 2, \cdots, N_P; j = 1, 2, \cdots, D) \tag{1}$$

In expressions, *rand*[0,1] say generating an even rand number in [0,1].If the preliminary solution exists in advance, the initial population is produced by the solution with random deviation of the normal distribution. It enhances reconstruction result.

2.2 Mutation

For each objective vector $x_{i,G}\left(i = 1, 2, \cdots, N_P\right)$, the variation vector of DE Algorithm is as follows:

$$v_{i,G+1} = x_{r1,G} + F \cdot (x_{r2,G} - x_{r3,G}) \tag{2}$$

In expressions, $r1, r2, r3$ serial number of random selection are different each other. It is also different between objective vector i and $r1, r2, r3$. Also $N_P \geq 4$ is needful. Mutation operator $F \in [0, 2]$ is a real constant factor and controls the amplification of deviation variable.

2.3 Crossover

In order to increase the diversity of interference parameter vector, crossover operation is introduced. The test vector changes to:

$$u_{i,G+1} = (u_{1i,G+1}, u_{2i,G+1}, \cdots, u_{Di,G+1}) \tag{3}$$

$$u_{ji,G+1} = \begin{cases} v_{ji,G+1} & \text{if } (randb(j) \le CR) \text{ or } j = rnbr(i) \\ x_{ji,G+1} & \text{if } (randb(j) > CR) \text{ and } j \ne rnbr(i) \end{cases} \quad (i = 1,2,\cdots,N_P;\ j = 1,2,\cdots,D) \quad (4)$$

In expressions, *randb(j)* is a uniform distribution random number within [0,1] , *j* is on behalf of the *j*th variable; *rnbr(i)* is a sequence of random selection and used to ensure $u_{i,G+1}$ getting at least one parameter from $v_{i,G+1}$, *CR* is crossover operator and its scope is [0, 1].

2.4 Selection

Whether the vector $u_{i,G+1}$ will become the next generation member, according to the greedy rule DE will compare the vector with object vector of current population. If the objective function will be minimized, the vector with minor value of objective function will accept in the next generation population. All individuals of the next generation population are better or at least as well.

$$x_{i,G+1} = \begin{cases} u_{i,G+1}, & \text{if } f(u_{i,G+1}) \le f(x_{i,G}) \\ x_{i,G}, & \text{otherwise} \end{cases} \quad (5)$$

2.5 Manipulation of Boundary Conditions

It is necessary to ensure new individual parameters value in feasible domain for the problem with boundary constraints. A simple method is to replace the new individual mismatched boundary constraints by the parameter vector randomly generated in the feasible domain.

$$\begin{aligned} & \text{if } u_{ji,G+1} < x_j^L \text{ or } u_{ji,G+1} > x_j^U, \text{ then} \\ & u_{ji,G+1} = rand_j[0,1] \cdot (x_j^U - x_j^L) + x_j^L; \quad (i = 1,2,\cdots,N_P;\ j = 1,2,\cdots,D) \end{aligned} \quad (6)$$

3 Model of Land Use Allocation

3.1 Variable Settings

According to classification of general plan of land utilization in Henan province, the variable settings are respectively cultivated land (x_1), garden (x_2), woodland (x_3), grassland (x_4),water surface(x_5), residential areas and mining areas (x_6),Lands used for transportation(x_7), land for water facilities (x_8) and unused land (x_9). In Henan province, the total land area S = farmland ($x_1 + x_2 + x_3 + x_4 + x_5$) +construction land ($x_6 + x_7 + x_8$) +unused land (x_9).

3.2 Objective Function

From the standpoint of optimized resource utilization, we should pay attention to social returns and ecological environment in pursuit of economic efficiency. Area of utilized land is an important evaluation index of land carrying capacity. Therefore, in the model of land resource optimization Allocation, the minimum area of land used is chosen as major objective, i.e. the maximum area of unused land (x_9), $f(X)$ is used to express it.

$$f(X) = \min \sum_{j=1}^{18} \sum_{i=1}^{8} x_{ij} = \max \sum_{j=1}^{18} x_{9j} \tag{7}$$

In expressions:

X——Area of various types of land in each district ;
i——Various types of land ;
j——Each district in Henan province.

3.3 Constraints

(1)Land area constraint. All kinds of land area should be equal to total land area of Henan province.

S=1655.36（million hectares）

(2)Economic benefit constraint. The GDP should be no less than the current level through the optimized allocation of land resources.

$$\sum_{j=1}^{18} \sum_{i=1}^{9} w_i k x_{ij} \geq 19724 \quad \text{（billion Yuan）}$$

In expressions:

w_i ——Relative weight coefficients about the benefits of various types of land;
k ——Profit Coefficients.

(3)Coordinate development constraint [5]. The social and economic development area in resources and environment should be allowed limits, its development speed and scale of land resources, environment and the bearing capacity of adaptation. Coordinated development index is used to measure coordination of resource, economics and environment. At the same time, the coordinated development should satisfy the minimum requirements.

$$\mu = \left|\Delta x_{ij}\right| / x_{ij} \leq 15\% \tag{8}$$

In expressions:

μ ——Coordinate development index ;
$\left|\Delta x_{ij}\right| / x_{ij}$ ——Change scope of various types of land areas in each district.

(4)Total population constraint. According to the total population of the population prediction results. Population size is the main factor of structural adjustment of land use, its changes determines the future demand of various types of land. Load bearing population of agricultural land and urban land should be controlled in 2015 and 2030 population planning.

$$D_1\sum_{j=1}^{18}\sum_{i=1}^{5}x_{ij}+D_2\sum_{j=1}^{18}\sum_{i=1}^{5}x_{ij}\le P \tag{9}$$

In expressions,

D_1 ——The average prediction density of farmland;

D_2 ——The average prediction density of urban land;

P ——The population of regional planning.

(5)Arable land constraint. According to the general plans for land use in Henan province (1997-2010), cultivated land area keeps dynamic balance. By 2010, the total amount of cultivated land retains more than 812.03 million hectares. Stable Cultivated area and High Quality improved overall grain production capacity. Therefore, in the planning of arable land in goal by 2010.

$\sum_{j=1}^{18}X_{1j}\ge 812.03$ (million hectares)

(6)Ecological Environment Constraint. In order to satisfy ecological requirements, garden plot and woodland should not be lower than current level.

Garden plot: $\sum_{j=1}^{18}X_{2j}\ge 36.36$ (million hectares)

Woodland: $\sum_{j=1}^{18}X_{3j}\ge 297.27$ (million hectares)

(7)Constraint of actual situation. Residential areas and mining areas are less then current areas; Land area of traffic facilities and water facilities generally exceeds.

Residential areas and mining areas: $\sum_{j=1}^{18}X_{6j}\le 175.22$ (million hectares)

Lands used for transportation: $\sum_{j=1}^{18}X_{7j}\ge 40.61$ (million hectares)

Land for water facilities: $\sum_{j=1}^{18}X_{8j}\ge 31.80$ (million hectares)

(8)Variable non-negative constraints.

$x_{ij}\ge 0$

4 Optimization Calculation and Analysis

According to relevant data from the general land use planning (1997-2010) of Henan province, various types of land utilization are used as optimization variables, land use area as optimal objective, while the value of N_P is twenty and the value of D is nine. After optimizing and computing the data, two planning annual allocation schemes are obtained by 2015, 2030 (Table 1, S=1655.36).The optimization structure of land use planning is more reasonable and sustainable than current situation.

Table 1. Results of optimization (million hectares)

year		x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	total
2010	area	812.03	36.36	297.27	6.45	61.38	175.22	40.61	31.80	194.24	S
	percent	49.05	2.196	17.96	0.39	3.71	10.58	2.45	1.92	11.73	100
2015	area	818.53	45.82	320.50	14.35	64.79	162.10	44.19	34.18	150.90	S
	percent	49.44	2.77	19.36	0.87	3.91	9.79	2.67	2.06	9.11	100
2030	area	830.00	60.54	358.76	24.25	70.47	131.89	50.65	40.53	88.27	S
	percent	50.14	3.66	21.67	1.46	4.26	7.97	3.06	2.45	5.33	100

In Table 1， the 2015 optimum solution: The total amount of cultivated land keeps stable growth. The increase of garden plot and woodland contributes to improving the environment; the decrease of residential areas and mining areas is because of reduction of rural residential area. From the 2030 optimization results, the total amount of arable land basically meets the growing population peak demand. Land-use structure and layout tend to be more reasonable. Benefit of land utilization becomes maximization. Calculated results can provide the reasonable planning and utilization about land resources of Henan province for reference.

References

1. Smith, T.F., Waterman, M.S.: Identification of Common Molecular Subsequences. J. Mol. Biol. 147, 195–197 (1981)
2. May, P., Ehrlich, H.C., Steinke, T.: ZIB Structure Prediction Pipeline: Composing a Complex Biological Workflow through Web Services. In: Nagel, W.E., Walter, W.V., Lehner, W. (eds.) Euro-Par 2006. LNCS, vol. 4128, pp. 1148–1158. Springer, Heidelberg (2006)
3. Foster, I., Kesselman, C.: The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, San Francisco (1999)
4. Czajkowski, K., Fitzgerald, S., Foster, I., Kesselman, C.: Grid Information Services for Distributed Resource Sharing. In: 10th IEEE International Symposium on High Performance Distributed Computing, pp. 181–184. IEEE Press, New York (2001)
5. Foster, I., Kesselman, C., Nick, J., Tuecke, S.: The Physiology of the Grid: an Open Grid Services Architecture for Distributed Systems Integration. Technical report, Global Grid Forum (2002)

Hybrid Differential Evolution for Knapsack Problem

Changshou Deng, Bingyan Zhao, Yanling Yang, and Anyuan Deng

School of Information Science and Technology, JiuJiang University 332005 Jiujiang, China
{csdeng,byzhao,yangyanling,dengany}@jju.edu.cn

Abstract. A hybrid Differential Evolution algorithm with double population was proposed for 0-1 knapsack problem. The two populations play different roles during the process of evolution with the floating-point population as an engine while the binary population guiding the search direction. Each gene of every chromosome in the floating-point encoding population is restricted to the range [-1, 1], while each gene of every chromosome in the binary encoding population is zero or one. A new mapping operation based on sign function was proposed to generate the binary population from the floating-point population. And a local refining operation called discarding operation was employed in the new algorithm to fix up the solutions which are infeasible. Three benchmarks of 0-1 knapsack problem with different sizes were used to verify the new algorithm and the performance of the new algorithm was also compared with that of other evolutionary algorithms. The simulation results show it is an effective and efficient way for the 0-1 Knapsack problem.

Keywords: Knapsack Problem, Hybrid Differential Evolution, sign function, discarding operation.

1 Introduction

Differential Evolution (DE), proposed by Storn and Price [1], is a simple yet powerful algorithm for global optimization over continuous spaces, which use the greedy criterion to make decision. Under the greedy criterion, a new parameters vector is accepted if and only if it reduces the value of the objective. Since its invention, DE has been successfully used in many numerical optimization problems. Chiou (2007) use DE to optimize the large scale economic dispatch problems [2]. Abbass (2002) [3] used DE to train artificial neural network for breast cancer diagnosis. Aydin and Temeltas (2004) [4] applied fuzzy DE to solve the optimal problem of the trajectories of a unicycle mobile robot on a predefined path. Cruz and Willigenburg and Straten(2003) [5] used DE to solve the problem of optimal control. Kaelo and Ali(2006) [6] proposed modifications in mutation and selection rules resulting in two new versions of the original algorithm. Numerical experiments over multiple benchmark test problems showed that the new version of DE algorithms is better than the original DE.

Despite the improvement of DE and successful applications in many engineering fields, its application in solving combinational optimization problems with binary decision variables is still unusual. One of the possible reasons for this lack is that DE cannot keep the closure of result of the original DE mutation operation when it is used in binary domain, for the operations designed in the original DE are designed only for continuous domain.

A new version of DE was proposed in this paper to extend the continuous application field of DE to the binary field. In the new DE, two populations were used to search the solution space in coordination. One population is called floating point population in which each gene of every chromosome was constricted in the range $[-1,1]$. And a local search is called discarding operation was proposed to deal with the infeasible solutions.

The remainder of the paper is constructed as follows. Section 2 gives a brief introduction of the 0-1 knapsack problem. Hybrid DE (HDE) for Knapsack Problems is presented in section 3. Three commonly used benchmarks were used to verify the Hybrid DE and results of the experiments are also reported in section 4. Section 5 concludes this paper.

2 Model of 0-1 Knapsack Problem

The typical 0-1 knapsack problem (KP) is that there are n given items has to be packed in a knapsack of capacity V. Each item has a profit p_i and a weight w_i. The problem is to select a subset of the items whose total profit is a maximized, while whose total weight does not exceed the capacity V. Then the 0-1 knapsack problem can be formulated as (1).

$$\begin{aligned} &Maximize \quad f=\sum_{i=1}^{n} p_i w_i \\ &s.t. \quad \sum_{i=1}^{n} w_i x_i \le V \\ &\qquad x_i \in \{0,1\} \quad (i=1,2,\cdots n) \end{aligned} \tag{1}$$

where $x_i=1$ denotes that the item i is chosen, otherwise $x_i=0$.

3 Hybrid DE for 0-1 Knapsack Problem

3.1 Differential Evolution

The DE is a class of direct search algorithm and the greedy search converges fast but can be trapped by local minima. Running several vectors simultaneously can eliminate this disadvantage. This is the main idea of DE algorithm. DE algorithm is a kind of floating-point encoding evolutionary optimization algorithm. AT present, there have been several variants of DE [1]. One of the most promising schemes, DE/RAND/1/BIN scheme of Storn & Price, is presented in great detail. It is supposed that we are going to find the minimization of the objective function $f(x)$.

3.1.1 Generation of Initial Population

The DE Algorithm starts with the initial target population $X=(x_{ij})_{m\times n}$ with the size m and the dimension n, which is generated by the following way.

$$x_{ij}(0)=x_j^l+rand(0,1)(x_j^u-x_j^l)\ . \tag{2}$$

where $i=1,2,\cdots,m$, $j=1,2,\cdots,n$, x_j^u denotes the upper constraints, and x_j^l denotes the lower constraints.

3.1.2 Mutation Operation

For each target vector $x_i(i=1,2,\cdots,m)$, a mutant vector is produced by

$$h_i(t+1)=x_{r1}+F(x_{r2}-x_{r3})\ . \tag{3}$$

where $i,r_1,r_2\in\{1,2,\ldots,m\}$ are randomly chosen and must be different from each other. And F is the scaling factor which has an effect on the difference between the individual x_{r1} and x_{r2}.

3.1.3 Crossover Operation

DE employs the crossover operation to add the diversity of the population. The approach is given below.

$$u_i(t+1)=\begin{cases}h_i(t+1), & if \quad rand\le CR \ \ or\ j=rand(i)\\ x_i(t), & otherwise\end{cases}\ . \tag{4}$$

where $i=1,2,\cdots,m$, $j=1,2,\cdots,n$, $CR\in[0,1]$ is crossover constant and $rand(i)\in(1,2,\cdots n)$ is the randomly selected index. In other words, the trial individual is made up with some of some components of the mutant individual, or at least one of the parameters randomly selected, and some of other parameters of the target individual.

3.1.4 Selection Operation

To decide whether the trial individual $u_i(t+1)$ should be a member of the next generation, it is compared to the corresponding $h_i(t+1)$. The selection operation is based on the survival of the fitness among the trial individual and the corresponding one such that:

$$x_i(t+1)=\begin{cases}u_i(t+1), f(u_i(t+1))<f(x_i(t))\\ x_i(t), \quad otherwise\end{cases}\ . \tag{5}$$

DE can adapt itself during the search process and find the optimum efficiently and effectively. However, the mutation operation as formula (2) can only keep closure of the search set in the continuous domain. When DE is employed in the discreet domain, this closure of the mutation results must be concerned.

3.2 Hybrid DE for Knapsack Problem

In this section, a hybrid DE for the 0-1 knapsack problems will be discussed. Two populations were used in this hybrid DE. The first population was floating-point encoding with each gene restricted to the interval $[-1,1]$, which as an engine to drive the evolution process in the hybrid DE. The other population is binary encoding which is generated from the first population with a new defined mapping operation. This binary population guided the direction of the evolution during the searching process.

3.2.1 Mapping Operation

A very simple mapping operation was defined with sign function to generate the binary population from the floating-point encoding population. It is as formula (6).

$$[y_1, y_2, \ldots, y_n] = (1 + sign([x_1, x_2, \ldots, x_n]))/2 \ . \qquad (6)$$

An example of mapping results of four individuals where each chromosome has five genes is given in Table 1.

Table 1. Results of mapping operation

floating-point encoding					Binary encoding				
x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3	y_4	y_5
-0.1627	0.69244	0.050305	-0.59471	0.34427	0	1	1	0	1
0.67624	-0.96072	0.36255	-0.24104	0.66359	1	0	1	0	1
0.0056258	0.41894	-0.14222	-0.39077	-0.62069	1	1	0	0	0

3.2.2 Local Discarding Operation

During the evolution of searching the best solution of the knapsack problem, infeasible solutions may appear. A local discarding operation is defined to handle this problem. Firstly, all the items are sorted in profit density ascending order. Then one solution is infeasible, the items are discarded in order of the sorted ascending order until it is feasible.

3.2.3 Coordination between the Two Populations

During the evolution of the hybrid of DE, the floating-point population generated the binary population using the mapping operation, while the selection operation on the binary population with the local discarding operation guides the evolution of the hybrid DE.

3.2.4 Flowchart of the Hybrid DE

The flowchart of the Hybrid DE is illustrated as Fig. 1.

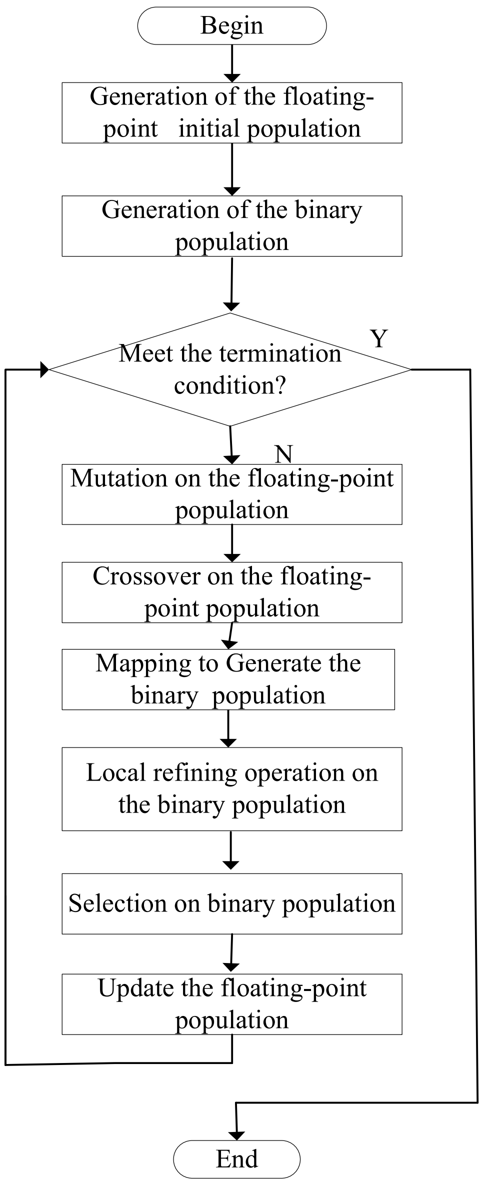

Begin
Generation of the floating-point initial population
Generation of the binary population
Meet the termination condition?
Y
N
Mutation on the floating-point population
Crossover on the floating-point population
Mapping to Generate the binary population
Local refining operation on the binary population
Selection on binary population
Update the floating-point population
End

Fig. 1. The flowchart of hybrid DE

4 Numerical Experiments

In this section, the hybrid DE was used to solve the three 0-1 knapsack problems which are used as benchmark problems in [7]. The knapsack problems are with size of 20, 50 and 100 respectively. For the details about the data, refer [7] please.

The parameters of the hybrid DE is presented in Table 2.

Table 2. Parameters for the three KPs

Examples	Population size	F	CR	Maxgen
Kp1	20	0.5	0.5	30
Kp2	20	0.5	0.5	300
Kp3	20	0.5	0.5	2000

For each of the three knapsack problems, 100 trials have been conducted and the best results (Best), average results (Avg), worst results (Worst) and standard deviations (Dev) are shown in Table 3.

Table 3. The statistical results of the three knapsack problems

Examples	Best	Avg	Worst	Dev
Kp1	1042	1041.3	1037	1.7944
Kp2	3119	3116.8	3110	2.1569
Kp3	26559	26552	26535	8.7952

The optimums of the three Knapsack Problems and its solutions are shown in Table 4.

Table 4. Optimums and solutions

Examples	Optimum	Solutions
Kp1	1042	10111111010111111101
Kp2	3119	1101010111101001101101111 1111100001011011000000010
Kp3	26559	1111111111111111111111111 1111111111111111111110111 1111101000101101101111111 0001110111000000000000001

The results in Table 4 show us that hybrid DE can find the optimums of the three Knapsack Problems with comparable small size of population.

The compared best results with that of several versions of GA are also given in Table 5. The results of the GAs are cited from [7] directly.

Table 5 shows that for each Knapsack Problem, the HDE can find the best known optimum. Particularly, for KP2, the best result got by HDE is the best one in the five different algorithms. And the time that HDE needed is also comparable to other methods.

Table 5. Compared Results with other Algorithms

Algorithm	KP1	KP2	KP3
TGA	1042	3077	25848
GGA	1042	3112	26559
HGA	1037	3103	26487
SEGA	1042	3116	26559
HDE	1042	3119	26559

5 Conclusions

DE is a recently developed heuristic algorithm that has empirically proven to be very efficient for global optimization over continuous spaces. In order to extend the field of DE from the continuous domain to the binary domain, a HDE is proposed. In the HDE, two populations were used to search the best solution coordinately. The floating-point encoding population was use as an engine to drive the search process while the binary encoding population was used to guide the search direction. Two new operations were defined to connect the two populations. One is the mapping operation which was used to generate the binary encoding population from the floating-point encoding population. The other operation is a local discarding operation which was used to refine the infeasible solution. Initial experiments on the three different sizes of Knapsack Problems show it is an effective and efficient way to solve the 0-1 knapsack problem. The mechanism of using two different encoding populations in the HDE can also be imported into other floating-point encoding efficient evolutionary algorithms, such as Evolution Strategy etc. to be applied in the resolving the binary optimization problem.

Acknowledgments. This work is partially support by the National Natural Science Foundation of China under grant No. 50705039. It is also supported by the Science and Technology Foundation of Jiangxi Province ,China under grant No.GJJ09347 and GJJ10616.

References

1. Storn, R., Price, K.: Differential Evolution-A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization 11, 241–354 (1997)
2. Chiou, J.P.: Variable scaling hybrid differential evolution for large scale economic dispatch problems. Electric Power System Research 77, 212–218 (2007)
3. Abbass, H.A.: An evolutionary artificial neural networks approach for breast cancer diagnosis. Artificial Intelligence in Medicine 25, 265–281 (2002)
4. Aydin, S., Temeltas, H.: Fuzzy-differential evolution algorithm for planning time optimal trajectories of a unicycle mobile robot on a predefined path. Advanced Robotics 18, 725–748 (2004)

5. Cruz, L., Willligenburg, V., Straten, G.: Multimodal optimal control problem. Applied Soft Computing 3, 97–122 (2003)
6. Kaelo, P., Ali, M.: A numerical study of some modified differential evolution algorithms. European Journal of Operation Research 169, 1176–1184 (2006)
7. Yan, L., Chao, L.: A Schema-Guiding Evolutionary Algorithm for 0-1 Knapsack Problem. Iacsit-sc. In: 2009 International Association of Computer Science and Information Technology –Spring Conference, pp. 160–164. IEEE Press, Singapore (2009)

Bottom-Up Tree Evaluation in Tree-Based Genetic Programming

Geng Li and Xiao-Jun Zeng

School of Computer Science
University of Manchester
Oxford Road, Manchester, M14 9PL, UK
{lig,x.zeng}@cs.man.ac.uk

Abstract. In tree-based genetic programming (GP) performance optimization, the primary optimization target is the process of fitness evaluation. This is because fitness evaluation takes most of execution time in GP. Standard fitness evaluation uses the top-down tree evaluation algorithm. Top-down tree evaluation evaluates program tree from the root to the leaf of the tree. The algorithm reflects the nature of computer program execution and hence it is the most widely used tree evaluation algorithm. In this paper, we identify a scenario in tree evaluation where top-down evaluation is costly and less effective. We then propose a new tree evaluation algorithm called bottom-up tree evaluation explicitly addressing the problem identified. Both theoretical analysis and practical experiments are performed to compare the performance of bottom-up tree evaluation and top-down tree evaluation. It is found that bottom-up tree evaluation algorithm outperforms standard top-down tree evaluation when the program tree depth is small.

Keywords: Genetic Programming, Tree Evaluation, Top-down Evaluation, Bottom-up Evaluation.

1 Introduction

Genetic programming (GP) [1] is an optimization technique inspired by Darwin's natural selection principle. Genetic programming applies the methodology of genetic algorithm into the domain of computer programs. Both genetic algorithm and genetic programming belong to optimization algorithms in artificial intelligence. Unlike optimization algorithms such as hill climbing or gradient based algorithms, genetic algorithm and genetic programming are population based searching algorithms. Algorithms like hill climbing and gradient based algorithms evolve a single candidate solution, while genetic algorithm and genetic programming search for optimal solution by evolving a population of candidate solutions based on nature. Inspired by nature, GP has been successfully applied to solve a number of problems in a "novel and unexpected way" [2].

But, while applying genetic algorithm and genetic programming to solve practical problems, they have been widely criticized because they are "very slow".

This is mainly due to the fact that genetic algorithm and GP concurrently evolve a population of candidate solutions. Due to the phenomenon of bloating [2], the performance issue is more serious in GP compared with genetic algorithm. Hence, in order to maintain the feasibility of GP, it is essential to improve GP runtime performance.

In GP workflow, the most computation extensive phase is fitness evaluation. As a result, fitness evaluation is the primary target in optimization and a number of optimization methods have been developed. For example, Keijzer [3] examined a number of subtree caching mechanisms to improve the runtime efficiency of fitness evaluation. Teckett [4] developed brood recombination crossover, in which only a small fraction of test cases are evaluated to best estimate offspring's fitness. Jackson [5] introduced fitness-preserving crossover and used it to avoid evaluation of offspring produced by fitness-preserving crossover. In [6], a statistical method was developed to select a fraction of test cases to evaluate rather than using all test cases. In [7], a genetic programming population clustering method was proposed to decrease the total number of fitness evaluation performed.

The most widely used tree evaluation algorithm is top-down evaluation. Top-down tree evaluation evaluates program tree from the root to the leaf of the tree. In this paper, we identify a scenario when standard top-down tree evaluation fails to perform well. Then a new *bottom-up* tree evaluation algorithm is developed to explicitly address the problem identified. The rest of the paper is organised as follows. In the next section, we firstly introduce the motivation of developing bottom-up tree evaluation and then discuss the algorithm implementation. Then, we perform a theoretical analysis of bottom-up evaluation. In section 4, we use Multiplexer 11 domain to compare the performance of top-down and bottom-up evaluation. We conclude this paper with a discussion of when bottom-up evaluation outperforms top-down evaluation and further works.

2 Bottom-Up Tree Evaluation

In fitness evaluation, we typically have got a number of test cases. Each individual program in the population needs to execute with inputs from each test case. Then the output is compared with expected result. The raw fitness usually is how many times the program produces "correct" output. Standard tree evaluation works top-down recursively starting from the root node (see Figure 1). The sequence that nodes evaluate is the same as a depth-first traverse of the tree.

In some problems, for example Multiplexer problem, the test cases are permutation of input values (or semi permutation of input values). For example, in 11 Multiplexer problem, there are 11 inputs. Since each input is binary (can either be 0 or 1), the permutation of inputs has $2^{11} = 2048$ different cases. Although every value in this permutation is distinct, if we iterate through this permutation, we can find that not *every* input changes every time. For example, for the permutation in Figure 2, from Case 1 to Case 2, only input D7 is changed. From Case 2 to Case 3 only D6 and D7 changes. If we let $C(x)$ be the number of

```
def TopdownEval(node) ->
    if node is Terminal ->
        return value of Terminal input
    if node is Function ->
        foreach input in node's inputs ->
            TopdownEval(input)
        return evaluatedResult

TopdownEval(root)
```

Fig. 1. Pseudo Code for Top-down Evaluation of Program Tree

```
Inputs:  A0 A1 A2 D0 D1 D2 D3 D4 D5 D6 D7
Values:  0  0  0  0  0  0  0  0  0  0  0   Case 1
         0  0  0  0  0  0  0  0  0  0  1   Case 2
         0  0  0  0  0  0  0  0  0  1  0   Case 3
         0  0  0  0  0  0  0  0  0  1  1   Case 4
         0  0  0  0  0  0  0  0  1  0  0   Case 5
         .  .  .  .  .  .  .  .  .  .  .
         .  .  .  .  .  .  .  .  .  .  .
         .  .  .  .  .  .  .  .  .  .  .
         1  1  1  1  1  1  1  1  1  0  0   Case 2045
         1  1  1  1  1  1  1  1  1  0  1   Case 2046
         1  1  1  1  1  1  1  1  1  1  0   Case 2047
         1  1  1  1  1  1  1  1  1  1  1   Case 2048
```

Fig. 2. Viewing test case inputs as permutation in Multiplexer 11 problem

changes in a permutation for an input x, then for a given input x whose index is n (index starts 1 from left to right) and the total number of inputs is N, then:

$$C(x) = \frac{2^N}{2^{N-n}} = 2^n \tag{1}$$

For example, for the permutation in Figure 2:

$$C(D7) = \frac{2048}{2^0} = 2048, C(A0) = \frac{2048}{2^{10}} = 2.$$

Given the above fact, now let's consider evaluating the following program tree using the above permutation as test case inputs:

Not(And(Or(A0, A1), D7))

And's left subtree Or(A0, A1)'s value changes at maximum 4 times ($C(A1) = 4$), while the right subtree D7's value changes 2048 times. Using Top-down evaluation algorithm, the left subtree evaluates 2048 times while most of the evaluations give the same result. In fact, since the left subtree only changes at maximum 4 times, 4 evaluations are enough. Thus, for this subtree, $2048 - 4 = 2044$ evaluations are wasted. In Multiplexer 11 problem, this may not be a very big issue since functions like And, Or, Not, If are very fast to execute. But for problems whose function set consists of very complex functions, the top-down evaluation algorithm is not very efficient.

The above scenario gives an example when top-down evaluation fails to perform good. Going back to the example, if the node And has a single cache which stores the previous result of left subtree, then when And evaluates, it takes the new value of D7 and then compute output using value stored in cache. In this way, then the left subtree of And does not need to be re-evaluated when both A0 and A1 are not changed, i.e. the above problem of top-down evaluation is solved.

But how does node And know if A0 or A1 is changed or not without checking them explicitly? The bottom-up tree evaluation solves this problem. In bottom-up tree evaluation, each node in the tree has a single cache which stores the previous result of this node. The value in the cache can be accessed by other nodes. There is also a queue needed for evaluation process. The tree evaluation has two phases. In the first phase, every terminal node is checked if the new value equals previous value stored in cache or not. If two values are not equal, then the cache is updated and the parent of the node is enqueued into the central queue. In the second phase, every node stored in the central queue is dequeued. The node is re-evaluated and if the output of the node changes, the node's parent is enqueued. This process repeats until the queue is empty. Then the output of this evaluation is stored in the root node's cache. The bottom-up evaluation can be viewed as the pseudo code in Figure 3. Intuitively, bottom-up evaluation outperforms top-down evaluation because in bottom-up evaluation, only "necessary" evaluations are performed, redundant unnecessary evaluations in top-down evaluations are eliminated. In addition, the percentage of saving also depends on the particular permutation of test cases. This is because for a given tree, the performance of bottom-up evaluation varies for different permutation of test cases. In the rest

```
def BottomupEval(root) ->
    // Phase 1
    foreach terminal node ->
        if terminal node's value changes ->
            update local cache
            enqueue parent
    // Phase 2
    while queue is not empty ->
        node = queue dequeue
        evaluates node
        if node's value changes ->
            update local cache
            enqueue parent
    return root cache value
```

Fig. 3. Pseudo Code for Bottom-up Evaluation of Program Tree

of this paper, we perform detailed analysis of the performance of bottom-up evaluation algorithm.

3 Theoretical Analysis

The bottom-up evaluation adds a central queue and local cache for each node into tree evaluation process in order to eliminate wasted function evaluation in top-down evaluation. It is assumed that function node evaluation is complex and the overheads introduced by queue operations can be neglected compared with savings. Based on this assumption, the performance of both top-down and bottom-up evaluation links to the number of function calls on non-terminal nodes in a tree. Using number of function calls as comparison criteria, the performance of both algorithms can be theoretically analyzed.

Assuming each function's arity is two (each function has two inputs) and there are N different inputs (terminals). Considering a Full tree T of depth d. Since in a full tree, all nodes of depth smaller than d are function nodes, the total number of nodes in T is:

$$Num(T) = 2^d - 1$$

The number of non-terminal nodes is:

$$NonNum(T) = 2^{d-1} - 1$$

Let the total number of test cases be $N_{testcases}$, so the number of function calls for Top-down evaluation (i.e. the performance of top-down evaluation) is:

$$P_{Topdown}(T) = (2^{d-1} - 1) \cdot N_{testcases} \tag{2}$$

The equation 2 only works for some problem domains. Equation 2 assumes that the tree is traversed (every node is evaluated) while using top-down evaluation. This is true for problem domains such as symbolic regression. In symbolic regression problem, all functions in function set are mathematical operators. In this case, all children nodes of the function node needs to be evaluated.

But in problem domains such as Multiplexer, the tree is not traversed in top-down evaluation. In Multiplexer domain, there are four functions: And, Or, If and Not. For function Not, the child node of Not always evaluates. For function If, only two children nodes evaluate every time (the condition and the according action node). And and Or function can be implemented in two different ways: bitwise or short-circuit. In short-circuit implementation of And, the first child of And always evaluates, the second child evaluates only when the first child evaluates true. Similarly, in short-circuit Or, the second child evaluates only when the first child is false. Because this nature of Multiplexer function set, when tree is top-down evaluated, only a fraction of nodes are actually evaluated, and hence equation 2 does not hold in Multiplexer domain.

To calculate the number of function calls in domains like Multiplexer, we need an explicit measurement of the above feature of function set. This feature can be studied using the concept of activation rate. This concept is formally defined and studied in [8]. The following is a brief introduction of activation rate.

Definition 1 (Activation). *Let x be a node in a tree T, in Top-down evaluation, an* Activation *of x is when x is evaluated. The number of activations of node x for a set of test cases Tcs is denoted as $\Lambda_{Tcs}(x)$.*

We can further define the *Rate of Activation*:

Definition 2 (Activation Rate of Node). *Let x be a node in a tree T, Tcs be a test cases set which contains N test cases, the* Activation Rate *of node x, $\Theta_{Tcs}(x)$ is:*

$$\Theta_{Tcs}(x) = \frac{\Lambda_{Tcs}(x)}{N}$$

With both definition above, we can define the *Activation Rate of Tree*:

Definition 3 (Activation Rate of Tree). *Let x be a node in a tree T, $N(T)$ be the number of nodes in T, Tcs be a test cases set which contains N test cases, the* Activation Rate *of tree T, $\Theta_{Tcs}(T)$ is:*

$$\Theta_{Tcs}(T) = \frac{\sum_{x}^{x \in T} \Theta_{Tcs}(x)}{N(T)}$$

Using the concept of activation rate, the true function call counts of top-down evaluation in any domain C_{True} and the function call counts using top-down traverse evaluation $C_{Traverse}$ has the following relationship:

$$C_{True} = \Theta \cdot C_{Traverse} \tag{3}$$

where Θ is the average activation rate of all non-leaf nodes in a program tree. Clearly, we can see that equation 2 implicitly assume that $\Theta = 1$. In problem domains like symbolic regression, program tree's activation rate $\Theta \equiv 1$, while in problem domains like Multiplexer, $\Theta \leq 1$. So, for a full tree T, equation 2 can be generalized using equation 3:

$$\begin{aligned} P_{Topdown}(T) &= C_{True} \\ &= \Theta \cdot C_{Traverse} \\ &= \Theta \cdot P_{Traverse} \\ &= \Theta \cdot (2^{d-1} - 1) \cdot N_{testcases} \end{aligned} \tag{4}$$

In Bottom-up evaluation, let X(T) be the set of terminals in a tree T, then:

$$P_{Bottomup}(T) = \sum_{x}^{x \in X} C(x)E(X) \tag{5}$$

Where C(x) is defined in Equation 1, and E(X) is the average number of nodes affected when some $x \in X$ changes. The exact form of E(X) is very hard to deduct because it is related to not only which terminal is in X but also how those terminals are connected in the tree. Without further assuming the tree structure, which leads to loss of generality of analysis, it is impossible to estimate E(X). Similarly, since the value of Θ in (4) also depends on the shape of tree, without further assuming the tree structure, it is impossible to estimate $P_{topdown}$ for problem domain like Multiplexer.

But we can perform worst case scenario analysis for problem domains whose tree's activation rate $\Theta \equiv 1$. Worst case scenario happens when every terminal x in X has index $n = N$. In this case, the cache has no effect because every input changes every time.

$$E(X) = 2^{d-1} - 1 \tag{6}$$

Substituting C(x) and E(X) in (5) with (1) and (6), we get:

$$P_{Bottomup}(T) = 2^n \cdot (2^{d-1} - 1)$$

Since $n = N$ and $2^n = N_{testcases}$, so we get:

$$P_{Bottomup}(T) = N_{testcases} \cdot (2^{d-1} - 1)$$

Which is the same as $P_{Topdown}(T)$. So bottom-up evaluation performs the same as top-down evaluation in the worst case for problem domains in which $\Theta \equiv 1$, i.e.

$$P_{Bottomup}(T) \leq P_{Topdown}(T).$$

Now considering the probability that worst case scenario happens. For a full tree T of depth d, the probability p:

$$p = (\frac{1}{N})^{2^{d-1}},$$

Which is very small.

4 Experiments in Multiplexer 11 Domain

Theoretical analysis in previous section qualitatively shows bottom-up evaluation is better than top-down evaluation for problem domains in which $\Theta \equiv 1$. A quantitative analysis of how much better the bottom-up evaluation cannot be theoretically deducted. In addition, the performance of bottom-up evaluation for domains in which $\Theta \leq 1$ cannot be theoretically analyzed without further assuming the structure of tree evaluated. To fill both gaps, this section uses experiments in Multiplexer 11 problem domain to quantitatively compare performance of bottom-up evaluation and top-down evaluation.

In the first experiment, we compare performance of bottom-up and top-down evaluation for problem domain in which $\Theta \equiv 1$. Although in Multiplexer 11 domain, the average activation rate of non-leaf nodes are generally smaller than 1, we can artificially converts it to 1. This is done by converting short-circuit And and Or to bitwise And and Or. For If, both true subtree and false subtree are evaluated. Using this approach, when a program tree is evaluated, every node is traversed. We call this traversing evaluation top-down full evaluation.

Table 1. Top-down Full and Bottom-up Evaluation Performance (7000-trees)

Depth	Size	Top-down Full	Bottom-up	Performance Enhance (%)
2	3.4571	2048	373	81.79
3	6.9253	5960	1062	82.18
4	13.7057	12935	2643	79.57
5	26.9746	26545	5748	78.35
6	53.4802	53880	12127	77.49
7	103.3013	104852	24029	77.08
8	195.7506	199130	47073	76.36
9	373.1875	380656	88593	76.73
10	258.6831	263774	65297	75.25
11	441.6177	450245	110474	75.46
12	683.4360	698935	172932	75.26
13	1102.4485	1127861	278861	75.28

In this experiment, we compare top-down full evaluation and bottom-up evaluation. The experiment is designed as follows. 7000 trees are randomly generated using ramp-half-and-half method [1]. The depth of generated tree ranges from 2 to 13. Duplicated trees are removed. we evaluate it twice firstly using top-down full approach and then bottom-up approach. Then we compare the number of non-leaf node calls using each method. The experiment result can be found in table 1. From the experiment data, we can find that a substantial amount of function calls ($\geq 75\%$) is saved using bottom-up tree evaluation. This experiment result verifies the theoretical conclusion we deducted in the previous section: bottom-up tree evaluation outperforms top-down evaluation for problem domains in which $\Theta \equiv 1$.

In the second experiment, we compare performance of bottom-up and top-down evaluation for domain in which $\Theta \leq 1$. The experiment is designed as follows. Using the same 7000 trees generated in the first experiment, we evaluate each tree twice firstly using top-down and then bottom-up evaluation algorithm. We also record the average activation rate of non-leaf nodes in top-down evaluation. The experiment result is summarized in table 2.

From experiment data in table 2, we can find that bottom-up evaluation outperforms top-down evaluation when tree depth is small. But as the depth of the tree increases, the performance enhancement becomes smaller. For very deep trees (depth bigger than 8 in table 2), top-down tree evaluation outperforms bottom-up evaluation. This is mainly because as the depth of the tree increases, the average activation rate of non-leaf nodes (Θ) decreases.

Let P1 be the performance enhance of bottom-up evaluation compared with top-down full evaluation and P2 be the performance enhance of top-down evaluation compared with top-down full evaluation. From the first experiment, we know that in Multiplexer 11 domain, $P1 \approx 0.75$. Since from (4), we know that:

$$P_{Topdown} = \Theta \cdot P_{TopdownFull}$$

Table 2. Top-down and Bottom-up Evaluation Performance (7000-trees)

Depth	Size	Top-down	Bottom-up	Performance Enhance (%)	Θ
2	3.4571	2048	373	81.79	1.0000
3	6.9253	4964	1062	78.61	0.8482
4	13.7057	8908	2643	70.33	0.7249
5	26.9746	14797	5748	61.15	0.6014
6	53.4802	23583	12127	48.58	0.4832
7	103.3013	35437	24029	32.19	0.3834
8	195.7506	51861	47073	9.23	0.3063
9	373.1875	77173	88593	-14.80	0.2341
10	258.6831	48792	65297	-33.83	0.2214
11	441.6177	62075	110474	-77.97	0.1633
12	683.4360	75248	172932	-129.82	0.1248
13	1102.4485	95206	278861	-192.90	0.0979

So:

$$\begin{aligned} P2 &= \frac{P_{TopdownFull} - P_{Topdown}}{P_{TopdownFull}} \\ &= 1 - \Theta \end{aligned}$$

When the tree depth is small (smaller than 8 in table 2), $\Theta > 0.25$. So, $P1 > P2$. When the tree depth is bigger than 8 in table 2, the $\Theta < 0.25$, then $P1 < P2$. So, in general, the performance of bottom-up evaluation compared with top-down evaluation P:

$$\begin{aligned} P &= \frac{P_{Topdown} - P_{Bottomup}}{P_{Topdown}} \\ &= \frac{(P_{TopdownFull} - P_{Bottomup}) - (P_{TopdownFull} - P_{Topdown})}{\Theta \cdot P_{TopDownFull}} \\ &= \frac{1}{\Theta} \cdot (\frac{P_{TopdownFull} - P_{Bottomup}}{P_{TopdownFull}} - \frac{P_{TopdownFull} - P_{Topdown}}{P_{TopdownFull}}) \\ &= \frac{1}{\Theta} \cdot (P1 - P2) \\ &= \frac{P1 - 1 + \Theta}{\Theta} \end{aligned}$$

Because, Θ negatively relates to the depth of the tree [8], bottom-up tree evaluation outperforms the top-down evaluation algorithm when the depth of the tree is small.

5 Conclusion and Further Work

In this paper, we introduce a new tree evaluation method, bottom-up evaluation. Bottom-up evaluation is suitable for problem domains, in which the number of inputs are large and not all inputs changes in each test case. We also experiment the performance of bottom-up evaluation and top-down evaluation using

Multiplexer 11 problem domain. Experiments show that bottom-up evaluation outperforms top-down evaluation when the tree depth is not very big. We also develop theoretical explanation of this phenomenon using the concept of activation rate.

Based on experiment data, we know that top-down tree evaluation performs well when tree is large while bottom-up evaluation performs well when tree is small. As a result, bottom-up evaluation is suitable for early generations in GP runs. In addition, although two evaluation methods are very different, this does not mean they are mutually exclusive. In fact, two methods can be combined: we can develop a hybrid evaluation method in which, some portion of trees are evaluated Top-down while other parts are evaluated Bottom-up. In this way, we can combine the strength of both approach. This is a good direction for further investigation of bottom-up evaluation.

References

1. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. The MIT Press, Cambridge (1992)
2. Poli, R., Langdon, W.B., McPhee, N.F.: A field guide to genetic programming (2008)
3. Keijzer, M.: Alternatives in subtree caching for genetic programming. In: EuroGP, pp. 328–337 (2004)
4. Tackett, W.A.: Recombination, selection, and the genetic construction of computer programs. PhD thesis, Los Angeles, CA, USA (1994)
5. Jackson, D.: Fitness evaluation avoidance in boolean GP problems. In: Proceedings of the 2005 IEEE Congress on Evolutionary Computation, Edinburgh, UK, vol. 3, pp. 2530–2536. IEEE Press, Los Alamitos (2005)
6. Giacobini, M., Tomassini, M., Vanneschi, L.: Limiting the number of fitness cases in genetic programming using statistics. In: Guervós, J.J.M., Adamidis, P.A., Beyer, H.-G., Fernández-Villacañas, J.-L., Schwefel, H.-P. (eds.) PPSN 2002. LNCS, vol. 2439, pp. 371–380. Springer, Heidelberg (2002)
7. Xie, H., Zhang, M., Andreae, P.: Population clustering in genetic programming. In: Collet, P., Tomassini, M., Ebner, M., Gustafson, S., Ekárt, A. (eds.) EuroGP 2006. LNCS, vol. 3905, pp. 190–201. Springer, Heidelberg (2006)
8. Li, G.: A novel approach to model genetic programming behaviors based on activation rate. Research Report, School of Computer Science, University of Manchester (2009)

Solving Vehicle Assignment Problem Using Evolutionary Computation

Marina Yusoff[1], Junaidah Ariffin[2], and Azlinah Mohamed[1]

[1] Faculty of Computer and Mathematical Sciences
Universiti Teknologi MARA
40450 Shah Alam, Selangor
Malaysia
{marinay,azlinah}@tmsk.uitm.edu.my
[2] Flood-Marine Excellence Centre
Faculty of Civil Engineering
Universiti Teknologi MARA
40450 Shah Alam, Selangor
Malaysia
junaidahariffin@yahoo.com

Abstract. This paper examines the use of evolutionary computation (EC) to find optimal solution in vehicle assignment problem (VAP). The VAP refers to the allocation of the expected number of people in a potentially flooded area to various types of available vehicles in evacuation process. A novel discrete particle swarm optimization (DPSO) algorithm and genetic algorithm (GA) are presented to solve this problem. Both of these algorithms employed a discrete solution representation and incorporated a min-max approach for a random initialization of discrete particle position. A min-max approach is based on minimum capacity and maximum capacity of vehicles. We analyzed the performance of the algorithms using evacuation datasets. The quality of solutions were measured based on the objective function which is to find a maximum number of assigned people to vehicles in the potentially flooded areas and central processing unit (CPU) processing time of the algorithms. Overall, DPSO provides an optimal solutions and successfully achieved the objective function whereas GA gives sub optimal solution for the VAP.

Keywords: Discrete Particle Swarm Optimization, Evacuation Process, Evolutionary Computation, Genetic Algorithm, Vehicle Assignment Problem.

1 Introduction

Manual process of vehicle assignment adopted by the National Security Council (NSC) of Malaysia had reported uneven distribution of transport, non-timely assistance, and poor coordination at operational level [1]. The above is evident during one of the worst flash floods in Kota Tinggi district in Johor State of Malaysia where more than ten thousand flood victims need to be evacuated to the relief centers. A solution to the above problem is vital to assist the NSC and other related agencies in managing and monitoring the evacuation process. This paper addresses the solution for the assignment of vehicle to the expected number of people at the potentially flooded areas using datasets from selected flash flood events.

Evolutionary computation (EC) algorithms for solving a combinatorial problem have been reported in various types of assignment problem [2][3]. Most of them compare discrete particle swarm optimization (DPSO) and genetic algorithm (GA) to obtain solution. In most cases DPSO provides better solution in comparison to genetic algorithm (GA). Limited research was found related to vehicle assignment problem (VAP) in emergency evacuation processes [4]. In previous investigations several approaches; mathematical method, exact algorithms, and heuristic algorithms [5] have been used to solve only evacuation process but fail to consider assignment of vehicles in solving evacuation processes. Nevertheless, DPSO has given near optimal solution to VAP using min-max approach based on minimum capacity of vehicles and average capacity of vehicles using small data sets [6]. This paper would however introduces the use of min-max approach with emphasis on the minimum capacity of vehicles and maximum capacity of vehicles.

It was discovered that GA is very much competitive to that of particle swarm optimization (PSO) [3]. Two evolutionary computation (EC) algorithms namely DPSO and GA are addressed to seek for VAP solution. Solutions obtained through DPSO and GA with respect to their fitness value and CPU processing time will be discussed. Fitness value refers to the maximum number of people that were assigned to vehicles divided by the number of expected people at each potentially flooded area. Details of the calculation of the fitness value are explained in Section 3.

The organizational structure of this paper is as follows. Section 2 presents PSO and the novel DPSO is discussed in Section 3. Discussion on the computational results are made in Section 4. In Section 5 we discuss the analysis of results of the algorithms. Section 6 presents conclusion and recommended future work.

2 Particle Swarm Optimization

PSO is a population-based stochastic approach grouped under swarm intelligence [7] to solve continuous and discrete problems. It was introduced by Kennedy and Eberhart [8]. PSO indicates the velocity and position of particles in multi-dimensional space. PSO is able to explore regions of the search space and exploit the search to refine a feasible solution. Many researchers have demonstrated the advantages of using PSO to solve several types of problems [2][9][10][11].

Improvements for PSO have been developed to measure the performance of PSO for various types of problems [12][13] and across standard benchmark datasets [10][12][14]. For example, the canonical PSO applies inertia weight in updating velocity to simulate the social behavior of birds. After two years of the PSO development, a discrete problem had concentrated on the initial work involving discrete binary PSO introduced by [15]. They proposed a new way of updating the position of particles to accommodate a discrete binary problem. This approach was then improved in several studies based on a benchmark dataset [16][17] and a real-world situation [2][18] of the discrete problem. When compared to other optimization methods, the performance of PSO is competitive to the genetic algorithm [2]. This demonstrates that PSO, with its global search capability and local exploitation, is a promising approach to finding the optimum solution and, therefore, it should be further investigated.

3 A Novel Discrete Particle Swarm Optimization

Discrete particle position from [6] is adopted to represent the number of vehicles for each potentially flooded area. The initialization of these values is randomly generated based on the expected number of vehicles allocated for the flood victims from the potential flooded areas. We introduced min-max approach for the discrete particle position generation as demonstrated in Equation 1, Equation 2, and Equation 3. The minimum discrete position values were initialized taken into account the minimum capacity whereas the maximum discrete position value refers to the maximum capacity of vehicles as shown in Equation 1 and Equation 2, respectively. Equation 3 shows the calculation of the initial discrete particle position.

$$P_{min} = T / C_{min} \tag{1}$$

$$P_{max} = T / C_{max} \tag{2}$$

where, P_{min} is the minimum position value, P_{max} is the maximum position value, T is the total number of people in the potentially flooded area, C_{min} is the minimum capacity of vehicle, and C_{max} is the maximum capacity of vehicle.

$$D = P_{min} + r\,(P_{max} - P_{min}) \tag{3}$$

where D is the initial discrete position value for DPSO (genes in GA), and parameter r is the random value in the range of [0,1]. DPSO algorithm as illustrated in Figure 1, starts with the initialization of the number of particles. Particles are represented in the form of matrices, X_{ij}, where i is the number of dimension of particle or the size of swarm and j is the number of the potentially flooded area. Step 3 declares parameters; W is the inertia weight and C_1 and C_2 are the acceleration constant parameters. Step 4 initializes velocity, V, using minimum velocity (V_{min}) and maximum velocity (V_{max}) of each particle using Equation 4. Step 5 involves the initialization of discrete position value using Equation 3 as mentioned in Section 2.3. The calculation of P_{best} is shown in step 8 and it is referring to Equation 5.

$$V = V_{min} + r\,(V_{max} - V_{min}) \tag{4}$$

$$P_{best} = t_a / t_l \tag{5}$$

where, t_a is the number of people assigned to the respective vehicles, meanwhile t_l is the total number of people in all potentially flooded area. P_{best} is the personal best of the i^{th} particle, and G_{best} is the fitness value from the best position derived from all particles in the swarm. Line number 11 and 12 are the calculation of new velocity value and new discrete particle position using Equation 6 and Equation 7, respectively [19].

$$V_{ij(new)} = W \text{ x } V_{ij(old)} + C_1 \text{ x } r\,(P_{best} - X_{ij(old)}) + C_2 \text{ x } r \text{ x } (G_{best} - X_{ij(old)}) \tag{6}$$

$$X_{ij(new)} = X_{ij(old)} + V_{ij(new)} \tag{7}$$

The next three lines, tries to find discrete particle positions that will lead to the best value of *G_{best}*. Lines17 to 20 are steps for the decision of the assignment of people to vehicles depending on the selected discrete particle position. Line 20, involves assign and re-assigned people until all the discrete particle position are allocated to vehicles. Finally, $P_{best\ (new)}$ and $G_{best\ (new)}$ are determined. The number of iteration of these algorithm starts from line 9 until line 23 and stop upon satisfying the stopping condition.

```
1.Begin
2.    Initialize number of particles
3.    Declare W, $C_1$ and $C_2$
4.    Initialize $V_{min}$ and $V_{max}$ and calculate Velocity, V using a
      random initialization
5.    Initialize the available vehicles with a random capacity
6.    Initialize D using min-max approach
7.    Assign vehicle according to the initialize discrete
      particle position, D
8.    Calculate $P_{best}$ and $G_{best}$ value for each particle
9.  Do
10.     For each particle
11.       Calculate $V_{(new)}$
12.       Calculate $D_{(new)}$
13.       If $G_{best\ (new)}$ is more than or equal $G_{best\ (old)}$
14.          employ discrete particle position, D
15.       else
16.          employ discrete particle position, $D_{(new)}$
17.       If (D is equals or less than the available vehicles)
18.          Assign people to vehicles according to the number of
             the employed discrete position, D
19.       else
20.          Assign and re-assigned people until all the employed
             discrete particle position are allocated to
             vehicles
21.       Calculate $P_{best\ (new)}$
22.       Calculate $G_{best\ (new)}$
23.  While (stopping condition is reached)
24.End
```

Fig. 1. A Novel DPSO Algorithm

4 Computational Results

This section presents the computational results to examine the quality of solution of the DPSO algorithm and GA in VAP. The quality of solution on two aspects were evaluated; fitness value and CPU processing time. The following sub sections discuss the experiment setup and results of DPSO and GA according to the aspect of the quality of solutions.

4.1 Experimental Setup

Table 1 shows the list of parameter for this experiment. The inertia weight of 0.9 was selected based on the recommendation of [20]. Statistical analysis were used to confirm the result inclusive of descriptive measurement; average, standard deviation, minimum and maximum. A paired t-tests were used to analyze the relationship between DPSO algorithm and GA based on 95 percents of confidence level (α=0.05). The dataset was gathered from the major flooded areas in Kota Tinggi district in Johor state that were badly affected in the December 2006 and January, 2007. Table 2 depicts the number of flooded areas and its total number of flood victims. Various types of vehicles of different standard capacities were used in the evacuation operation. Sixty seven of the available vehicles with a total of 650 in capacity were identified to be used before and during flash flood in this district.

Table 1. List of parameters

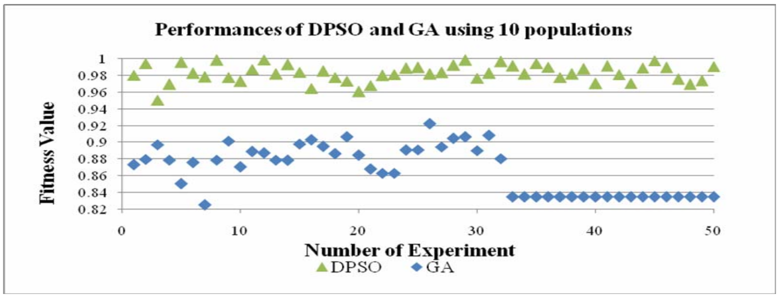
Parameter	Value
W	0.9
C_1	2.5
C_2	1.5
Available vehicles	67
Initial V_{min}	-4
Initial V_{max}	4
Sequence of vehicle capacity	Random
Stopping condition	*while* (G_{best} < 0.6)

Table 2. List of flooded areas and its flood victims' number

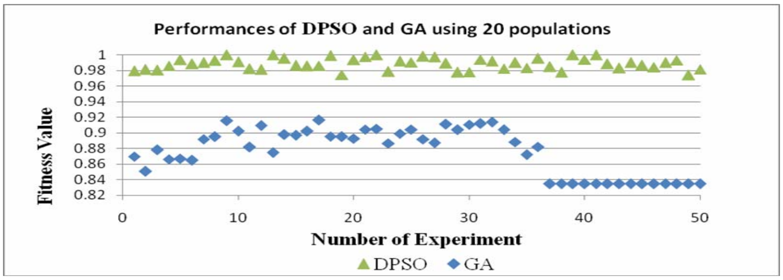
Flash flood event	Number of flood victims	Number of flooded areas
December , 2006	13112	26
January, 2007	5352	35

4.2 Performance of DPSO and GA According to Its Fitness Values

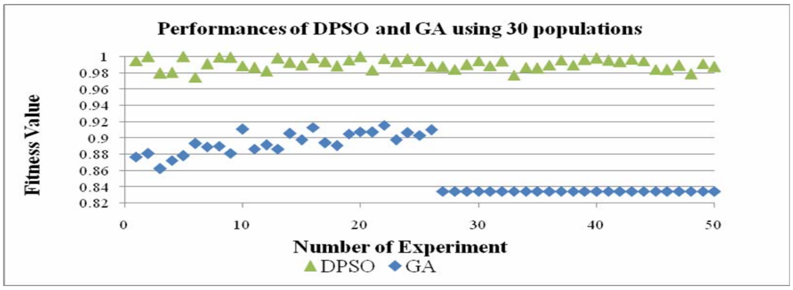
Figure 2 shows the fitness values tabulated from a different number of populations tested for DPSO and GA using dataset of December, 2006. It involves 26 flooded areas with a total of 13,112 flood victims. Generally, the scatter graph in Figure 2(a) and 2(b) present a similar pattern in fitness value for both GA and DPSO. It is apparent that the GA has resulted in the lowest value of fitness compared to DPSO. As can be seen in figures 2(a), 2(b), and 2(c) the fitness values are in the range of 0.82 to 0.94. This confirms that GA gives sub optimal solution for the VAP. Contrary to expectations, the fitness values generated using DPSO, in figure 2(a), 2(b), and 2(c) indicate that the performance of DPSO is better compared to GA for the three populations. This result is significant with a significant value equals to 0.00 on paired t-test performance. A strong evidence of good solution to DPSO was found in both figures, 2(b) and 2(c). These figures demonstrate that the fitness values remained steady between 0.96 and 1.00. In summary, DPSO had given optimal solution with all



(a)



(b)



(c)

Fig. 3. Performance of DPSO and GA based on the generated fitness value (a) 10 populations, (b) 20 populations, (c)30 populations

people successfully assigned to the vehicles at the potential flooded. An indication of the above is illustrated at the fitness value or G_{best} equals to one as can be seen in figure 2(b) and 2(c). Based on the above results, it is recommended for DPSO to use 30 populations for optimal solution. Other results are shown in Table 3 to confirm this argument.

Table 3 demonstrates that the fitness value for DPSO gave similar results for ~~the~~ both datasets on all experiments. DPSO gives higher average of fitness value compared to GA The maximum value of GA obtained is 0.92 for December, 2006 and 0.91 for January, 2007. The good performance of DPSO to GA is supported by the results of a paired t-test with a significant value equal to 0.00 ($\alpha < 0.05$). The overall performance of DPSO concludes that the objective function is achieved. The result from [6] had reported less than 0.95 for the average of fitness value. Thus, the use of min-max approach using minimum capacity and maximum capacity of vehicles in the initialization of discrete particle position gives better results with the assignment of larger number of people.

Table 3. Performance of DPSO and GA according to its fitness value

Dataset	DPSO				GA			
	Min	Max	Avg	Std. dev	Min	Max	Avg	Std. dev
December, 2006	0.97	1.00	0.99	0.01	0.83	0.92	0.86	0.03
January, 2007	0.97	1.00	0.99	0.01	0.82	0.91	0.87	0.02

4.3 Performance of DPSO and GA According to Its CPU Processing Time

This section presents the second aspect of quality solution which is CPU processing time. Both algorithms were tested to see its performance using the same data set as used in the previous section. There is an increase of CPU processing time for both GA and DPSO as shown in Table 4. This indicates that the number of people influences the CPU processing time. It can be seen from the table that the average score of CPU processing time for GA is fewer than DPSO for December, 2006. The minimum processing time for DPSO is about 18 seconds while the minimum processing time for GA is about 12 seconds. However, DPSO gives better CPU processing time for January, 2007 in about 12 seconds.

Table 4. Performance of DPSO and GA according to its CPU processing time

Data set	DPSO with CPU processing time (milliseconds)				GA with CPU processing time (milliseconds)			
	Min	Max	Avg	Std. dev	Min	Max	Avg	Std. dev
December, 2006	18794	65328	55188.40	10632.76	12090	72195	40522.66	16651.44
January, 2007	8949	77248	12827.26	9792.59	10424	35370	27341.96	5533.78

5 Analysis of Results

This section discusses the analysis of the results demonstrated in the previous section. We divide the discussion into two perspectives; DPSO perspective and GA perspective.

5.1 DPSO Perspective

The introduction of min-max approach establishes the initialization of minimum discrete particle position and maximum discrete position vehicles for each potentially flooded area yields a limitation in search space for each particle. This approach starts with the random initialization of discrete particle position as shown in equation 3. Prior to this initialization, the minimum discrete particle position and the maximum discrete position are calculated (see equation 1 and equation 2) for each potentially flooded area. Thus, the assignment of the expected number of people at each potential flooded area is within the range of the required number of vehicles. To obtain optimal solution (G_{best} equals to 1), the generated discrete particle position must be able to accommodate the number of people with different capacity of vehicles. Reassignment of the same vehicles is carried out with the aim to accommodate the discrete particle position. As illustrated in the above results, optimal solution can be obtained using 30 populations, but improvement is required for CPU processing time in conjunction with velocity clamping procedure.

The use of inertia weight equals to 0.9 proposed by [20], confirms to be suitable for the DPSO in solving the VAP for optimal solution. Inertia weight and random value in the range of [0,1] are used in the calculation of new velocity for each particle lead to a small range of the new generation of velocity. In addition, the initialization of random velocity also has an implication to the calculation of a new velocity value. A small range of velocity between 0.4 and -0.4, inertia weight, and random value (r) contributes to a small difference for the new generation of discrete particle position when compared to the discrete particle position generated in [6].

5.2 GA Perspective

Analyses have proven that GA produced sub optimal solution for vehicle assignment problem (VAP). The above is however contrary to the report by [3] that GA is able to give competitive results in comparison to DPSO in benchmark problems. The evidence is as demonstrated in Table 3 where the average of fitness values are 0.87 and 0.86 for December, 2006 and January, 2007, respectively.

Although GA applied the same solution representation as DPSO, GA theoretically has difficulty to obtain good solution. The crossover process has shown that the discrete particle position is not updated to a small range, and depends on the crossover point with changing range of discrete values. This can be illustrated in the following example, in an incident where gene is supposedly to be 6, where 6 vehicles are expected to accommodate 40 people at the potential flooded area. During a crossover, this value may change to 2. With 2 vehicles, the new gene value will not be able to accommodate this people. However, this problem can be improvised with a hybrid of GA and DPSO to improve the results.

6 Conclusion and Future Works

This paper introduces a novel DSPO in solving the VAP. With the adoption of a min-max approach the DPSO provides an optimal solution for the VAP. Assignment of people at the potential flooded areas can be made possible using this approach.

The findings show that the novel DSPO gives better solution compared to DPSO with average capacity [6] and GA. In general the proposed DPSO has shown significant performance while GA yields sub optimal solution. Nevertheless, CPU processing time needs to be improved to obtain a much faster results. This could be made possible with the introduction of velocity clamping. Different parameter values should be considered in future research, and to be experimented on a large evacuation datasets.

Acknowledgements. We gratefully acknowledge the help of Prof. Engelbreght from the University of Pretoria. This study has been made possible by the support of the Ministry of Science and Technology Malaysia through the Science fund and University Technology MARA.

References

[1] Omar, M.: Personal Communication, National Security Council, Malaysia (2007)
[2] Guner, A.R., Sevkli, M.: A Discrete Particle Swarm Optimization Algorithm for Uncapacitated Facility Location Problem. J. Artificial Evolution and Applications, 1–9 (2008)
[3] Gong, T., Tuson, A.L.: Particle Swarm Optimization For Quadratic Assignment Problems-A Forma Analysis Approach. J. Computational Intelligence Research 4(2), 177–185 (2008)
[4] Vargas-Villamil, F.D.: Vehicle Routing and Dispatching for Emergency Personnel Evacuation from Off-shore Oil Platforms. In: American Control Conference, pp. 4500–4505. IEEE Press, Los Alamitos (2006)
[5] Yusoff, M., Ariffin, A., Mohamed, A.: Optimization Approaches for Macroscopic Emergency Evacuation Planning: A Survey. In: International Symposium on Information Technology, vol. 3, pp. 1–7. IEEE Computer Society, Malaysia (2008)
[6] Yusoff, M., Ariffin, A., Mohamed, A.: An Improved Discrete Particle Swarm Optimization in Evacuation Planning. In: International Conference of Soft Computing and Pattern Recognition. IEEE Computer Society, Malaysia (2009)
[7] Eberhart, R.C., Kennedy, J.: A New Optimizer Using Particle Swarm Theory. In: 6th International Symposium on Micro Machine and Human Science, pp. 39–43 (1995)
[8] Engelbrecht, A.P.: Computational Inteligence: An Introduction. John Wiley & Sons, England (2002)
[9] Tasgetiren, M.F., Suganthan, P.N., Pan, Q.Q.: A Discrete Particle Swarm Optimization Algorithm for the Generalized Traveling Salesman Problem. In: 9th Annual Conference on Genetic and Evolutionary Computation, England, pp. 158–167 (2007)
[10] Zhong, W.L., Zhang, J., Chen, W.N.: A Novel Discrete Particle Swarm Optimization to Solve Traveling Salesman Problem. In: IEEE Congress on Evolutionary Computation, pp. 3283–3287 (2007)
[11] Ling, A., Ke, G., Ming, Z.: Hybrid discrete particle swarm optimization algorithm for capacitated vehicle routing problem. J. of Zhejiang University Science 7(004), 607–614 (2006)
[12] Veeramachaneni, K., Osadciw, L., Kamath, G.: Probabilistically Driven Particle Swarms for Optimization of Multi Valued Discrete Problems: Design and Analysis. In: IEEE Swarm Intelligence Symposium, pp. 141–149 (2007)
[13] AlHajri, M.F., AlRashidi, M.R., El-Hawary, M.E.: A Novel Discrete Particle Swarm Optimization Algorithm for Optimal Capacitor Placement and Sizing. In: Canadian Conference on Electrical and Computer Engineering, pp. 1286–1289 (1997)

[14] Al-Kazemi, B., Mohan, C.K.: Discrete Multi-Phase Particle Swarm Optimization. In: Information Processing with Evolutionary Algorithms. Advanced Information and Knowledge Processing, pp. 305–327. Springer, Heidelberg (2005)
[15] Kennedy, J., Eberhart, R.C.: A Discrete Binary Version of the Particle Swarm Algorithm. In: IEEE International Conference on Systems, Man, and Cybernetics, Orlando, FL, USA, vol. 5, pp. 4104–4108 (1997)
[16] Eberhart, R.C., Shi, Y.: Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization. Evolutionary Computation. In: Proceedings of the Congress on Evolutionary, vol. 1, pp. 84–88 (2000)
[17] Bui, L.T., Soliman, O., Abbass, H.A.: A Modified Strategy for the Constriction Factor in Particle Swarm Optimization. In: Randall, M., Abbass, H.A., Wiles, J. (eds.) ACAL 2007. LNCS (LNAI), vol. 4828, pp. 333–344. Springer, Heidelberg (2007)
[18] Gao, F., Cui, G., Zhao, Q., Liu, H.: Application of Improved Discrete Particle Swarm Algorithm in Partner Selection of Virtual Enterprise. J. of Computer Science and Network Security 6(3A), 208–212 (2006)
[19] Shi, Y., Eberhart, R.C.: Empirical study of particle swarm optimization. In: Congress on Evolutionary Computation, pp. 1945–1950 (1999)
[20] Shi, Y., Eberhart, R.C.: A Modified Particle Swarm Optimizer. In: IEEE World Congress on Computational Intelligence, Anchorage, AK, pp. 69–73 (1998)

A Computerized Approach of the Knowledge Representation of Digital Evolution Machines in an Artificial World

Istvan Elek

Eotvos Lorand University, Faculty of Computer Science,
Pazmany Peter setany 1/a, 1117 Budapest, Hungary
elek@map.elte.hu
http://team.elte.hu

Abstract. The models of the evolution researchers are focused on the origin and the multiplication of organisms. This paper introduces a possible approach of the evolution of organisms being concerned in the appearance of the intelligence. This process is self developing and adaptive, producing the knowledge graph, which is the result of a lifelong data capturing and acquisition task of an artificial digital organism. This article tries to outline the appearance of the intelligence based on the principles of the creation process of a knowledge graph and its functions. The result is a non linear network of knowledge snippets, consisting of atoms, and their combinations, called contexts. Finally we introduce a startup information system[1], which is the realization of digital evolution machines and their ensemble in the artificial world. This special world is the world wide web.

Keywords: digital evolution, knowledge graph, artificial world.

1 Introduction

The evolution modeling works focus on the multiplication and the competitive exclusion theory, which is one of the most famous low after Darwin in the evolution biology. The competitive exclusion theory ([2, Gause], 1932) comes from Georgii Frantsevich Gause, who was a Russian biologist in the 20th century. The next important low of the evolution process is the irreversibility, which was revealed by [1, Dollo], and expounded by [3, Gould], 1970. The key moment of the evolution is the appearance of the intelligence. The early intelligence was really low level, but it could help to survive the fluctuating environment. The changing Earth and its climate were the most remarkable challenge of the ancient organisms.

Many papers [7, Adami], 2002, [8, Adami], 2003, [9, Ofria at al], 2004, Ostrowski at al [11], 2007, dealing with artificial organisms emphasized the importance of the complexity. Not only life shows serious complexity. There are human

[1] Sponsored by ELTE-Soft, KMOP.

made constructions as well, such as the topology of the Internet, that have increasing complex structures with fractal properties (Barabasi [5, Barabasi], 2002 [4, Barabasi], 2003, [10, Barabasi–Newman], 2006).

The multiplication and the competition are in the center of works mentioned above. The evolution of organisms' body and their intelligence can not be independent. While we are going to model the organisms' evolution, we need to model the development of their intelligence too.

There is no right definition of intelligence of the organisms (Turing's definition can not be translated to DEMs' case). Instead of making useless definitions, it is more promising to set up some essential principles, which adjust the process of collecting and interpreting data from the surrounding world. There are many research papers and books that describe machines which collect data from the surroundings, and they have some kind of remembrance [6, Russel], 2002 (for example the wumpus world). Their ability of interpretation of the environment is restricted. The limits of these constructions are obvious: their intelligence never becomes similar to that of mammals or octopuses.

This paper is about a computerized approach of the digital evolution machines. The principles were established in [12, Elek], 2008 and [13, Elek], 2009.

1.1 Evolution and Knowledge

Paleontology and geology serve many exciting examples of the one way evolution. The time flows in one direction, forward. Life always tries to adapt itself to the circumstances, mainly to the weather (temperature, seasons, climate). If the climate has changed, adaptive organisms also change their right properties, skills and manners. If one million years later the climate changed back, the evolution did not remember the previous stage of the environment. The adaptivity also produces new properties that help to survive the changes. The evolution seems to be recursive. It means the regulation is a kind of recursion, like a feedback. The current stage is the basis of the next step. Evolution never gets back. There are no general physical laws in the background, where the processes can be computed from.

If an organism did not recognize the enemy, the food and the environment, it had died soon or immediately. Consequently, organisms had to collect data from their surroundings (perception) and interpret them (understanding). The remembrance is probably one of the most important components of understanding the world. The unreversable time, i.e. the serial recording of events and data of the environment produces a huge database. It contains everything that happened to an organism in its lifetime. The key of surviving is the interpretation of the surrounding world and the action.

The weather and climate were the most effective factors of the evolution of organisms. Every organism needs to interpret the measured and stored data of the environment if they wanted to survive. This ability required a huge database containing everything that ever happened to the organism. This is the experience based knowledge base.

2 Experience vs. Knowledge

Biologists say there is a kind of self organization among protein molecules in vitro. Even if we assume it was the first step toward a real organism, it is evident there is no huge knowledge database and complicated interpretation logic in it. Protein molecules seem to be inclined to form combinations. Their data collection logic also has to be simple. Consequently a knowledge database can be constructed from simple steps of data collection. The searching algorithm of the knowledge has to be simple.

Let us construct the knowledge graph. It consists of atoms and their connections. Atoms are the nodes, and connections are the edges of knowledge graph. Let us define a context that includes arbitrary atoms of the graph; consequently, the structure of the graph is not predefined. It depends on the atomic connections, which depend on a time series, when events took place in time order one after the other. The general principles declared the equality of atoms and contexts. In other worlds a context can contain simple atoms or other contexts as well.

Let a_{ij} denote the i-th atom in the j-th context which contains N atoms. Let the quantity of the knowledge of a context (k^j) be the sum of the quantity of the knowledge of every atom in it:

$$k^j = \sum_{i=1}^{N} a_i^j \tag{1}$$

The knowledge-base has two basic functions: receiving a question and answering. The key of the problem is to find the path from the questioning node to the answering node in the knowledge base.

Let l_{ij} denote the strength of the connection between a_i and a_j . Let $l_{ij} = l_{ij} + 1$ if the tour produces good result and $l_{ij} = l_{ij} - 1$ if the result is bad. This logic makes good paths stronger and bad ones weaker. Look at the u and f nodes in the knowledge graph. The right distance definition depends on the length of the path along branches, so it is graph tour based.

Let a_i and a_j be two nodes of the knowledge graph where the path includes $m = j - i$ atoms between them. Let d_{ij} be the distance of these two nodes, let the strength of their connection be denoted by l_{ij} , which is the reciprocal of the sum of the strength of the connections between a_i and a_j. The stronger the connection between two nodes, the closer they are.

$$d_{ij} = 1/\left(\frac{\sum_{k=i}^{j-1} l_{k(k+1)}}{m}\right) \tag{2}$$

The goal of the knowledge graph is to answer a question arising from the circumstance. How to find the rigth path from the questioning node to the answering one? The fastest (nearest) is the right path, probably. This logic produces very fast reaction in well known problems and may result fail in unknown cases. The fail means unsuccessful escape, capture or something important for the organism. If it survived the situation, i.e. the reaction was successful, and the path, which produced the success, became stronger. If it did not survive the situation or the result of the action was failed, i.e. the result of the action was unsuccessful, the organism was knocked out or a path in the knowledge graph became weaker.

2.1 Knowledge Graph – Knowledge Matrix

Let **K** denote the knowledge base which consists of n atoms a_i, $a_j \in \mathbf{K}$. Let us name it the knowledge matrix.

$$\mathbf{K} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \tag{3}$$

Some of the atoms are in touch with other atoms in **K**. Let us describe the links of a_i and a_j atoms with l_{ij}, where

$$l_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \text{ and no link between them} \\ u - v & \text{else where } u \text{ successful and } v \text{ unsuccessful} \end{cases}$$

Let us organize the atomic links into a matrix form, name it the link matrix, and denote it by **L**. The elements of the link matrix are l_{ij}, which describes the link of the atomic point pairs. **L** is diagonal ($l_{ii} = 1$) and describes the relationships of the atoms in the knowledge matrix:

$$\mathbf{L} = \begin{pmatrix} 1 & l_{12} & \dots & l_{1n} \\ l_{21} & 1 & \dots & l_{2n} \\ \vdots & \vdots & \ddots & \\ l_{n1} & l_{n2} & \dots & 1 \end{pmatrix} \tag{4}$$

Look at the atomic relationships point by point.

1. Every atom has one link at least.
2. Let **C** be a context. $\mathbf{C} \subseteq \mathbf{K}$, i.e. **C** consist of any atoms of **K**.
3. Any atoms can be the member of any context.
4. Any contexts can be a member of any contexts. In this case, the knowledge matrix (**K**) is a hyper matrix where matrix elements can be matrices.

In summary an atom can be

– a simple atom, which is really elementary and belongs to one context.
– a multi-member atom, which is also elementary, but belongs to more than one context.
– an aggregated atom, which is a kind of simple atom, but its value is a representation of a context. In other words, its value is a determinant or a spur of the knowledge matrix of a certain context.
– a complex atom, which is a context including any kinds of atoms above.

2.2 Constructional Snippets

1. The knowledge base is a continuously increasing database which stores everything that happened to it. This is a one way process. The knowledge base is different from entity to entity.
2. There is no changing or erasing function for existing nodes in the knowledge base.
3. Let the data collection be extensive while a DEM entity perceives the circumstance and stores data. There are some important consequences of the extensive mode:
   (a) Every individual knowledge base is different. It depends on the career of a certain DEM entity. There are as many DEM entities as many kinds of knowledge base exist.
   (b) If some changes happened, which certainly produced the same circumstance in the past, the previously recorded atoms has not been changed, but simply a new atom appended to the knowledge base.
4. Let us have another mode that was named intensive, when there is no perception. It is a ,,meditative" stage when the knowledge base acquires the data came from the extensive stage. In intensive mode the result of the data acquisition may produce new contexts, faster path, more reliable work. This stage is extremely important for learning the circumstance.
5. The feedback is a process, when a DEM entity is informed about the result of its reaction. The result of this process is success or fail. As mentioned previously, the success/fail makes stronger/weaker a certain path in the knowledge base. A certain DEM entity's knowledge base becomes more reliable if the feedback such as rewarding or punishment comes from an external intelligent entity, because its knowledge base can be considered as an included context.
6. In the Earth history there was never only one organism. There were always ensembles. Ensembles make organisms competitive. Competition results different skills i.e. different knowledge bases.
7. Various circumstances cause various experiences for the organisms.

The evolution is a process in time and in space dimensions. Regarding the results of the paleontology it is known, that the evolution is recursive, i.e. an evolution step depends on the previous step only, and it is a one way process. It can not be turned back. This is known as Dollo's law [1] in paleontology. This law was first stated by Dollo in this way: ,,An organism is unable to return, even partially, to a previous stage already realized in the ranks of its ancestors". According to this hypothesis, a structure or organ that has been lost or discarded through the process of evolution will not reappear in that line of organisms.

The development means not only physical but mental changes as well in an organism. Mental means intelligence in this context. The adaptivity comes from its knowledge base. The quality of the knowledge base has to influence the physical properties also. Consequently the physical and mental evolution work collaterally, since organisms live in various climate, in many challenges, having different experiences with different chances with various knowledge but in many, different places at the same time.

3 The Artificial World

If we are willing to create experienced DEM entities there is no wizard unfortunately. There is no a recipe to install them from little pieces. How to construct a DEM entity? Probably, it is impossible to construct only one of it. First we should create an ensemble from many initial DEM entities and leave them to live in their own world.

We have two access points to this problem. The first one is to construct an artificial circumstance where DEM entities live in. The second one is to construct many initial DEM entities with simple perception and interpretation functions. Leave them alone and always watch them. Look at some details:

1. The artificial world (AW) has some essential properties that define the frame of this world.
   (a) Let AW be huge, where circumstances have spatial dependencies. Regarding the size of this world, environmental parameters are obviously different. If we leave DEM entities alone in this world, they will have different experiences and different knowledge bases because of climatic differences.
   (b) If the AW is huge enough, the survival strategies will be different. One of the DEM entities escapes from unfriendly circumstances, but others try to adapt. Different strategies result different knowledge -bases.
2. If there are many DEM entities on the same territories, what is the consequence?
   (a) There are many DEM entities who try to get better strategy in order to be more successful than others. Someone gets advantages but someone gets disadvantages since it fails to answer a certain question.
   (b) Regarding the different DEM entities and unique knowledge-bases, many different strategies can coexist in the AW. Consequently, many different strategies can be successful at the same time. Someone prefers the escape, but someone the competition. Many DEM entities will have many different knowledge-bases.

The question is how to construct the prototype of a DEM entity? Before the construction of the prototype, let us create the artificial world, which will be the space for DEM's life. If AW has been created already many DEM entities should be available to start up in it. Properties and abilities of DEM entities were introduced previously, so the task is to make their software representation. Regarding the quantity of DEMs and the huge sized world with different spatial properties may result many formed DEMs, and these entities have different knowledge bases as it was proved above.

An intelligent DEM entity can exist in ensemble only. There is no lonely intelligence, because it is a collective product, which is appeared and realized in some successful DEM entities.

4 The Computerized Approach

Finally focus on the appearance of the intelligence, and its properties in a digital representation. Digital evolution machines and the Internet, which is an artificial world, where they ,,live in", are in interaction. The purpose of the work is to model this interaction. The task is to store everything sequentially that happened to DEMs, and to construct the knowledge graph, where their nodes are the pointers for sequentially stored events. Briefly the system works as follows:

1. There are a lot of DEMs in the Internet. An individual DEM starts his career in a starting URL.
2. This DEM investigates the certain URL, what kind of files are linked to this. Some tags point to another URL-s (< a ref = http://...>) in this file. Some of the pointed files are considered as resources which are needed to DEM's life (for instance *doc, pdf, txt, jpg, gif* files can be considered resources with various energy content, but *exe, bat* files are considered poison, and so on).
3. A successful step, when a DEM reached a resource, produces energy input for a certain DEM, makes path stronger. If the reached node (file) is neutral or contains poison, the path becomes weaker.
4. If a DEM has visited a resource, the node (file) runs out of its energy for a short period (it needs time for recover itself), so DEM has to move to another URL for further resources (for instances, it follows an < a ref = http://...> html tag).
5. go to 2

Last but not least, in order to find the rigth URL for the next move, a DEM needs an '*adviser*', which is one of the most remarkable component of the system. '*Adviser*' uses the knowledge graph, and give the most promising URL for a DEM. Summarily sequentially stored events and '*adviser*' functionality figure out a certain DEM's intelligence.

For the technical implementation, there is a server, where a MySql database management system stores sequentially the every day events, (the table name is '*logbook*'). The table '*dems_register*', stores data for the identification of individual DEMs. This table contains the resource requirements also of a certain DEM. The visited web nodes are stored in the table '*artiworld_nodes*'. A related table, named '*urls_content*' stores the link of an URL (resource files, and potential URLs for further moves). The DEMs' web site is *http://dem.elte.hu.*

DEMs will affect the artificial world, and AW influences DEMs life. This interaction is the target of our research. The system is working, but not finished yet, of course, because this is a changing database.

References

1. Dollo, L.: http://en.wikipedia.org/wiki/Dollo's_law
2. Gause, G.F.: Experimental studies on the struggle for existence. Journal of Experimental Biology 9, 389–402 (1932)

3. Gould, S.J.: Dollo on Dollo's law: irreversibility and the status of evolutionary laws. Journal of the History of Biology (Netherlands) 3, 189–212 (1970)
4. Albert, Barabasi: Statistical mechanics of complex networks. Reviews of Modern Physics 74 (January 2002)
5. Barabasi, Albert, Jeong: Mean-field theory for scale-free random networks. Preprint submitted to Elsevier Preprint, July 5 (2002)
6. Russel, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice-Hall, Englewood Cliffs (2002)
7. Adami, C.: Ab Initio of Ecosystems with Artificial Life, arXiv:physics0209081 v1 (September 22, 2002)
8. Adami, C.: Sequence Complexity in Darwinian Evolution, Complexity. Wiley Periodicals 8(2) (2003)
9. Chow, S.S., Wilke, C.O., Ofria, C., Lenski, R.E., Adami, C.: Adaptive radiation from resource competition in digital organisms. Science (New York, N.Y.) 305(5680), 84–86 (2004)
10. Newman, M., Barabasi, A.L., Watts, D.L.: The structure and Dynamics of Networks. Princeton University Press, Princeton (2006)
11. Ostrowski, E.A., Ofria, C., Lenski, R.E.: Ecological specialization and adaptive decay in digital organisms. The American Naturalist. 169(1), E1-20 (2007)
12. Elek, I.: Principles of Digital Evolution Machines. In: International Conference of Artificial Intelligence and Pattern Recognition, Orlando, FL (2008)
13. Elek, I.: Evolutional Aspects of the Construction of Adaptive Knowledge Base. In: The Sixth International Symposium on Neural Networks (ISNN 2009), Wuhan, China (2009)

An Improved Thermodynamics Evolutionary Algorithm Based on the Minimal Free Energy

Fahong Yu[*], Yuanxiang Li, and Weiqin Ying

State Key Lab of Software Engineering, Wuhan University, Wuhan 430072, China
fhyu520@126.com

Abstract. In this paper, an improved thermodynamics evolutionary algorithm (ITEA) is proposed. The purpose of the new algorithm is to systematically harmonize the conflict between selective pressure and population diversity while searching for the optimal solutions. ITEA conforms to the principle of minimal free energy in simulating the competitive mechanism between energy and entropy in annealing process, in which population diversity is measured by similarity entropy and the minimum free energy is simulated with an efficient and effective competition by free energy component. Through solving some typical numerical optimization problems, satisfactory results were achieved, which showed that ITEA was a preferable algorithm to avoid the premature convergence effectively and reduce the cost in search to some extent.

Keywords: evolutionary algorithm, free energy component, similarity entropy.

1 Introduction

The evolutionary algorithm is wildly used in searching, optimization, and machine learning [1], in which connotative parallel character and efficiency of using global information are two prominent characteristics. However, it often suffers from a contradiction that premature convergence and low rate in searching, which is a tunable contradiction between exploration and exploitation in evolutionary algorithm.

There are a wide variety of improvements in evolutionary algorithms focused on harmonize the contradiction such as scaling the fitness, sharing the fitness, and driving all individuals moving as the most possible [2] etc. Kirkpatric has proposed another general optimization algorithm called the simulated annealing (SA) [3] which controlled the process of searching systematically by the cooling temperature and the Metropolis rule. NDEA [5] is an algorithm based on information entropy defined in search space *VSP* divided to some grids while it can not be used to real-number optimization problem with high-dimension. Mori and his fellows have proposed the thermodynamics genetic algorithm (TDGA) [6] combined SA and GA. They introduced a greedy thermodynamics selection rule based on the principle of minimal free energy in TDGA and attempted control to control population diversity systematically through tuning to temperature and entropy in annealing. Although TDGA was effective to

[*] Corresponding author.

solve some problems, the satisfied performance and extremely high cost in computation were still unstable. To improve the stability and decrease the computational cost of TDGA, an improved thermodynamics evolutionary algorithm (ITEA) is proposed.

2 Description of ITEA

The second law of thermodynamics describes an isolated system that is always evolving towards the direction of increasing entropy and its formula can be described in algebra as the express $dS \geq 0$, which means that the isolated system will evolve towards the stable status with the continually increasing of the entropy.

There is a useful connection between annealing in solids and convergence in GA. The population and the individuals in GA may be regarded as a thermodynamics system and particles, and the energy, entropy and temperature is served as the fitness, measurement of population diversity and controllable parameter respectively. This analogy provides an approach for ITEA to simulate the competitive mechanism between energy and entropy in annealing to systematically harmonize the conflict between selective pressure and population diversity in GA.

2.1 Similarity Entropy and Measurement of Population Diversity

It is a critical part how to measure population diversity when introducing the competitive mechanism into GA. Therefore, in order to measure population diversity more effectively, similarity entropy defined in search space is introduced.

Definition 1: Let X_o be the best individual in population $P=\{X_1, X_2, \dots, X_n\} \subseteq H^D$, $X_i = (x_{i1}, x_{i2}, \dots, x_{iD})^T$, and set $X_i = X_i / \|X_i\|$ when $\|X_i\| \neq 0$, The directional similarity degree be defined as the express $S_i = 1 - X_i^T X_o \in [0,1]$.

Definition 2: Let $R = \max\{d_i | i=1, 2, \dots, n\}$ be the maximal distance between the X_o and X_i in population $P=\{X_1, X_2, \dots, X_n\} \subseteq H^D$ and $l = \delta + \lceil D(I+D)/\sqrt{popsize} \rceil$, the variable I be the longest span of boundary in decision-making space and δ be a parameter, set a partition $\pi = \{L_j | 1 \leq j \leq k \Lambda j \in N\}$ on field $[0,1]$, $kl = 1$ and $L_1 \Lambda L_2 \dots \Lambda L_k = \Phi$. The individual X_i belong to the partition L_j when $(d_i/R + S_i)/2 \in [l(j-1),\ lj)$. If the number that the individuals in population P belonged to L_j is n_j, the similarity entropy be defined as $S(P)$.

$$S(P) = -\sum_{j=1}^{k} \frac{n_j}{n} \log_k \frac{n_j}{n} \cdot \tag{1}$$

The higher degree the individuals in population P congregate, the easier convergence the optimization problem is. According to definition 1 and definition 2, the more near to zero S_i is, the more similar the individual X_i and X_o in their directions and $S_i = 0$ when X_i and X_o is the same completely in direction. In addition, $d_i = 0$ when X_i and X_o is in the same position. So the smaller the value of the express $d_i/2R + S_i/2$ is, the individual X_i and X_o is more similar in direction and position and the more high the similar degree of that two individuals. Since the variable $n_j (i=1, 2, \dots, k)$ is the number belonged to partition L_j for those individual in population P, the sum $\sum_{i=1}^{k} n_j$ is n. Similarly, since the probability p_j can be get by the formula $n_j/n.$, $(j=1,2,...,k)$, the sum $\sum_{j=1}^{k} p_j$ is 1. Any change for the probability to be close to the same value will contribute the similarity entropy to increase. If all value of the probability p_j is i/k, $(j=1,2,...,k)$, the similarity entropy $S(P)$ is 1 that is its maximum. If only one value of the probability is 1 and the others is 0, the similarity entropy $S(P)$ is 0 that is it's the minimum.

By similarity entropy, the process in calculating is simplified greatly and the diversity of population can be measured effectively. The calculation of similarity entropy on similarity degree and distance has avoided the complicated calculation on every gene.

2.2 Minimization of Free Energy at Each Temperature

In order to apply the principle of minimal free energy in thermodynamics to ITEA, the method of free energy of population is presented as follow.

Definition 3: For $P_t=\{X_1,X_2,...,X_n\} \in H^n$, the energy of population P_t be defined as $E(P_t)$.

$$E(P_t) = \frac{\mu}{n}\sum_{i=1}^{n} f(X_i) \cdot \tag{2}$$

The value of variable μ will be set -1 if the optimized problem is minimum *max (f(X))* or the value of variable μ will be set -1.

Definition 4: For $P_t=\{X_1,X_2,...,X_n\} \in H^n$, $\pi=\{L_j|1\le j\le k \Lambda j \in N\}$, $F(\pi, T, P_t)$ be called the free energy of population P_t at temperature T for partition π.

$$F(\pi,T,P_t)=E(P_t)-TS(\pi,P_t) \tag{3}$$

The aim of algorithm is to minimize the free energy of population at each temperature during the process of evolution so as to drive the thermodynamic system towards equilibrium state.

In order to minimize the free energy rapidly at each temperature, a thermodynamics selection rule to descend the free energy of the next generation most steeply should be designed. Its mission is to select n individuals from n parent individuals and m offspring individuals as the next generation with the minimum free energy. However, It is infeasible to exactly minimize the free energy for each generation because of the extremely high complexity $O(nC_{n+m}^{n})$. Hence, TDGA uses a greedy thermodynamics selection (GTS) rule with the complexity $O(mn^2)$. But its reliability can't be guaranteed. In this section, a competition of free energy components (CFC) by assigning the free energy of the population to its individuals will be proposed.

Definition 5: Let $P=\{X_1,X_2,...,X_n\} \in H^D$, $\pi=\{L_j|1\le j\le k \Lambda j \in N\}$. For an individual X_i belong to the partition L_d, the free energy of the individual X_i in P at temperature T for π is defined as $F_c(\pi, T, P, X_i)$.

$$F_c(\pi, T, P, X_i) = \mu f(X_i) + T\log_k(n_d / n) \qquad (4)$$

It is clear that the time complexity is lowed to *O((n+m)m)*. The procedure of competition by free energy component *CFC(π, T, P_t)* is as follows:

1. Produce an interim population P_t^{n+m} of size *n+m* by appending the offspring population O_t to the parent population P_t^n, calculate a partition according to definition 1 and definition 2 and count particles in each partition.
2. Calculate the free energy $F_c(\pi, T, P_t^{n+m}, X_i)$ according to the formula (4).
3. Select individuals with the minimum free energy to serve as a next population P_{t+1}^n from population P_t^{n+m} .

2.3 Procedure of ITEA

Procedure of ITEA is as follows:

```
1. Initial parameters. i.e., population size n, $T_0$, MAXGENS, offspring population size
   m, randomly generate $P_0$;
2. Calculate the number of field partition $k$;
3. t=0;
4. while Termination_test($P_t$) < > true do
5.   Generate offspring population $O_t$ with $m$ individuals by crossover and mutation;
6.   $P_{t+1}$= CFC(π, T, $P_t$);
7.   t++, $T_t=T_0/(1+t/100)$;
8. endwhile
9. Output the final results.
```

3 Experiments and Results Analysis

The significance of a new optimization strategy is applied to solve practical problems depends effectively. Here chose two functions as test examples, which are often used for optimization of the test by many scholars at home and abroad.

Schaffer:

$$\max f(x) = 0.5 - \frac{\left(\sin^2\sqrt{x_1^2+x_2^2} - 0.5\right)}{\left[1+0.001\left(x_1^2+x_2^2\right)\right]^2} \qquad -100 \le x \le 100$$

Schaffer [10] function is a strong oscillatory behavior function with multiple local maximization and a global maximization is *maxf(x)=1* when *X=(0, 0)*.

Needle in haystack (NiH):

$$\max f(x) = (x_1^2+x_2^2)^2 + \left(\frac{3.0}{0.05+(x_1^2+x_2^2)}\right)^2 \qquad -5.12 \le x \le 5.12$$

NiH is a typical kind of strong deceptive problem with the global maximization is *maxf(x)=3600* in central point and multiple local maximization near the point *X=(±5.12,±5.12)*.

In this experiment, the individual was expressed in vector $X = (x_1, x_2, \ldots, x_D)$ with real-number encode and four algorithms included simple genetic algorithm (SGA), niche genetic algorithms (NGA), greedy thermodynamic evolutionary algorithm (GTEA) and ITEA were applied. In order to compare rationally, a uniform circumstance were adopted in every algorithm, which include the same machine, the same arithmetic crossover operator with probability $P_c=0.8$, mutation operator with probability $P_m=0.02$, population size $n=30$, were utilized. The other parameters were set as follow: the niche radius on NGA $r=4$, the offspring population size $m=4$ and the length of Markov under the temperature T_k is L_k=2000 for GTEA and ITEA. The worst results, the mean results, the best results and first hitting time respectively for 30 times were gotten through four algorithms carried out on functions Schaffer and NiH. The compare of statistic results are given as table 1.

Table 1. Statistic results on Schaffer and NiH for four algorithms

Instance	Algorithm	Worst	Mean	Best	First hitting time
Schaffer	SGA	0.962776005	0.987615296	1.0000000000	>4.464
	NGA	0.999999821	0.999999961	1.0000000000	=5.626
	GTEA	0.990284144	0.996599451	1.0000000000	>665.248
	ITEA	1.000000000	1.000000000	1.0000000000	=1.491
NiH	SGA	2748.782226562	3344.634667969	3600.0000000000	>1.415
	NGA	2748.782226562	3004.147558594	3600.0000000000	>17.298
	GTEA	2748.782226562	3131.830224609	3600.0000000000	>696.211
	ITEA	3600.000000000	3600.000000000	3600.0000000000	=0.807

There were some statistic results included the worst results, the mean results, the best results and first hitting time respectively for 30 times in table I. The first hitting time implies that the first time to achieve $f(x)\geq 0.999$ for Schaffer and $f(x)\geq 3599.9$ for NiH. According to the count s for every individual being taken part to evaluate respectively before the best solution were get at the first time on four algorithms carried on Schaffer and NiH every time, the performance can be evaluated on four ranks contained that excellent ($s\leq 8.0\times 10^4$), good ($8.0\times 10^4<s\leq 3.2\times 10^5$), weak ($3.2\times 10^5<s\leq 1.28\times 10^6$) and bad ($s>1.28\times 10^6$) and the distribution of algorithm performance on Schaffer and NiH were described as fig.1, in which the vertical axes represent the performance ratio and the level axes represent the ranks. The average convergence curves on Schaffer and NiH were described as fig.2.

In addition, the effect caused by offspring population size on ITEA was tested as table 2 and fig.3.

Viewing from results of the experiment, some conclusion can be drawn that ITEA could obtain very good accuracy, stability and faster rate to converge comparing other three algorithms for its strong search capabilities in every generation, reliability to maintain the right search direction and ability to avoid premature convergence effectively. So it is regarded as a very important algorithm for some engineering optimization problems.

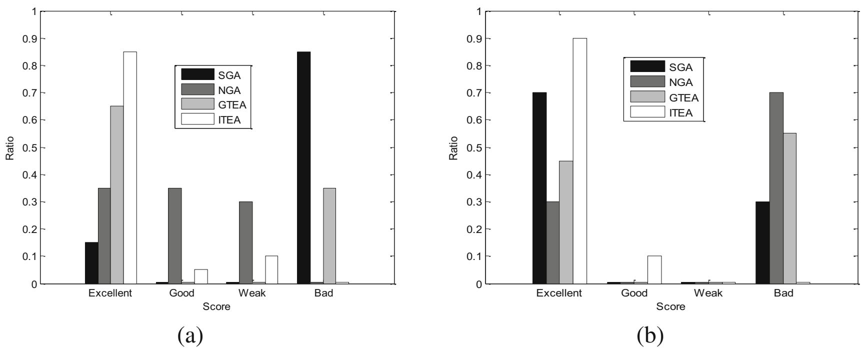


Fig. 1. Distribution of algorithm performance on Schaffer (a) and NiH (b)

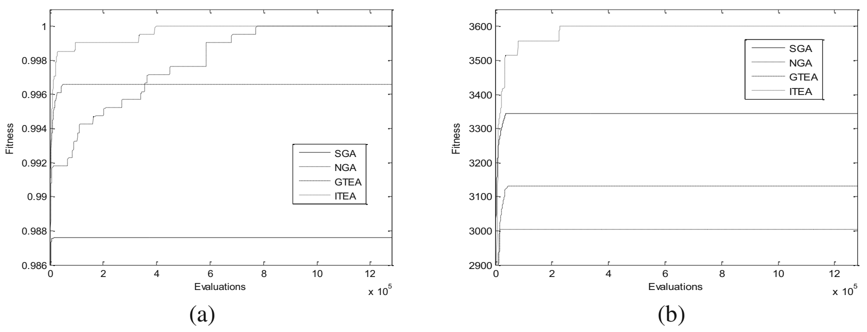


Fig. 2. Convergence curves on Schaffer (a) and NiH (b)

Table 2. Statistic results on Schaffer and NiH for ITEA with various offspring population sizes

Instance	size	Worst	Mean	Best	First hitting time
Schaffer	4	1.0000000000	1.0000000000	1.0000000000	=1.491
	8	0.9999999404	0.9999999970	1.0000000000	=0.202
	16	0.9626941681	0.9908478171	1.0000000000	>4.445
NiH	4	3600.0000000000	3600.0000000000	3600.0000000000	=0.807
	8	2748.7822265625	3514.8782226563	3600.0000000000	>1.597
	16	2748.7822265625	2958.2632568359	3600.0000000000	>4.208

The main reason why the outstanding performance could be archived for ITEA lies in its adaptive capability to maintain the dynamic balance between “selective pressure” and “species diversity” by adjusting the temperature. Generally, in competition between “selective pressure” and “species diversity”, the entropy occupies the predominance for high temperature in the early stages of evolution which lows level of aggregation and energy plays main function for low temperature in the latter evolution which allows a higher degree of aggregation contributing to converge to the optimization solution set.

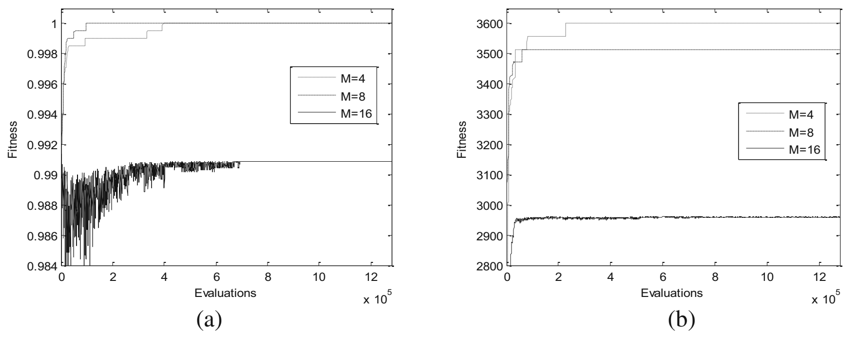


Fig. 3. Convergence curves on Schaffer (a) and NiH (b) for ITEA with various offspring population sizes

4 Conclusion

In order to systematically harmonize the conflict between selective pressure and population diversity, an improvement of thermodynamics evolutionary algorithm was proposed. In ITEA, the definition of similarity entropy and the application of free energy component thermodynamic selection strategy were given. ITEA simulates the competitive mechanism between energy and entropy in annealing to harmonize the conflict between selective pressure and population diversity effectively by temperature and reduced the complexity of the algorithm to some extent. By solving some typical numerical optimization problems, the efficiency and good performance of the ITEA were achieved. Therefore, ITEA avoids premature convergence successfully and obtains faster rate, good accuracy and stability.

Acknowledgment. This work was supported by the National Natural Science Foundation of China under Grant No. 60773009.

References

1. Pan, Z.J., Kang, L.S., Chen, Y.P.: Evolutionary computation. Tsinghua Uinv. Press, Beijing (1998) (in Chinese)
2. Michalewicz, Z., Fogel, D.: How to Solve It: Modern Heuristics. Springer, Berlin (2000)
3. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science 220, 671–680 (1983)
4. Vavak, F., Fogarty, T.C.: A comparative study of steady state and generational genetic algorithms for use in nonstationary environments. In: Fogarty, T. C. (ed.) AISB workshop on Evolutionary Computing, pp. 297–304. Springer, Brighton (1996)
5. Hu, T., Li, Y.X., Ding, W.: A new dynamical evolutionary algorithm based on the principle of minimal free energy. In: Kang, L.S., Cai, Z.H., Yan, X.S. (eds.) Progress in Intelligence Computation and Applications, pp. 749–754. China University of Geosciences, Wuhan (2005)
6. Mori, N., Yoshida, J., Tamaki, H., Kita, H., Nishikawa, Y.: A thermodynamic selection rule for the genetic algorithm. In: Fogel, D.B. (ed.) Proc. of IEEE Conf. on Evolutionary Computation, pp. 188–192. IEEE Press, New Jersey (1995)

7. Ying, W.Q., Li, Y.X.: Improving the Computational Efficiency of Thermodynamical Genetic Algorithms. Journal of Software, 1613–1622 (2008)
8. Su, X.H., Yang, B., Wang, Y.D.: A Genetic Algorithm Based on Evolutionarily Stable Strategy. Journal of Software 14(11), 1863–1868 (2003) (in Chinese)
9. Abs da Cruz, A.V., Vellasco, M.M.B.R., Pacheco, M.A.C.: Quantum-inspired evolutionary algorithm for numerical optimization. In: Yen, G.G., et al. (eds.) Proc. of IEEE Congress on Evolutionary Computation, pp. 2630–2637. IEEE Press, New Jersey (2006)

A Hybrid Evolutionary Algorithm Based on Alopex and Estimation of Distribution Algorithm and Its Application for Optimization[*]

Shaojun Li , Fei Li, and Zhenzhen Mei

Institute of Automation, East China University of Science and Technology, Shanghai, 200237, P.R. China
lishaojun@ecust.edu.cn

Abstract. Alopex is a correlation-based algorithm, which shares characteristics of both gradient descent approach and simulated annealing. It has been successfully applied to continuous and combinatorial optimization problems for years. Estimation of Distribution Algorithms (EDAs) is a class of novel evolutionary algorithms (EAs) proposed in recent years. Compared with the traditional EAs, it possesses unique evolutionary characteristics. In this paper, a hybrid evolutionary algorithm (EDA-Alopex) is proposed, which integrates the merits of both Alopex and EDA, and obtains more evolutionary information than these two approaches. The new algorithm is tested with several benchmark functions; numerical case study results demonstrate that EDA-Alopex outperforms both EDA and AEA, especially for the complex multi-modal functions. Finally, the proposed algorithm is investigated on high-dimensional and multi-peaks benchmark functions, and it also achieves satisfactory results.

Keywords: Alopex, Estimation of distribution algorithm, Meta-heuristics, Evolutionary algorithm.

1 Introduction

Alopex (Harth and Tzanakou, 1974) [1-3] was first proposed to solve combinatorial optimization and pattern match problems. Inspired by the Alopex algorithm, a new evolutionary algorithm (EA) was proposed by the author in [2], called as Apolex-based evolutionary algorithm (AEA). In the AEA, the correlation calculation and Boltzmann-type probability in Alopex algorithm are introduced to evolve swarms, and an adaptive annealing schedule is adopted instead of the one used in the original Alopex algorithm. These are the two major differences between the AEA and the Alopex. Moreover, during the iterative progress of the AEA, two populations need to be generated to calculate the correlation which is used for heuristic information. Therefore, to some extent, the information contained in the two populations determines the convergence

[*] This work was supported by Chinese National Programs for High Technology Research and Development(Grant No.2007AA04Z171) and National Natural Science Foundation of China(Grant No.20976048).

speed and solution quality. Comprehensive comparisons had been carried out between the AEA and genetic algorithm (GA) [4], particle swarm optimization (PSO) [5], as well as differential evolution (DE) [6]. The results show that the performance of AEA is superior to these EAs[2].

Estimation of Distribution Algorithm (EDA) [7-9] was proposed by Mühlenbein and Paaß (1996) as an extended version of GA, which is a novel kind of EAs. The solutions of EDA are generated by sampling from probability model derived from the promising individual set instead of crossover, mutation or differential operations implemented in the traditional EAs [8]. In EDAs, the global probability information is used as heuristic information.

Inspired by the search information extracted by both EDA and AEA is fundamentally distinct, in this paper, EDA is embedded into AEA to form a hybrid algorithm [10, 11] EDA-Alopex, for the purpose of improving the population diversity and acquiring more heuristic information from searching process. In EDA-Alopex, EDA is used to generate two populations, which are then passed to AEA for correlation calculation, providing more information for the evolution of AEA and in favor of an evolving towards to the promising domain. Due to the EDA is computation inexpensive, there is no significant increase of runtime of AEA, however, a relatively substantial improvement of performance is achieved, especially for the convergence rate.

The outline of the paper is organized as follows. In section 2, the basic principles of AEA and EDA will be illustrated for completeness. In section 3, the detailed steps of the EDA-Alopex are listed. In section 4, ten benchmark functions and different test methods are used to investigate the performance of the hybrid algorithm. Conclusions are given in the last section.

2 Principles of AEA and EDA

AEA is a new swarm intelligence evolutionary algorithm based on Alopex algorithm. With regard to the basic Alopex algorithm [3], the temperature 'T' for the first annealing cycle is preset subjectively. It is a great blindness of setting an initial temperature empirically, especially for the optimization problem, of which no priori domain knowledge is obtained. In AEA, adaptive annealing strategy is adopted. There is no need to set the initial temperature. From the beginning, the process of annealing is self-regulation, which not only improves the performance of the algorithm but also makes the algorithm simpler. The schedule of annealing applied in AEA is described by the following Eq.5.

The main process of the population-based AEA is expressed as follows [2, 3]: consider a minimization for the functions $F(x_1, x_2 \cdots x_N)$, where, $x_1, x_2 \cdots x_N$ are variables. Suppose there are two populations $X1, X2$, the population size is L and variable dimension is N. Randomly select two individuals from $X1$ and $X2$, which represent two state-vector at moment $t-1$ and $t-2$, respectively. And then the product $C = \Delta x \Delta F$ between vector difference Δx and corresponding function value ΔF is calculated, which describes the correlation between different individuals in the population. And then the probability is used to determine walk direction of each variable of individual, aiming at making each variable walk towards to the direction that can reduce the function value and enhancing the climbing ability. Then each individual in population $X1$ is renewed according to Eq.1 to form new individuals. Determine whether the function value of the new individuals get improvement, the lesser one is retained as the individual for the next generation. The formula is as follows:

$$X1^k(i,j)' = X1^k(i,j) + \left(X2^k(i,j) - X1^k(i,j)\right) \cdot \delta(i,j) \cdot rand(0,1) \tag{1}$$

$$\delta(i,j) = \begin{cases} 1 & with\ probability\ p^k(i,j) \\ -1 & with\ probability\ 1 - p^k(i,j) \end{cases} \tag{2}$$

$$p^k(i,j) = \frac{1}{1 + \exp\left(\frac{\pm C^k(i,j)}{T^k}\right)} \tag{3}$$

$$C^k(i,j) = \left[X1^k(i,j) - X2^k(i,j)\right] \times \left[F\left(X1^k(i,:)\right) - F\left(X2^k(i,:)\right)\right] \tag{4}$$

$$T^k = \frac{1}{LN} \sum_{i=1}^{L} \sum_{j=1}^{N} C^k(i,j) \tag{5}$$

Where, $X1^k(i,j)$ denotes the j-th variable of i-th individual in population $X1$ at the k-th iteration. $rand(0,1)$ is a random number which subjects to uniform distribution between 0 and 1. $\delta(i,j)$ is walk direction of j-th variable of i-th individual. The selection of positive and negative sign in the Eq.3 depends on the purpose of optimization. The positive or negative sign minimize or maximize the object to be optimized, respectively. In Eq.4, $F\left(X1^k(i,:)\right)$ denotes function value of i-th individual in population $X1$. In Eq.5, T^k is the annealing temperature during the k-th iteration.

According to the dependence between variables, there are several kinds of EDAs have been proposed. In this paper, UMDAc [12] (Univariate Marginal Distribution Algorithm for Continuous Domains) is employed, in which interdependence between variables are not considered. In UMDAc, the n-dimensional joint probability distribution is factorized as a product of n independent normal distributions [13]. Suppose the k-th iteration, the joint density function of n-dimensional variable is computed as follows:

$$f^k(x) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma_i^k}} \exp\left(\frac{1}{2}\left(\frac{x_i^k - \mu_i^k}{\sigma_i^k}\right)^2\right) \tag{6}$$

In Eq.6, the mean μ_i^k and the standard deviation σ_i^k need to be estimated. In this chapter, maximum likelihood estimation is adopted for the estimation of parameters.

Overall speaking, process of EDA can be divided into four parts: selection, modeling, sampling and replacement. Firstly, best individuals are selected from the initial population to form a promising individual set. There are two problems for selection need to be determined: Selection ratio and selection method. The number of selected individuals divided by total population denoted the selection ratio ω. As for selection method, truncation selection [14] or tournament selection [12] can be applied. Secondly, the probability can be established by estimating the mean μ and the standard deviation σ. The third step is sampling, which guarantees the descendents sampled from the probability model to scatter on the most favorable search area. Finally, whether the new population replaces the old one totally or not is determined by the scheme of replacement in which the replacement ratio ν within the range of 0 to 1 is put up.

3 EDA-Alopex Algorithm

EDA and AEA evolve in different ways. In EDA, there is no operation of individuals and the probability model is used to guide the evolution of population, which is the largest distinction compared with the traditional swarm intelligent algorithm. In EDA, evolutionary information is obtained through the probability model which characterizes the global statistical information [13]. On the contrary, the most parts of information acquired by AEA are local correlation information [3]. On the foundation of AEA and EDA, a hybrid evolutionary algorithm called EDA-Alopex is proposed in this chapter. In the new algorithm, EDA is embedded into AEA to expect to provide more search information for the evolution of AEA and enhance the diversity of the population.

From the main process of AEA described in section 2, we can know that two populations $X1$ and $X2$ is produced for the computation of the correlation C, which is the main driving force for the evolution of population. Therefore, the more evolution information contained in the two populations, the greater probability owned to find the optimal solution. Inspired by the unique evolutionary model adopted by EDA and only need to allocate less time for the implementation of EDA, EDA is embedded into AEA to generate the two populations, which brings a richer population and improves the convergence speed of AEA.

The step size of original AEA expressed in section 2 is fixed, but varied step length changing with search process is more reasonable in the perspective of evolution. In order to complete AEA, a varied step size changing with iterations is applied in this paper. In the early stage of search process, a relatively lager step size promotes the swarm skim over an extensive search area. As for the later period of search process, due to the difference between individuals is little; the population is easier to be trapped at a local optimum, which can be alleviated through using of a relatively larger step size. The scheme of step size varying with iterations is given as follows:

$$X1^k(i,j)' = X1^k(i,j) + \left(X2^k(i,j) - X1^k(i,j)\right)\cdot\delta(i,j)\cdot rand(0,1)\times\gamma^k \tag{7}$$

$$\gamma^k = \begin{cases} \alpha 0\times\exp\left(\frac{k}{M}\log\left(\frac{\alpha 1}{\alpha 0}\right)\right) & k \le M/2 \\ \beta 0\times\exp\left(\frac{2k}{M}\log\left(\frac{\beta 1}{\beta 0}\right)\right) & k > M/2 \end{cases} \tag{8}$$

Where: γ^k is an shrinkage factor, decreasing or increasing exponentially with the iterations, M denotes the total number of iterations, $\alpha 0$, $\alpha 1$, $\beta 0$ and $\beta 1$ are constant.

Here, the iterative steps for EDA-Alopex are described as follows: also a minimization for the functions $F(x_1, x_2 \cdots x_N)$ is used as an example for explanation.

Step 1. Initialize the parameters for EDA-Alopex. A given maximum generation or optimization target for objective function is preset, initialize population randomly and calculate the objective function value of every individual in population, set the number of iteration $k := 0$

Step2. According to the objective function value, select the promising solutions to form the promising solutions set. Truncation selection is applied in this chapter.

Step3. Based on the promising solutions set, compute the mean μ and the standard deviation σ of every variable, and then establish a probability model described by Eq.6.

Step4. Generate new solutions by sampling from the constructed probabilistic model; in according with the replace ratio ν, replace the old population totally or in part to form a new population $X1$.

Step5. Re-arrange the order of the individuals in population $X1$ to form a second population $X2$, according to Eq. 3-5 given above, the probability p, the correlation C and annealing temperature T are calculated respectively.

Step6. Each individual in population $X1$ is renewed according to Eq.7 expressed above to form an intermediate population $X1'$, value of objective function $F(X1')$ for population $X1'$ and the corresponding function value $F(X1)$ for population $X1$ is compared, the lesser one is retained as the individual for the next generation.

Step7. If the given termination criterion is met, yes, go to step 8, else $k = k+1$, go to step2.

Step8. Output the best individual and the best value.

As can be seen from the above steps, from step 1 to step 4, an EDA is implemented. The step 5 and 6 are steps for AEA. Each generation, the excellent solutions of population are used to construct probability model; then the population retaining the promising individuals of previous generation and the best individuals sampled from the probability model are passed to AEA for the correlation calculation. Therefore, compared with the single EDA or AEA, EDA-Alopex obtains a more diversity population.

In EDA-Alopex, for the existence of replacement ratio ν, part of population comes from the offspring generated by EDA and the other part of population stems from descendents yielded by AEA for every generation. Therefore, two different evolutionary strategies are applied to drive the population moving towards the optimal solution. Actually, some individuals in population evolve in EDA way, while the other individuals evolve in AEA way. So offspring generated by EDA-Alopex include not only probability information which describes evolution from the macro level but also an Alopex heuristic information which is similar to gradient descent. For this reason, the hybrid algorithm can integrate the advantage of both AEA and EDA, guiding the population towards a more promising search space.

4 Simulation Results and Analysis

To verify the performance of EDA-Alopex, several benchmark functions, listed in table1, are used in the test. The comparisons between PSO, DE, AEA, and

EDA-Alopex are carried out. The parameters of PSO are given as follows: the initial inertia weight, w_start=0.95, the final inertia weight, w_end=0.4, the percentage of maximum iteration for a linearly changing weight, w_for=0.7, acceleration factor, c1=c2=2, constriction factor Chi=1 and the maximum velocity is fixed at one-tenth of variable interval. The parameters involved in DE are listed as follows: the mutation operator, F=0.8, DE strategy is DE/rand-to-best/1/bin, crossover probability, CR=0.5. In the EDA-Alopex, the value of $\alpha 0$, $\alpha 1$, $\beta 0$ and $\beta 1$ are fixed at 1.2, 0.8, 0.8, 1.2, respectively. The value of ω and ν is set as 0.5, 0.9, separately.

In order to make a fair comparison, a total of 50 runs for each algorithm are implemented and the population size for the first two problems f_1-f_2 and the remaining eight functions f_3- f_{10} are fixed at 50 and 100, respectively. The maximum generations for all the four algorithms is set as 3000. At the same time, for the purpose of examining the performance of algorithms from another point of view, the target values of each test function is preset. The global minimum of the test functions f_1- f_4 is -1,-1.03, 0, and -4189.82, respectively. All the other test functions own a global minimum of 0.A trial is considered to be succeeded if the value obtained is less than or equal to the target within the maximum iterations, otherwise it is believed the current search is failed. The target for every test function is also listed in table 1.

Table 1. Test functions

No	Name	Function	D	Target	Bounds
f_1	*Easom*	$-\cos(x_1)\times\cos(x_2)\times\exp\left(-(x_1-\pi)^2+(x_2-\pi)^2\right)$	2	-0.99	[-100,100]
f_2	*Six hump*	$\left(4-2.1x_1^2+x_1^{\frac{4}{3}}\right)x_1^2+x_1x_2+\left(-4+4x_2^2\right)x_2^2$	2	-1.03	[-5,5]
f_3	*Ellipsoidal*	$\sum_{i=1}^{n}(x_i-i)^2$	10	0.001	[-10,10]
f_4	*Schwefel*	$-\sum_{i=1}^{n} x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)$	10	-4189.8	[-500,500]
f_5	*Exponential*	$1-\left(\exp\left(-0.5\sum_{i=1}^{n} x_i^2\right)\right)$	30	0.001	[-1,1]
f_6	*Griewank*	$1+\frac{1}{4000}\sum_{i=1}^{n} x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)$	30	0.1	[-600,600]
f_7	*Rastrigin*	$10n+\sum_{i=1}^{n}\left[x_i^2-10\cos(2\pi x_i)\right]$	30	100	[-5.12,5.12]
f_8	*Rosenbrock*	$\sum_{i=1}^{n-1}\left[100\left(x_{i+1}-x_i^2\right)^2+(x_i-1)^2\right]$	30	100	[-2.048,2.048]
f_9	*Sphere*	$\sum_{i=1}^{n} x_i^2$	30	0.001	[-5.12,5.12]
f_{10}	*Ackley*	$-20\exp\left(-0.02\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e$	30	1	[-30,30]

To make a fair comparison, the average best functions value (ABFV); standard deviations (SD) over 50 runs of the total ten problems for the four algorithms are quoted in table 2.

Table 2. Comparative average best function value and standard deviation over 50 runs

No	EDA-Alopex	AEA	PSO	DE
f_1	**-1±0**	-1±0	-1±0	-0.98±1.41E-01
f_2	**-1.03±0**	-1.03±0	-1.03±2.24E-16	-1.03±0
f_3	**0±0**	0±0	8.00E-02±2.74E-01	0±0
f_4	**-4.19E+03±1.84E-12**	-4.19E+03±1.84E-12	-3.40E+03±2.26E+02	-4.15E+03±7.42E+0
f_5	**0±0**	2.42E-16±4.86E-17	1.04E-16±2.66E-17	1.11E-16±0
f_6	**0±0**	5.61E-03±8.31E-03	1.51E-02±1.60E-02	.34E-07±6.85E-07
f_7	1.80E+01±1.96E+01	**7.53E+00±6.03E+00**	4.21E+01±1.07E+01	1.19E+02±8.16E+00
f_8	1.83E+01±1.21E-01	2.37E+01±2.54E-01	2.38E+01±7.68E+00	**1.35E+01±4.63E-01**
f_9	**1.25E-239±0**	8.07E-52±9.83E-52	8.13E-31±2.38E-30	2.00E-32±1.31E-32
f_{10}	**4.00E-15±0**	2.10E-14±4.00E-15	1.50E-14±4.00E-15	8.85E-15±2.22E-15

In table 2, the best values obtained by four algorithms are marked by bold. The number of no worse solutions compared to the other three algorithms given by EDA-Alopex is 8, by AEA is 5, by PSO is 2, by DE is 3. Totally, most of solutions given by EDA-Alopex is superior to the one obtained by other algorithms. In table 3, rate of successful minimizations (RS)， average generations(AG),average objective function evaluation numbers(AFEN) over 50 runs are listed to investigate the convergence and efficiency of algorithms. '-' denotes that the algorithm can not reach the target within the maximum iteration.

Table 3. Comparison of convergence and efficiency of four algorithms over 50 runs

No	EDA-Alopex			AEA			PSO			DE		
	RS	AG	AFEN	RS	AG	AFEN	RS	AG	AFEN	RS	AG	AFEN
f_1	50	12	1321	50	55	2882	50	335	16810	50	58	2964
f_2	50	7	778	50	17	892	50	62	3147	50	21	1106
f_3	50	32	6520	50	97	9786	43	880	88112	50	95	9574
f_4	50	193	38796	50	501	50244	-	0	0	38	891	89161
f_5	50	33	6696	50	100	10092	50	1149	114992	50	304	30518
f_6	50	60	12116	50	244	24540	50	1559	156016	50	634	63544
f_7	50	770	154108	50	773	77448	50	848	84886	2	2650	265050
f_8	50	16	3380	50	35	3568	50	343	34440	50	228	22852
f_9	50	51	10288	50	195	19642	50	1454	145522	50	454	45538
f_{10}	50	40	8192	50	130	13132	50	1330	133106	50	392	39250

In table 3, it can be clearly observed that 100% success for all the test problems given by EDA-Alopex and AEA. However, EDA-Alopex needs less iteration to reach the target value compared with AEA, especially for the number of objective function calls; a substantial decline is witnessed except function 7.

On the whole, EDA-Alopex achieves a balance between convergence speed and convergence precision. With regard to test function 7, the current parameter settings of EDA-Alopex are not recommended. We have confirmed that the new algorithm can achieve a better optimization result compared with the result given by AEA via using other parameters combination. To further validate the performance of the algorithms, three multi-apices and high-dimensional functions are employed for a scalability study. The test functions are Ackley (F1), Schwefel (F2) and Griewank (F3), the dimension is fixed at 100, 30 and 100 respectively and the global optimum of three test functions is 0, -12569.48 and 0, respectively. Number of maximum iterations is still fixed at 3000. The average best functions value (ABFV), standard deviations (SD) over 50 runs are listed in table 4.

Table 4. Optimization results for high-dimension test functions

No	EDA-Alopex	AEA	PSO	DE
F1	4.20E-14±7.00E-15	2.65E+00±2.17E-01	1.55E-01±2.90E-01	7.65E-02±1.04E-02
F2	-1.24E+04±2.59E+02	-1.17E+04±5.38E+02	-7.88E+03±5.63E+02	-9.18E+03±8.96E+02
F3	5.42E-04±2.24E-03	1.13E-03±2.87E-03	1.28E-02±3.60E-02	9.53E-02±2.39E-02

As can be seen from table 4, in terms of Ackley function, ABFV and SD obtained by EDA-Alopex are significantly better than the other three algorithms. In terms of the difficulty level of optimization, Schwefel function is a difficult test problem for optimizing. However, ABFV obtained by EDA-Alopex is closest to the global optimal solution .With regard to the function of Griwank, EDA-Alopex obtains the best results compared with other three algorithms. Consequently, EDA-Alopex also demonstrates a good performance for the optimization of the function of the high-dimensional and multiple local minimum. Overall speaking, EDA-Alopex outperforms others, considering in a comprehensive point of view.

5 Conclusion

In this paper, a new hybrid algorithm EDA-Alopex is proposed, which combines the Alopex with the EDA. In the EDA-Alopex, EDA is embedded into AEA, aiming at improving the performance of AEA. As a global optimization algorithm, AEA shows better performance compared with the basic EAs, such as GA, POS, and DE, and it also has a high potential of parallelism. In the EDA-Alopex, two populations used for correlation calculation in AEA are generated by EDA, for the purpose of improving the population diversity. Therefore, with a higher possibility, the offspring generated by the EDA-Alopex can contain not only the global probability information but also the local correlation information. Furthermore, an adaptive variation scheme for step size with iterations is introduced in the EDA-Alopex, which improves the convergence speed of the EDA-Alopex.

Ten widely used benchmark functions with different dimensions are utilized to investigate the performance of the EDA-Alopex. Each test problem possesses different characteristics, which examines the performance of the EDA-Alopex in a comprehensive way. Also different test methods are employed for comparison. The case study results show that the EDA-Aloepx can achieve faster speed and better solution, however, without a substantial cost in the execution time compared with AEA. Consequently, based on the analysis above, it can be concluded that the proposed algorithm can achieve better performance than traditional EAs.

References

1. Harth, E., Tzanakou, E.: Alopex: A Stochastic Method for Determining Visual Receptive Fields. Vision Research 14, 1475–1482 (1974)
2. Li, S.J.: An Alopex based Evolutionary Optimization Algorithm. Pattern Recognition and Artificial Intelligence 22(3), 452–456 (2009) (in Chinese)
3. Unnikrishnan, K.P., Venugopal, K.P.: A Correlation-Based Learning Algorithm for Feed-Forward and Recurrent Neural Networks. Neural Computation 6(3), 469–490 (1994)
4. Holland, J.H.: Adaptation in Natural and Artificial Systems. The university of Michigan Press, Ann Arbor (1975), 2nd ed. MIT Press, Cambridge (1992)
5. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: Proceeding of the 1995 IEEE International Conference on Neural Networks, pp. 1942–1948. IEEE Service Center, Piscataway (1995)
6. Storn, R., Price, K.: Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces. Journal of Global Optimization 11(4), 341–359 (1997)
7. Mühlenbein, H., Paaß, G.: From Recombination of Genes to the Estimation of Distributions. In: Mühlenbein, H., Bendisch, J., Voigt, H. (eds.) PPSN 1996. LNCS, vol. 1141, pp. 178–187. Springer, Heidelberg (1996)
8. Mühlenbein, H., Höns, R.: The Estimation of Distributions and the Minimum Relative Entropy Principle. Evolutionary Computation 13(1), 1–27 (2005)
9. Zhou, S.D., Sun, Z.Q.: A Survey on Estimation of Distribution Algorithms. Acta Automatica Sinica 33(2), 113–124 (2007) (in Chinese)
10. Abraham, A., Corchado, E., Corchado, J.M.: Hybrid Learning Machines. Neurocomputing 72(13-15), 2729–2730 (2009)
11. Lozano, M., García-Martínez, C.: Hybrid Metaheuristics with Evolutionary Algorithms Specializing in Intensification and Diversification: Overview and Progress Report. Computers & Operations Research 37(3), 481–497 (2010)
12. González, C., Lozano, J.A., Larrañaga, P.: Mathematical Modeling of UMDAc Algorithm with Tournament Selection Behavior on Linear and Quadratic functions. International Journal of Approximate Reasoning 31(3), 313–340 (2002)
13. Sun, J.Y., Zhang, Q.F., Edward, P.K.T.: DE/EDA: A New Evolutionary Algorithm for Global Optimization. Information Sciences 169(3-4), 249–262 (2005)
14. Grahl, J., Minner, S., Rothlauf, F.: Behavior of UMDAc with Truncation Selection on Monotonous Functions. In: IEEE Congress on Evolutionary Computation (CEC), vol. 3, pp. 2553–2559 (2005)

A Hybrid Swarm Intelligent Method Based on Genetic Algorithm and Artificial Bee Colony

Haiyan Zhao[1], Zhili Pei[1,2], Jingqing Jiang[1], Renchu Guan[3], Chaoyong Wang[4], and Xiaohu Shi[3,5,*]

[1] College of Computer Science and Technology, Inner Mongolia University for Nationalities, Inner Mongolia, Tongliao 028043
[2] Mathematics School and Institute, Jilin University, Jilin, Changchun, 130012
[3] College of Computer Science and Technology, Jilin University, Jilin, Changchun, 130012
shixh@jlu.edu.cn
[4] School of applied science, Jilin Teachers' Institute of Engineering and Technology, Jilin, Changchun,130052
[5] State Key Lab. for Novel Software Technology, Nanjing University, Jiangsu, Nanjing, 210093

Abstract. By integrating artificial bee colony and genetic algorithm, a novel hybrid swarm intelligent approach is proposed in this paper. The main idea of the approach is to obtain the parallel computation merit of GA and the speed and self-improvement merits of ABC by sharing information between GA population and bee colony. To exam the proposed method, it is applied to 4 benchmark functions for different dimensions. For comparison, simple GA and ABC methods are also executed. Numerical results show that the proposed hybrid swarm intelligent method is effective, and the precision could be improved.

Keywords: Swarm intelligence, artificial bee colony, genetic algorithm, optimization problem.

1 Introduction

Swarm Intelligence (SI) has become an exciting development in computer science. It amounts to designing any algorithms inspired by the collective behavior of social insect colonies and other animal societies [1]. However, some researchers consider the collection of any interacting agents or individuals as "swarm". So, those former developed population-based algorithms such as Genetic Algorithm (GA) are considered as SI methods as well. The most popular SI method is Particle Swarm Optimization (PSO), which is developed by Eberhart and Kennedy in 1995 [2]. It models the social behavior of bird flocking or fish schooling. Another classic SI is Ant Colony Optimization (ACO), of which the swarm refers to the searching food ant colony[3].

Artificial bee colony (ABC) algorithm is an interesting SI algorithm originally developed by Karaboga in 2005 [4]. It is a method for optimization on metaphor of the foraging behavior of bee colony. Because ABC has a lot of advantages on memory,

[*] Corresponding author.

local search, solution improvement mechanism, and so on, it is able to get excellent performance on optimization problems [5-8]. Presently, ABC has attracted much attention in the related research fields. For example, Karaboga and Basturk extended ABC from unconstrained optimization to constrained optimization problems by introducing a constrained handling method [9], Wong and Chong developed an efficient ABC-based method for traveling salesman problem[10], Chong et. al. proposed a bee colony optimization algorithm for job shop schedule problem [11], Fathian et. al. developed a honey-bee mating optimization algorithm for clustering [12], Kang et. al. applied ABC method to structural inverse analysis [13]. One can refer to Ref.[14] for a comprehensive review of ABC-based methods.

In this paper, a novel hybrid swarm intelligent approach is proposed by integrating artificial bee colony and genetic algorithm. The algorithm executes the two systems simultaneously and exchanges information between bee colony and chromosome population. Therefore the hybrid approach possesses both the parallel computation merit of GA and the speed and self-improvement merits of ABC. To test the effectiveness of our proposed method, it is applied to a series of benchmark test functions, for comparison, the simple GA and ABC are executed on the same set as well. Numerical results show that the proposed hybrid approach is effective, and is superior to both of simple GA and ABC methods.

2 Background

2.1 Simple Genetic Algorithm (SGA)

Genetic algorithms (GAs) are a family of computational models developed by Holland [15,16], which is based on the principles of natural biological evolution. For a specific problem, GA codes a solution candidate as an individual chromosome. The approach begins with an initial chromosome population which represent the set of initial search points in the solution space of the problem. Then the genetic operators such as selection, crossover and mutation are applied to obtain a new generation of chromosomes. Since the operators are under the principle of "survival of the fittest, extinction of the unfitness", it is expected that over all the quality the chromosomes will be improved with the generation increasing. This process is executed iteratively until the termination criterion is met, and the best chromosome of the last generation is reported as the final solution.

2.2 Artificial Bee Colony (ABC)

Bee colony is one of the populations with thinnest social division in nature. Different tasks are performed by different individuals. The goal of the whole population is to maximize the nectar amount stored in the hive by performing efficient division of labor and self-organization. There are three essential components among a bee colony: food sources, employed foragers and unemployed foragers [7]. Each food source is companied with a value which represents the "profitability" of the source. Employed bees are those associated with a particular food source for exploring currently. They can carry the information of the food sources they are "employed" at and share the information with a certain probability. There are two kinds of unemployed bees, one is onlooker, and the other is scout. Onlookers wait in the nest and find a food source through the information shared with employed foragers, while scouts randomly search the environment for new valuable food sources [7].

In ABC algorithm, a food source represents a possible solution to the optimization problem and the nectar amount corresponds to the fitness of the solution. There are three main groups of bees, namely, employed bees, onlookers and scouts. Both the numbers of employed bees and onlookers are equal to that of food sources. Denote the food source number as *SN*, the position of the *i*th food source as $\boldsymbol{x}_i$ (i=1,2,…,*SN*). So *SN* food sources are randomly produced and assigned to *SN* employed bees correspondingly at the beginning of the approach. And then employed bee associated to the *i*th food source searches for new solution according to Eq.(1)

$$v_{ij}=x_{ij}+\varphi_{ij}\,(x_{ij}-x_{kj}), \tag{1}$$

where j=1,2,…,D, and D is the dimension of the optimized problem, φ_{ij} is a random generalized real number within the range [-1, 1], k is a randomly selected index number in the colony. Then $\boldsymbol{v}_i$ is compared with its original position $\boldsymbol{x}_i$, and the better one should be remained. Next, each onlooker chooses a food source with the probability (Eq.(2)) and produces a new source in selected food source site by Eq.(1).

$$p_i = fit_i \Big/ \sum_{j=1}^{SN} fit_j \tag{2}$$

where fit_i is the fitness of the solution $\boldsymbol{x}_i$.

After onlookers are all allocated a food source, if a source is found that its fitness hasn't been improved for a given number (this number is called *limit*) steps, it should be abandoned, and the employed bee associated with which becomes a scout and makes a random search by Eq.(3).

$$x_{ij}=x_j^{min}+r(x_j^{max}-x_j^{min}), \tag{3}$$

where r is a random real number within the range [0, 1], and x_j^{min} and x_j^{max} are the lower and upper borders in the jth dimension of the problem space.

The main steps of the algorithm could be summarized as below:

1. Initialize Population
2. Place the employed bees on their food sources
3. Place the onlookers on the food sources depending on their nectar amounts
4. Send the scouts to the search area for discovering new food sources
5. Memorize the best food source found so far
6. If requirements are met, output the best solution, otherwise go to step (2).

3 Hybrid Method Based on GA and ABC

After GA was developed 35 years ago, it has become one of the most highlight algorithms in the related research field. The key of its success is the inherent parallel characteristics, which guarantees the approach could find the global optimization with large probability. ABC is also parallel inherently, while its most significant features are the solution's self-improvement and local search ability. In this paper, we would like to obtain both of their excellent features by synthesizing the two algorithms. So, a novel approach called hybrid swarm intelligent method based on GA and ABC (HSIGA) is proposed.

The main idea of HSIGA is to exchange information between GA population and bee colony. At beginning, the approach executes GA and ABC simultaneously, and then two information exchange processes will be introduced. Firstly, bee colony shares the information from GA population through scouts. During each iteration, the scouts, if we have, will obtain GA information with a given probability. The information randomly comes from GA population, while those individuals with high fitness are prefered to be selected. In the second process, a small given number of individuals will be randomly selected from GA population and bee colony simultaneously. With the same schedule as above, those individuals with high fitness have more opportunities to be choosen. Then the selected individuals will be matched in pair, and crossovered. The obtained offspring will be added into GA population. We define these two processes as Information Exchange Process1 and Information Exchange Process2, respectively. The main steps of the algorithm could be described as follows:

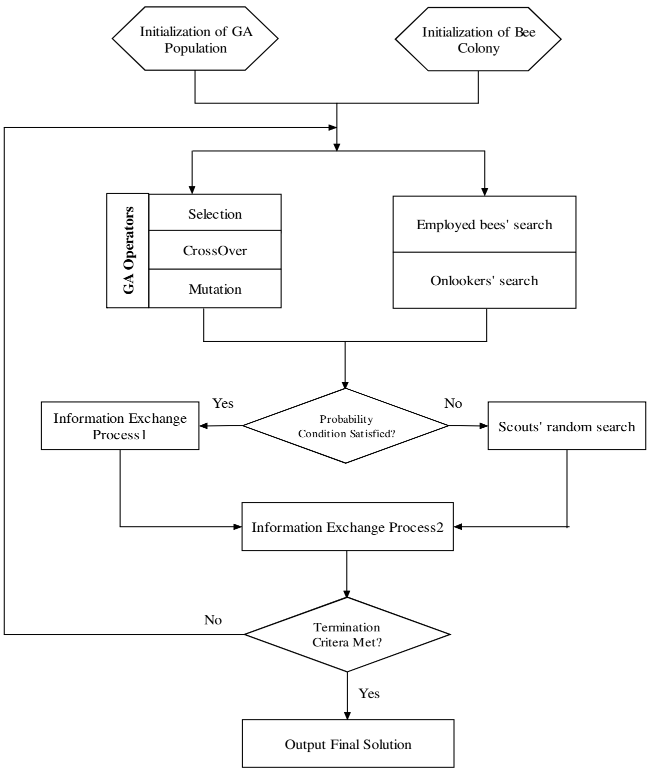


Fig. 1. Flow chart of HSIGA

1. Initialize GA and ABC sub-systems respectively.
2. Execute selection, crossover and mutation operators on GA population.
3. Execute employed bees' search and onlookers' selection and search processes on bee colony.
4. Execute employed scouts' search process. The start search points should be determined according to a given probability, whether they are randomly produced by Eq.(3) or they are obtained through the Information Exchange Process1.
5. Execute Information Exchange Process2.
6. Memorize the best solution as the final solution and stop if the best individual in one of the two sub-systems satisfies the termination criterion.
   The flow chart of the HSIGA is shown in Fig 1.

4 Numerical Results

The proposed HSIGA is tested using 4 benchmark functions. For comparison, SGA and ABC are also executed on these 4 functions. Table. 1 shows the details of test functions. Simulations are performed in C++ with a 2.53GHz Pentium PC. In SGA, the population is set as 60 and the rate of mutation is 0.02. The parameters of ABC are as follows: colony size is 40, *limit* is 100. In HSIGA, the population of GA and the size of bee colony are both set as 60, the rate of GA mutation is 0.02, the *limit* in ABC subsystem is 100, and the probability of Information Exchange Processs1 is 0.5, respectively.

Table 1. Test functions used in the experiments

Function	Formulae	Range	Minimum value
Sphere	$f(X)=\sum_{i-1}^{n} x_i^2$	[-100,100]	$f(\mathbf{0})=0$
Rosenbrock	$f(X)=\sum_{i-1}^{n-1}\left[100\left(x_{i+1}-x_i^2\right)^2+(x_i-1)^2\right]$	[-30,30]	$f(\mathbf{1})=0$
Griewank	$f(X)=\frac{1}{4000}\sum_{i-1}^{n} x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	[-600,600]	$f(\mathbf{0})=0$
Rastrgin	$f(X)=\sum_{i-1}^{n}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	[-5.12,5.12]	$f(\mathbf{0})=0$

Table. 2 to Table. 5 list the comparison results of these 3 methods for 4 benchmark functions of 5 different dimensions. Each of experiments was repeated 50 times, and the mean values and standard deviations are listed in the tables, of which the bold mean values are the best results among the 3 methods. From the tables it can be seen that with the dimension increasing, the precisions of all the 3 methods are decreased. For Sphere function, HSIGA is better than the other two approaches in all of the 5 dimensions. For the Rastrgin function, only ABC shares the best results with HSIGA at dimension 10. HSIGA monopolizes the best results of the other 4 dimensions. HSIGA also obtains the best results of 4 dimensions for both of the other two functions, while

Table 2. Comparison Results for Sphere function

Dimension		10	30	50	100	200
SGA	Mean	2.18E-14	6.51E-14	1.08E-13	2.17E-13	5.86E+00
	Std	3.19E-30	5.10E-29	5.10E-29	7.65E-29	3.60E+01
ABC	Mean	6.41E-17	6.46E-16	1.67E-15	9.88E-15	1.13E-13
	Std	1.48E-17	8.528E-17	3.03E-16	4.49E-15	1.06E-13
HSIGA	Mean	**4.80E-17**	**5.38E-16**	**1.34E-15**	**4.41E-15**	**2.23E-14**
	Std	1.11E-17	6.48E-17	1.79E-16	8.84E-16	9.04E-15

Table 3. Comparison Results for Rosenbrock function

Dimension		10	30	50	100	200
SGA	Mean	3.48E+02	1.47E+03	2.11E+03	3.72E+03	6.42E+03
	Std	520.6786	1972.145	1925.509	2535.072	4981.866
ABC	Mean	8.45E-02	**1.06E-01**	2.68E-01	1.21E+00	3.34E+00
	Std	0.078007	0.141269	0.379678	2.493321	4.379442
HSIGA	Mean	**8.00E-02**	1.14E-01	**1.17E-01**	**1.44E-01**	**4.10E-01**
	Std	0.078156	0.214262	0.199364	0.247321	0.472663

Table 4. Comparison Results for Griewank function

Dimension		10	30	50	100	200
SGA	Mean	2.62E-16	7.47E-09	3.41E-15	**2.19E-15**	3.56E-05
	Std	8.62E-17	5.28E-08	1.69E-14	1.88E-16	0.000197
ABC	Mean	8.88E-17	6.57E-16	1.76E-15	9.69E-15	1.19E-13
	Std	4.49E-17	1.42E-16	2.85E-16	4.91E-15	1.23E-13
HSIGA	Mean	**3.11E-17**	**4.73E-16**	**9.99E-16**	2.22E-15	**3.42E-14**
	Std	5.04E-17	4.92E-17	1.96E-16	3.17E-16	2.19E-14

ABC possesses the best results at dimension 30 for Rosonbrock function, and SGA is executed best at dimension 100 for Griewank function, respectively. Fig. 2 gives a comparison of the error evolution curves of a typical run on Griewank function at dimension 200. From the tables and Fig.2, we could find that ABC and HSIGA are much better than SGA overall, and HSIGA performs best among all the 3 methods.

Table 5. Comparison Results for Rastrgin function

Dimension		10	30	50	100	200
SGA	Mean	4.51E+00	3.64E+01	7.65E+01	1.94E+02	4.23E+02
	Std	2.292706	7.88172	13.5188	23.39157	33.16444
ABC	Mean	**0.00E+00**	3.69E-15	1.19E-11	2.00E-02	1.12E+00
	Std	0	4.72E-15	4.17E-11	0.140701	1.020167
HSIGA	Mean	**0.00E+00**	**1.42E-16**	**3.43E-14**	**3.90E-10**	**1.63E-01**
	Std	0	4.87E-16	4.66E-14	1.6E-09	0.346746

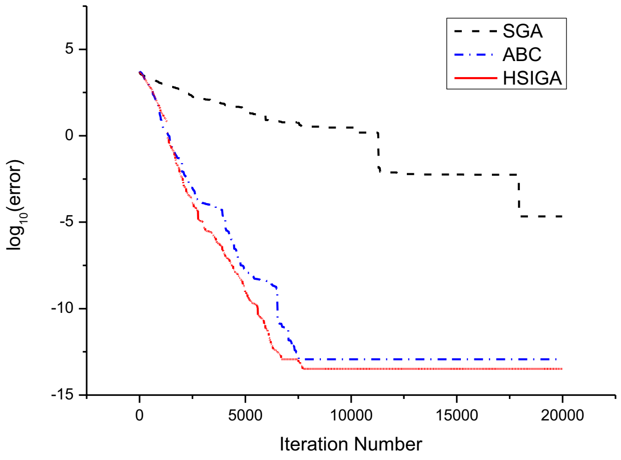


Fig. 2. The error evolution curves of a typical run on Griewank function of dimension 200

5 Conclusions

A novel hybrid swarm intelligent approach is proposed based on artificial bee colony and genetic algorithm in this paper. By introducing two information exchange processes between GA population and bee colony, the hybrid approach could possess both merits of GA and ABC methods. To validate our proposed method, it is applied to 4 benchmark functions for different dimensions together with simple GA and ABC methods. Numerical results show that the proposed method is much better than SGA, and is also superior to ABC approach.

Acknowledgments. The authors are grateful to the support of the National Natural Science Foundation of China (60703025, 10872077), the science-technology development project of Jilin Province of China (20090152, 20080708), the Special Fund for Basic Scientific Research of Central Colleges (421032011421), Jilin University and Inner Mongolia Natural Science Foundation (200711020807, 20080404MS0908).

References

1. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, NY (1999)
2. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, vol. (4), pp. 1942–1948. IEEE Service Center, Piscataway (1995)
3. Dorigo, M.: Optimization, Learning and Natural Algorithms (in Italian). PhD thesis, Dipartimento di Elettronica, Politecnico di Milano, Italy, 140 (1992)
4. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical Report TR06, Erciyes University, Engineering Faculty, Computer Engineering Department (2005)
5. Basturk, B., Karaboga, D.: An artificial bee colony (abc) algorithm for numeric function optimization. In: IEEE Swarm Intelligence Symposium 2006, Indianapolis, Indiana, USA (May 2006)
6. Karaboga, D., Basturk, B.: A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (abc) algorithm. Journal of Global Optimization 39(3), 459–471 (2007)
7. Karaboga, D., Basturk, B.: On the performance of artificial bee colony (abc) algorithm. Applied Soft Computing 8(1), 687–697 (2008)
8. Karaboga, D., Akay, B.A.: Comparative Study of Artificial Bee Colony Algorithm. Applied Mathematics and Computation 214(1), 108–132 (2009)
9. Karaboga, D., Basturk, B.: Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems. In: Melin, P., Castillo, O., Aguilar, L.T., Kacprzyk, J., Pedrycz, W. (eds.) IFSA 2007. LNCS (LNAI), vol. 4529, pp. 789–798. Springer, Heidelberg (2007)
10. Wong, L.P., Chong, C.S.: An Efficient Bee Colony Optimization Algorithm for Traveling Salesman Problem using Frequency-based Pruning. In: Proceeding of the IEEE International Conference on Industrial Informatics, INDIN, pp. 775–782 (2009)
11. Chong, C.S., Low, M.Y.H., Sivakumar, A.I., Gay, K.L.: A Bee Colony Optimization Algorithm to Job Shop Schedule. In: Proceedings of the Winter Simulation Conference, pp. 1954–1961 (2006)
12. Fathian, M., Amiri, B., Maroosi, A.: Application of honey-bee mating optimization algorithm on clustering. Applied Mathematics and Computation 190, 1502–1513 (2007)
13. Kang, F., Li, J.J., Xu, Q.: Structural inverse analysis by hybrid simplex artificial bee colony algorithms. Computers and Structures 87, 861–870 (2009)
14. Karaboga, D., Akay, B.A.: A survey: algorithms simulating bee swarm intelligence. Artif. Intell. Rev. 31, 61–85 (2009)
15. Holland, J.H.: Adaptation in Natural and Artificial System. The University of Michigan Press, Ann Arbor (1975)
16. Goldberg, D.E.: Genetic Algorithms in Search, Optimization & Machine Learning. Addison-Wesley, Reading (1989)

A Hybrid PSO/GA Algorithm for Job Shop Scheduling Problem

Jianchao Tang[1,2], Guoji Zhang[3], Binbin Lin[3], and Bixi Zhang[4]

[1] School of Computer Science and Engineering, South China University of Technology, 510641 Guangzhou, Guangdong, China
michealtang@163.com
[2] Computer School, South China Normal University, 510631 Guangzhou, China
[3] College of Science, South China University of Technology, 510641 Guangzhou, China
[4] School of Economics and Management, Guangdong University of Technology, 510090 Guangzhou, China

Abstract. The job shop scheduling problem is a well-known NP hard problem, on which genetic algorithm is widely used. However, due to the lack of the major evolution direction, the effectiveness of the regular genetic algorithm is restricted. In this paper, we propose a new hybrid genetic algorithm to solve the job shop scheduling problem. The particle swarm optimization algorithm is introduced to get the initial population, and evolutionary genetic operations are proposed. We validate the new method on seven benchmark datasets, and the comparisons with some existing methods verify its effectiveness.

Keywords: particle swarm optimization algorithm, hybrid genetic algorithm, Job shop scheduling problem.

1 Introduction

The job shop scheduling problem (JSSP) is a well known NP-HARD problem [1]. Only small scale problem can be solved exactly. So, researchers proposed some methods to get nearly optimal solutions, such as simulated annealing (SA), taboo search (TS), genetic algorithm (GA) and hybrid algorithms [2]-[9] .

In the last few decades, genetic algorithm has been widely applied to solve JSSP. In these researches, genetic algorithms are presented with different encoding schemes and different operators. A comprehensive survey on the encoding schemes has been given by L. Wang (2003) [10], in which, 9 kinds of encoding schemes are introduced. C.G. Wu proposed a genetic algorithm to solve JSSP based on the concepts of operation template and virtual job shop (2004) [6]. L. Wang and D. Z. Zheng developed a general, parallel and easily implemented hybrid optimization framework of GA and SA, and applied it to job-shop scheduling problems (2001) [8]. Compared with other traditional heuristic search algorithms, genetic algorithm starts with a set of initial solutions. Due to its potential parallelism, it can travel the whole problem space strongly, so GA is a popular approach for JSSP. However, due to the lack of major evolution direction, the relative deviations are not stable. And the particle swarm optimization (PSO) is another stochastic approximation algorithm, X. Chen and Y. Li analyzed its convergence [11]-[12]. In this paper, PSO algorithm is adopted to provide a better induct, and evolutionary genetic operators are proposed.

This paper is organized as follows. In section 2, combining PSO algorithm and an evolutionary genetic algorithm (EGA), a new hybrid algorithm (PSO/GA) is presented to solve JSSP. In section 3, the computational and comparative results on benchmark instances are given. Finally, we conclude the paper with a summary in section 4.

2 The PSO/GA Algorithm for JSSP

2.1 Description of Job Shop Scheduling Problem

The JSSP can be described as follows: Given n jobs $J=\{J_1,J_2,...,J_n\}$ which must be processed on m machines $M=\{M_1,M_2,...,M_m\}$, each job consists of predetermined operations in a specific sequence on a given machine for a fixed duration without interruption. Each machine can handle at most one operation at a time. The objective is to find a feasible schedule with the shortest makespan. The operation number of job J_i is denoted as n_i , ($i=1,2,...,n$), thereby, the total number of operations is $s=\sum_{i=1}^{n} n_i$ .Many algorithms request that each job must pass through all the machines, this constraint is relaxed in this paper.

2.2 Encoding and Decoding of the PSO/GA Algorithm

The PSO/GA algorithm encodes a schedule as a sequence of operations and each gene stands for an operation number. For an n-job and m-machine problem, each chromosome contains $s=\sum_{i=1}^{n} n_i$ genes. Each individual chromosome in the population represents a schedule, which means a permutation of the operations for all jobs. For example, a given 3×3 JSSP is shown in Table 1. In this table, the i th row, the k th column's content is (j,t), which means job i 's operation k is processed on machine j for t processing time.

Table 1. 3×3 JSSP

Job no.	Oper. 1	Oper. 2	Oper. 3
1	2, 3	3, 7	1, 5
2	1, 6	3, 9	2, 8
3	3, 4	1, 3	2, 9

Continuous numbers are assigned to operations for all jobs, that is, natural numbers 1 to 9 are assigned to all nine of operations. Each operation can be denoted as a five tuple: $p(f,i,k,j,t)$, which means job i's operation k is processed on machine j for t processing time, and we called it the f operation ($f = 1, 2, ..., s$). According to the definition, each chromosome can be mapped to a 9×5 matrix, which represents a schedule.

For example, the chromosome is shown in Fig.1 means a schedule's operation sequence. And its corresponding 9×5 matrix is shown in Fig.2. The first row of the matrix is the chromosome in Fig.1.

9	1	3	6	8	2	5	7	4

Fig. 1. An example of chromosome representation

$$\begin{pmatrix} 9 & 1 & 3 & 6 & 8 & 2 & 5 & 7 & 4 \\ 3 & 1 & 1 & 2 & 3 & 1 & 2 & 3 & 2 \\ 3 & 1 & 3 & 3 & 2 & 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 2 & 1 & 3 & 3 & 3 & 1 \\ 9 & 3 & 5 & 8 & 3 & 7 & 9 & 4 & 6 \end{pmatrix}^T$$

Fig. 2. The information matrix of the Figure 1's chromosome

Any individual chromosome in a population can be mapped to a sequence of operations according to the 9×5 matrix. Such a sequence of operations represents a schedule. Recall that the operations in each job have precedence constraints, therefore not all the chromosomes are legitimate. To ensure the legitimacy, each chromosome is decoded according to the conflict resolution as follows:

(1) Scan the genes in the chromosomes.
(2) The first available operation is added to the task queue.
(3) Repeat the step (1) and (2) till all of the operations are arranged.

For example, the chromosome in Fig.1 can be decoded a real schedule shown in Fig.3.

1	2	7	4	3	8	5	9	6

Fig. 3. The legitimate chromosome for Fig.1

According to the legitimate chromosome, the makespan of the schedule can be calculated with the Gantt chat.

2.3 PSO Algorithm for Initialization

The process of initialization often plays an important role in the implementation. Because regular genetic algorithm generates initial solutions randomly, so it is lacking in the major evolution direction, the relative deviations are not stable, and the effectiveness is restricted. In this work, particle swarm algorithm (PSO) is introduced to provide induct to get better initial solutions.

Defining that the search space of PSO is p-dimensional and then the i th particle of the swarm is $x_i = (x_{i1}, x_{i2}, ..., x_{is})$, the velocity of the particle is $v_i = (v_{i1}, v_{i2}, ..., v_{is})$, where $x_{ij}, v_{ij} \in [a,b](j = 1,2,...,s)$. The local best particle is denoted as $pbest_i = (pbest_{i1}, pbest_{i2}, ..., pbest_{is})$, the global best particle is denoted as $gbest = (gbest_1, gbest_2, ..., gbest_s)$. Take 3×3 JSSP as an example, the particle representation is as follows.

Table 2. Particle representation for 3×3 JSSP

	1dim.	2dim.	3dim.	4dim.	5dim.	6dim.	7dim.	8dim.	9dim.
Position x_i	6.5	3.2	-4.7	1.0	-1.4	5.1	-3.8	2.3	7.2
No.	8	6	1	4	3	7	2	5	9

Generate the position x_i randomly, and then in ascending order to get the second row in Table 2 , which corresponds to the chromosome in section 2.2.

From the initial particles, the particle updates its position and velocity by (1), (2) and (3) iteratively as follows.

$$\omega_k = \omega_{\max} - \frac{(\omega_{\max} - \omega_{\min}) * k}{M} \quad . \tag{1}$$

$$v_{ij}^{k+1} = \omega_k * v_{ij}^k + c_1 * rand() * (pbest_{ij}^k - x_{ij}^k) + c_2 * rand() * (gbest_j^k - x_{ij}^k) \; . \tag{2}$$

$$x_{ij}^{k+1} = x_{ij}^k + v_{ij}^{k+1} \quad . \tag{3}$$

In (1), ω denotes inertia weight ($\omega_{\min} = 0.4$ and $\omega_{\max} = 1.2$), M denotes the maximum of iterative generation and k is the current iterative generation.

By applying PSO algorithm, the process of initialization is performed.

2.4 Fitness Function

We cite the fitness function in [6] as follows:

$$f(x) = \frac{\alpha R}{makespan(x)} \quad . \tag{4}$$

Where $\alpha \in [0,1]$, $R = \frac{1}{m}\sum_{i=1}^{9} L(i)_5$ ($L(i)_5$ represents the fifth column and the i th row of the matrix L shown in Fig.2, $makespan(x)$ is the completion time for all operations, and the value of the fitness function is ranged from 0 to 1.

2.5 Crossover Operator and Mutation Operator

In this paper, single linear order crossover is adopted. And to ensure to get the optimal solution, the elitist strategy (Goldberg (1089)) is applied. Take 3×3 JSP as an example, suppose the randomly selected position is 4, then we get the children q_1 and q_2, but q_1 and q_2 are not legitimate. We call the duplicate genes in q_1 as superfluous genes which are 6 and 7 in Fig.4, and the duplicate genes in q_2 as lacking genes which are 5 and 1. Then exchange them to get the new legal q_1 and q_2.

$$\begin{array}{l} p_1{:}2\ 6\ 4\ 7\ \underline{3\ 5\ 8\ 9\ 1} \\ p_2{:}4\ 5\ 2\ 1\ \underline{8\ 7\ 6\ 9\ 3} \end{array} \rightarrow \begin{array}{l} q_1{:}2\ \underline{\underline{6}}\ 4\ \underline{\underline{7}}\ \underline{8\ 7\ 6\ 9\ 3} \\ q_2{:}4\ \underline{\underline{5}}\ 2\ \underline{\underline{1}}\ \underline{3\ 5\ 8\ 9\ 1} \end{array} \rightarrow \begin{array}{l} q_1{:}2\ 5\ 4\ 1\ \underline{8\ 7\ 6\ 9\ 3} \\ q_2{:}4\ 6\ 2\ 7\ \underline{3\ 5\ 8\ 9\ 1} \end{array}$$

Fig. 4. An example of single linear order crossover

For the mutation operator, inverse mutation is adopted. By randomly selecting a candidate chromosome and then randomly selecting two positions, denote as N_1 and N_2 , and reverse the orders of genes between N_1 and N_2 .

2.6 The Flowchart of the PSO/GA Algorithm

The flowchart of the PSO/GA algorithm is shown in Fig. 5.As can be seen from Figure 5 that in hybrid PSO/GA algorithm, PSO algorithm is applied to get initialization solutions for GA, and evolutionary operations are performed.

3 Experimental Results

To illustrate the performance of the introduced PSO/GA for JSSP, we performed tests on 7 benchmark instances with different sizes. The population size is set to 100, the PSO's and GA's maximum of iterative generation M_1 and M_2 are set to 20 and 200. The crossover rate is 0.65 and the mutation rate is 0.95.

Compared with the PSO/GA algorithm, the initial solution of the evolution genetic algorithm (EGA) is produced randomly. The EGA is an improved algorithm according to the GA in [6]. The AGAA algorithm is proposed by S. H. Liu and L. H. Wang (2008) [9] which based on the combination of adaptive genetic algorithm and improved ant algorithm. Results for PSO/GA are shown in Table 4 and compare with GA, AGAA and EGA. In Table 4, the column "BKS" contains the best known solution, the column "BS" contains the minimum makespan, the column "RD" contains

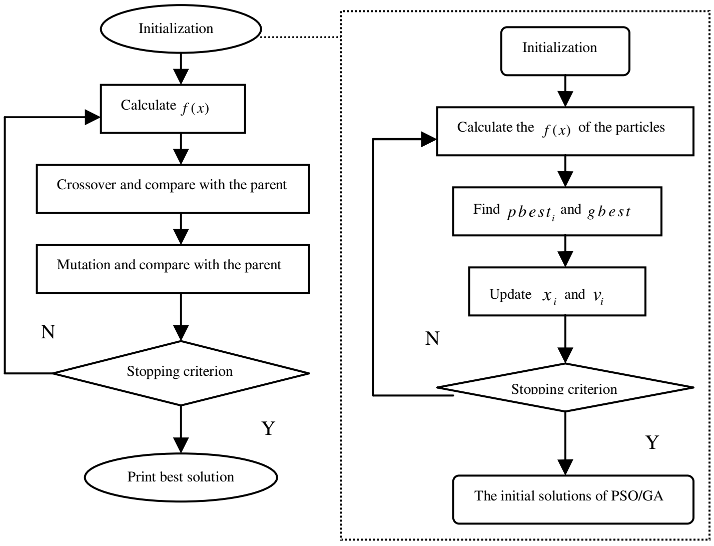

Initialization
Calculate $f(x)$
Crossover and compare with the parent
Mutation and compare with the parent
N
Stopping criterion
Y
Print best solution
Initialization
Calculate the $f(x)$ of the particles
Find $pbest_i$ and $gbest$
Update x_i and v_i
N
Stopping criterion
Y
The initial solutions of PSO/GA

Fig. 5. The flowchart of the PSO/GA

the average percent relative increase in makespan with respect to the best known solution after 20 respective runs.

$$RD = \frac{C_{avg} - C_{best}}{C_{best}} \times 100\% \quad . \tag{5}$$

In (5), C_{avg} denotes the average of the 20 times results, and C_{avg} is the best known solution.

Table 3. Results for PSO/GA and GA, AGAA, EGA for benchmark instances

	n,*m*	BKS	PSO/GA		EGA		AGAA		GA	
			BS	RD (%)	BS	RD (%)	BS	RD (%)	BS	RD (%)
FT06	6,6	55	**55**	0	55	1	55	0	55	0
FT10	10,10	930	965	4.2	965	5.3	990	6.4	997	11.87
LA01	10,5	666	666	0	666	0.55	666	0	666	0
LA06	15,5	926	926	0	926	0	943	0.16	938	0.9
LA11	20,5	1222	1222	0.09	1222	0.62	1227	0.05	1235	0.98
LA16	10,10	945	946	1.73	946	3.05	969	2.64	986	4.6
LA21	15,10	1046	1081	2.56	1081	4.59	1096	4.79	1156	10.56

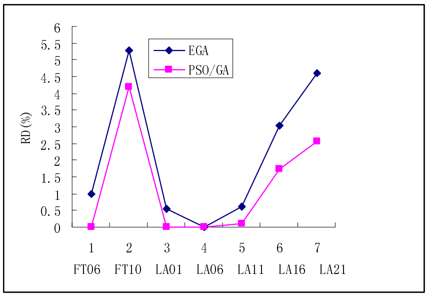


Fig. 6. Relative Deviations for PSO/GA and EGA

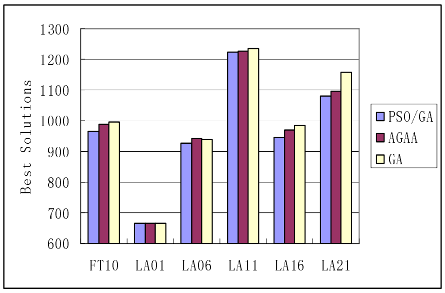


Fig. 7. Solutions for Three Algorithms

As can be seen from Table 4 that, for FT06, LA01, LA06 and LA11 benchmark instances, PSO/GA algorithm can find the best known scheduling solutions. With regard to FT10, LA16, LA21, PSO/GA can find better sub-optimal scheduling solutions than other algorithms cited.

And for all instances, EGA outperforms GA, which indicates the effectiveness of the proposed evolutionary genetic operators. Moreover, Fig.6 shows that PSO/GA has consistently lower performance Relative Deviations than EGA, which verifies that the initialization by PSO can lead to better stability. And PSO/GA can find better scheduling solutions and lower relative deviations than other cited algorithms for all seven benchmark instances.

4 Conclusion

In this paper, a hybrid algorithm (PSO/GA) combining PSO and EGA is proposed to solve the job shop scheduling problem. With the induction of the PSO, the PSO/GA algorithm has a relatively better stability. And proposed evolutionary genetic operators lead to better solution than the regular GA. Experimental results verify the effectiveness of our algorithm that it can find the best known solutions or the sub-optimal solutions for all cited benchmark instances.

Acknowledgments. The authors are grateful to the support of National Natural Science Foundation of China under Grant No. 70971026, and Natural Science Foundation of Guangdong Province under Grant No. 9151009001000040.

References

1. Lenstra, J.K., Kan, A.H.G.R., Brucker, P.: Complexity of machine scheduling problems. Annals of Discrete Mathematics 7, 2100–2118 (1977)
2. Kolonko., M.: Some new results on simulated annealing applied to the job shop scheduling problem. European Journal of Operational Research 113(1), 123–136 (1999)

3. Nowicki, E., Smutnicki, C.: A fast taboo search algorithm for the job-shop problem. Management Sciences 42(6), 797–813 (1996)
4. Nowicki, E., Smutnicki., C.: An advanced taboo search algorithm for the job shop problem. Journal of Scheduling 8(2), 145–159 (2005)
5. Croce, F.D., Tadei, R., Volta., G.: A genetic algorithm for the job shop problem. Computers & Operations Research 22, 15–24 (1995)
6. Wu., C.G.: Genetic Algorithm Application on the Job Shop Scheduling Problem China. In: Proceedings of the Third International Conference on Machine Learning and Cybermetics, vol. 8, pp. 2102–2106 (2004)
7. Bael, P.V., Devogelaere, D., Rijckaert, M.: A Hybrid Evolutionary Search Scheduling Algorithm to Solve the Job Shop Scheduling Problem. In: Proceedings of the 1999 Congress on Evolutionary Computation, pp. 1103–1109 (1999)
8. Wang, L., Zheng., D.Z.: An effective hybrid optimization strategy for job-shop scheduling problems. Computers and Operations Research, China, 585–596 (2001)
9. Liu, S.H., Wang, L.H.: Hybrid genetic algorithm for job shop scheduling problem. Computer Engineering and Applications, China 44(29), 73–75 (2008)
10. Wang, L.: Job shop scheduling and Genetic Algorithm, pp. 68–76. Tsinghua University Press, Beijing (2003)
11. Chen, X., Li, Y.: A Modified PSO Structure Resulting in High Exploration Ability with Convergence Guaranteed. IEEE Transactions on System, Man and Cybernetics: Part B 37(5), 1271–1289 (2007)
12. Chen, X., Li, Y.: On Convergence and Parameters Selection of an Improved Particle Swarm Optimization. International Journal of Control, Automation, and Systems, Korea 6(4), 559–570 (2008)

A Hybrid Particle Swarm Optimization Algorithm for Order Planning Problems of Steel Factory

Tao Zhang[1], Zhifang Shao[1], Yuejie Zhang[2], Zhiwang Yu[1], and Jianlin Jiang[1]

[1] School of Information Management and Engineering, Shanghai University of Finance and Economics, Shanghai 200433, P.R. China
taozhang@mail.shufe.edu.cn
[2] School of Computer Science, Key Laboratory of Intelligent Information Processing, Fudan University, Shanghai 200433, P.R. China
yjzhang@fudan.edu.cn

Abstract. In this paper we construct an integer programming model for the order planning problem. Our model takes into account inventory matching and production planning simultaneously, and considers multiple objectives. We design a hybrid Particle Swarm Optimization, in which new heuristic rules to repair infeasible solutions are proposed, and then compare the results of using PSO, Tabu Search and the hybrid algorithm to solve the models of three different order quantities. Numerical results show that the hybrid PSO/TS algorithm provides more effective solutions.

Keywords: Order planning, inventory matching, particle swarm optimization (PSO), tabu search (TS).

1 Introduction

Order planning is a very important part of the management in steel factory, which is how to schedule machines and procedures in the production line according to due date, type, and quality requirement of the incoming order sequence. The main purpose of order planning is to maximize the utilization of the machines and at the same time to minimize the penalty due to the discrepancy between the actual delivery time and the due time promised in the customer order. Zhang et al [1] proposed a steel factory order planning model based on MTO, and corresponding optimization methods. Liu et al [2] proposed a multi-objective order planning model for steel manufacturing, which includes penalties on delayed orders, machine utilization ratio, inventory costs, etc. Zhang et al [3] formulated the order planning problem as a mixed integer program while considering order due time and production capacity, and designed an efficient genetic algorithm with three mutation operators.

Order planning problems are generally large scale combinatorial optimization problems, which are not easily handled by exact method within a satisfactory amount of time. However, heuristics are very popular for these problems and are the hot spot in recent researches. Among all of the heuristics, Particle Swarm Optimization (PSO) and Tabu Search (TS) algorithms are very popular and are widely applied in real large-scale global optimization problems, especially the scheduling problems in complex production planning. Kennedy and Eberhart [4] applied the improved PSO algorithm in the production scheduling problem which is trying to minimize the production cycles. Nguyen [5] studied the non-stop flow scheduling problem by using PSO. Rezazadeh et al [6] applied PSO in the dynamic machine capacity allocation problem. Liu et al [7] applied TS algorithm in solving the work flow problems in the job shops. Ben-Daya and Al-Fawzan [8] proposed a new neighbourhood generation method while applying TS algorithm in solving work flow scheduling problems. Ganapathy et al [9] proposed a hybrid TS and Simulated Annealing method to solve two-machine work flow problems. Hence, this paper combines these two algorithms to solve order planning problem.

2 Mathematical Models

In steel factories, there exists inventory matching of both finished and unfinished products in [10]. However, in this paper we only consider inventory matching for finished products. Suppose that there are N orders due in the planning horizon $[1,T]$. While making order planning, steel factories usually have three choices: inventory matching, workshop production and order cancellation. We only can choose one of the above three for each order. The decision variables are two sets of binary variable $Y_{i,k}$'s and $X_{i,j,t}$'s denoting inventory matching and production planning respectively, which are shown as follows,

$$Y_{i,k} = \begin{cases} 1, & order\, i\, is\, matched\, by\, inventory\, product\;\; k; \\ 0, & otherwise. \end{cases}$$

$$X_{i,j,t} = \begin{cases} 1, & \mathrm{Pr}\, ocess\;\; j\, of\;\; order\, i\;\; is\, scheduled\, at\;\; period\, t; \\ 0, & Otherwise. \end{cases}$$

where i is the order number, j denotes the number of working procedures, t denotes the time period, k denotes the number of the inventory product. All symbols and parameters are shown as followings.

N :	the total number of the orders;
K :	the number of the kinds of inventory;
J :	the total quantity of the processes;
T :	the planning horizon;
$[a_i, b_i]$:	the delivery time window of order i ;
η_i :	the demand of order i ;
Q_k :	the inventory of product k ;
E_{jt} :	the production capacity of process j in period t ;
v_j :	the penalty coefficient of insufficient utilization of production capacity of process j ;
α_i :	the earliness penalty coefficient of unit weight of order i ;

β_i	:	the delivery time penalty coefficient of unit weight of order i in the time window;
γ_i	:	the tardiness penalty coefficient of unit weight of order i;
p_i	:	order cancellation penalty coefficient of unit weight of order i ;
λ	:	the minimal expected load ratio of each process, a real number within $[0,1]$;
I_{jo}	:	the beginning stock of process j;
$I_{j\max}$	:	the inventory capacity of process j.
$C_{i,k}$	:	the cost of matching order i with inventory k

In order to capture most of the features of steel manufacturing, we are considering multiple objectives in this study, which are shown as follows:

The inventory matching cost:

$$f_1 = \sum_{i=1}^{I}\sum_{k=1}^{K} C_{i,k} Y_{i,k} \tag{1}$$

Penalties of not fully utilizing running machines:

$$f_2 = \sum_{t=1}^{T}\sum_{j=1}^{J} v_j \max\left(\lambda E_{j,t} - \sum_{i=1}^{n} \omega_i X_{i,j,t} \text{ , } 0 \right) \tag{2}$$

Penalties of deliveries within the required time window:

$$f_3 = \sum_{i=1}^{I} \beta_i \omega_i \min\left[\max\left(\sum_{t=1}^{T} t X_{i,J,t} - a_i \text{ , } 0 \right), b_i - a_i \right] \tag{3}$$

Penalties of deliveries outside of the required time window:

$$f_4 = \sum_{i=1}^{I} \omega_i \left[\alpha_i \max\left(a_i - \sum_{t=1}^{T} t X_{t,J,t} \text{ ,} 0 \right) + \gamma_i \max\left(0, \sum_{t=1}^{T} t X_{i,J,t} - b_i \right) \right] \tag{4}$$

Penalty of order cancellation:

$$f_5 = \sum_{i=1}^{I} \eta_i \omega_i \left(1 - \sum_{t=1}^{T} X_{i,J,t} - \sum_{k=1}^{K} Y_{i,k} \right) \tag{5}$$

We combine the above five objectives into one objective function as follows,

$$\text{Minimize} \quad f = \pi_1 f_1 + \pi_2 f_2 + \pi_3 f_3 + \pi_4 (f_4 + f_5) \tag{6}$$

Subject to:

$$\sum_{i=1}^{I} h_{i,k} Y_{i,k} \le Q_k, \qquad k = 1,\ldots,K, \tag{7}$$

$$\sum_{t=1}^{T} X_{i,J,k} + \sum_{k=1}^{K} Y_{i,k} \le 1, \qquad j = 1,\ldots,J; t = 1,\ldots,T, \tag{8}$$

$$\sum_{i=1}^{I} \omega_i X_{i,j,k} \le E_{j,t}, \qquad j = 1,\ldots,J; t = 1,\ldots T, \tag{9}$$

$$\sum_{t=1}^{T} X_{i,j,t} = \sum_{t=1}^{T} X_{i,j+1,t}, \qquad i=1,\dots,I;\ j=1,\dots J-1, \tag{10.1}$$

$$\sum_{t=1}^{T} tX_{i,j,t} \le \sum_{t=1}^{T} tX_{i,j+1,t}, \qquad i=1,\dots,I;\ j=1,\dots J-1, \tag{10.2}$$

$$\sum_{j=1}^{J} X_{i,j,t} \le 2, \qquad i=1,\dots I;t=1,\dots,T, \tag{11}$$

$$s_j^0 + \sum_{i=1}^{I}\left(\sum_{\tau=1}^{t} \omega_i X_{i,j,\tau} - \sum_{\tau=1}^{t} \omega_i X_{i,j+1,\tau} \right) \le s_j^{\max}, \qquad \begin{array}{l} j=1,\dots,J-1; \\ t=1,\dots,T, \end{array} \tag{12}$$

$$X_{i,j,t} \in \{0,1\}, \qquad i=1,\dots I;\ j=1,\dots J;t=1,\dots,T, \tag{13}$$

$$Y_{i,k} \in \{0,1\}, \qquad i=1,\dots I;k=1,\dots,K, \tag{14}$$

Constraint (7) denotes the quantity constraint of each type of surplus products in inventory, where $h_{i,k}$ denotes the quantity of inventory of product k which matches order i. Considering that inventory matching might cause the material lost, $h_{i,k}$ is usually a constant a little greater than order quantity ω_i. Constraint (8) ensures that the same order could not choose both inventory matching and workshop producing at the same time; in addition, every order could be matched by only one kind of inventory product. Constraint (9) is the production capacity limitation constraint of each process. In a unit period the production quantity of one process could not be larger than its production capacity; besides, in order to keep a certain utilization ratio of the equipments, production quantity must be larger than a certain proportion of the production capacity. Constraint (10.1) makes sure that the order will go through all the required processes if it is going to be produced, and constraint (10.2) ensures the sequence of the processes. Constraint (11) ensures that, no more than two production procedures can be performed in any time period. Constraint (12) is the capacity constraint on the inventory of each working procedure.

3 PSO, TS and Hybrid PSO/TS Algorithms for Order Planning

PSO algorithm is first introduced by James Kennedy and Russell Eberhartin in 1995 [4]. Every particle is a vector of m dimensions, which presents a certain solution in the solution space of m dimensions. For example, $X_r = (x_1^r, x_2^r, \dots, x_m^r)$ represents a solution of particle r, and $V_r = (v_1^r, v_2^r, \dots, v_m^r)$ presents the flying speed of particle r, and $\hat{X}_r = (\hat{x}_1^r, \hat{x}_2^r, \dots, \hat{x}_m^r)$ presents the best position that particle r has ever experienced, $\hat{X}$ presents the best position that all the particle have ever experienced. The trajectory of a particle is determined by the evolution equations defined as follows:

$$V_r^{n+1} = \zeta_n V_r^n + \rho_1 \phi_1^n \left(\hat{X}_r^n - X_r^n \right) + \rho_2 \phi_2^n \left(\hat{X}_r^n - X_r^n \right) \tag{15}$$

$$X_r^{n+1} = X_r^n + V_r^{n+1} \tag{16}$$

where $r = 1,2,\dots,R$, and R represents the size of the particle swarm; n represents the iteration, and ξ_n is the inertia coefficient of the iteration n; ρ_1 and ρ_2 are two acceleration constants; ϕ_1^n and ϕ_2^n are random numbers which are normally distributed in range [0,1]. Some studies show that a bigger inertia coefficient is good for global optimum search and smaller inertia coefficient is good for local optimum search as in [11]. So this paper sets a relatively big original inertia coefficient and then attenuates it linearly to improve the local search ability. The linear attenuation formula we adopt is as follows:

$$\zeta_n = \zeta_{\max} - \frac{n}{n_{\max}}(\zeta_{\max} - \zeta_{\min}) \tag{17}$$

where n is the current iteration, and $n_{\max}$ is the maximum number of iterations.

3.1 Encoding Scheme

Because x_{ijt}'s and y_{ik}'s are binary variables, we introduce a integer variable vector composed of c_i 's and $p_{i,j}$'s as follows, $L^r = [c_1^r, c_2^r, \dots, c_N^r;\ p_{1,1}^r, p_{1,2}^r, \dots, p_{1,J}^r;\ p_{2,1}^r, p_{2,2}^r, \dots, p_{2,J}^r; \dots, p_{i,j}^r, \dots; p_{N,1}^r, \dots, p_{N,J}^r]$ to denote the position of particle r. The first part, $[c_1^r, c_2^r, \dots, c_N^r]$, denotes the inventory matching, in which $c_i^r (0 \le r \le M,\ 1 \le i \le N)$ represents the index of surplus products matching order i. If $c_i^r = 0$, there is no surplus inventory matching for order i. $p_{i,j}^r (1 \le j \le J)$ denotes the in-production time period of order i through working procedure j, and if $p_{i,j}^r = 0$ for any $j = 1,2,\dots,J$, it means that order i will not go into production..

3.2 The PSO Algorithm

We design the procedure of hybrid particle swarm optimization and tabu search algorithm as below:

```
Step 1: Initialize all parameters.
Step 2: Every swarm contains $R$ feasible particles randomly, and there are $S$
swarms altogether. Compute the value of objective function (6).
Step 3: For $n = 1$ to
            For $S = 1$ to $S_{\max}$
              For $r = 1$ to $R$
                  Use evolution equations (15) and (16) to move the particles;
                  Repair the infeasible solutions by infeasible solution repairing
                  strategy;
              End For
              Tabu search
                On the basis of the objective value computed by function (6),
              update $\hat{L}^s$ (the best solution of swarm $s$) and $\hat{L}_r^s$ (the historical
              best solution of particle $r$ of swarm $s$)
```

```
        End For
        Update $L_g$ (the globally best solution);
        If $n \bmod 10 = 0$.
            Randomly pair all swarms;
            For each pair, choose 30% of the particles and exchange the values of $L_r^s$'s.
        End If
        Update inertia coefficient according to function (17);
    End For
Step 4: Output $L_g$ as the result.
```

4 Numerical Results

We implement them to a real problem based on the order data of a steel factory. All codes are finished in MATLAB and run on an Intel Core Duo CPU T7300 (2G EMS memory) PC with XP system. We also postulate that the planning horizon $T = 10$ (take five days as a period unit), and number of processes $J = 3$.

Table 1. Comparison of the three algorithms

N	Algorithm	BF	AF	WF	CPU(s)	Iterations
60	PSO	12007.4	13031.7	14129.1	43	463
	TS	12786.4	13504.0	14303.9	17	349
	Hybrid	9786.0	11848.8	13072.8	193	385
140	PSO	12281.2	12509.8	12851.4	104	468
	TS	14722.6	15271.8	17185.0	38	354
	Hybrid	9216.9	9516.7	9773.0	348	419
220	PSO	14680.2	15539.1	16107.9	161	476
	TS	13942.6	18039.1	22140.6	74	328
	Hybrid	12078.0	12206.3	12384.6	563	397

Table 2. Solutions of the three algorithms

N	Algorithm	BF	NoM	NoP	NoE	NoD	NoC
60	PSO	12007.4	21	39	5	3	0
	TS	12786.4	23	37	4	8	0
	Hybrid	9786.0	19	41	3	2	0
140	PSO	12281.2	37	103	4	2	0
	TS	14722.6	51	88	5	13	1
	Hybrid	9216.9	42	98	2	1	0
220	PSO	14680.2	55	163	8	5	2
	TS	13942.6	92	127	8	24	1
	Hybrid	12078.0	51	168	5	3	1

Three sets of numerical experiments with different order size ($N = 60{,}140{,}220$) are performed by using all of the three algorithms, PSO, TS and hybrid PSO/TS algorithms. The computational results are shown in Table 1 and 2, where NoM , NoP, NoE , NoD and NoC denote numbers of matching, production, early finished orders, delayed order and canceled orders respectively.

As we can see from Table 2. Solutions of the three algorithms.

1, PSO algorithm is fast and always gives stable solutions; TS algorithm runs faster but its solutions are usually unstable; the hybrid algorithm is the slowest one and needs more iteration, but its solutions are much better than the others. If time allows, always choose the hybrid PSO/TS algorithm. As shown in Table 2, the NoM , NoP, NoE , NoD and NoC of hybrid PSO/TS algorithm are smaller than the other two for most of the time, which means that the on-time order delivery ratio is improved.

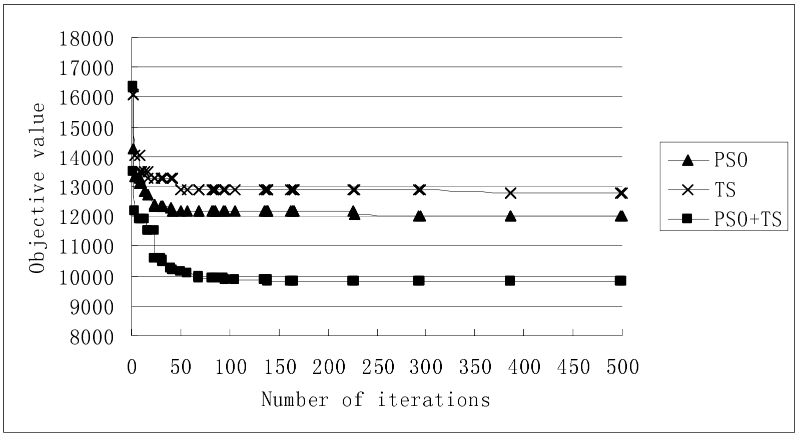


Fig. 1. Objective Value vs. Number of Iteration when $N = 60$

The convergence curves of the three algorithms for $N = 60$, $K = 6$, and $E = 6000$ are shown in Fig 1. PSO algorithm improves the solutions very fast within 50 iterations, and reaches a plateau after that. TS by itself is not a good algorithm since in the whole process it only improves the solutions a little bit. In contrast, the hybrid PSO/TS algorithm keeps improving for more iterations by diversifying particles, and gives much better solutions. Even though the hybrid PSO/TS algorithm is slower than the other two, we would always use it to handle the order planning problems for steel plants since it provides better solutions and its computational time is satisfactory.

5 Conclusions

In this paper we formulate the order planning problem as a mixed integer program, where multiple objectives are considered, such as penalties on delivery discrepancy, inventory costs, penalties on utilization ratio of machines, production costs, matching costs, and penalties on customer order violations. Based on natural number coding scheme, a hybrid PSO/TS algorithm is proposed to solve the order planning problem, where rules to both generate initial solutions and repair infeasible solutions are designed. By simulations, the best set of PSO parameters is determined. Also we run the three algorithms, PSO, TS, and hybrid PSO/TS, on three sets of data with different sizes. The results show that our model is valid for order planning problem, and the hybrid PSO/TS algorithm provides better solutions while being computationally satisfactory.

Acknowledgements

This work is supported by the National Natural Science Fund of China under Grant (No.60773124,70501018), the Natural Science Fund of Shanghai under Grant (No.09ZR1420400, 09ZR1403000, 08ZR1407400), the Open Fund of Shanghai Key Laboratory of Intelligent Information Processing, and 211 Project for Shanghai University of Finance and Economics of China (the 3rd phase Leading Academic Discipline Program). Yuejie Zhang is the corresponding author.

References

1. Zhang, T., Wang, M., Tang, L., Song, J., Yang, J.: The Method for the Order Planning of the Steel Plant based on the MTO Management System. Control and Decision 15, 649–653 (2000)
2. Liu, S., Tang, J., Song, J.: Order-Planning Model and Algorithm for Manufacturing Steel Sheets. International Journal of Production Economics 100, 30–43 (2006)
3. Zhang, T., Zhang, Y., Liu, S.: A Mixed Integer Programming Model and Improved Genetic Algorithm for Order Planning of Iron-Steel Plants. International Journal of Information and Management Science 19, 413–435 (2008)
4. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: IEEE International Conference on Neural Networks, pp. 1942–1948. IEEE Press, Piscataway (1995)
5. Nguyen, V.: A Multiclass Hybrid Production Center in Heavy Traffic. Operations Research 46(3), 13–25 (1998)
6. Rezazadeh, H., Ghazanfari, M., Saidi-Mehrabad, M., Sadjadi, S.J.: An Extended Discrete Particle Swarm Optimization Algorithm for the Dynamic Facility Layout Problem. Journal of Zhejiang University Science A 10, 520–529 (2009)
7. Liu, S.Q., Ong, H.L., Ng, K.M.: A Fast Tabu Search Algorithm for the Group Shop Scheduling Problem. Advances in Engineering Software 36(8), 533–539 (2005)
8. Ben-Daya, M., Al-Fawzan, M.: A Tabu Search Approach for the Flow Shop Scheduling Problem. European Journal of Operational Research 109, 88–95 (1998)
9. Ganapathy, V., Marimuthu, S., Ponnambalam, S.G.: Tabu Search and Simulated Annealing Algorithms for Lot-Streaming in Two-Machine Flow Shop. In: IEEE International Conference on Systems, Man and Cybernetics, pp. 4221–4225. The Hague, Netherlands (2004)
10. Denton, B., Gupta, D., Jawahir, K.: Managing Increasing Products Variety at Integrated Steel Mills. Interfaces 33(2), 41–53 (2003)
11. Trelea, I.C.: The Particle Swarm Optimization Algorithm: Convergence Analysis and Parameter Selection. Information Processing Letters 85, 317–325 (2003)

Hybrid Particle Swarm and Conjugate Gradient Optimization Algorithm

Abdallah Qteish[1] and Mohammad Hamdan[2]

[1] Department of Physics, Yarmouk University, Irbid 21163, Jordan
[2] Department of Computer Science, Yarmouk University, Irbid 21163, Jordan
{aqteish,hamdan}@yu.edu.jo

Abstract. In this work we propose a different particle swarm optimization (PSO) algorithm that employs two key features of the conjugate gradient (CG) method. Namely, adaptive weight factor for each particle and iteration number (calculated as in the CG approach), and periodic restart. Experimental results for four well known test problems have showed the superiority of the new PSO-CG approach, compared with the classical PSO algorithm, in terms of convergence speed and quality of obtained solutions.

1 Introduction

Biologically-inspired algorithms such as evolutionary algorithms has been successful in solving optimization problems. Recently, a new area called swarm intelligence [11] has attracted a lot of attention due to its success in solving optimization and combinatorial problems. The two well known algorithms are Ant Colony Optimization and Particle Swarm Optimization (PSO) [10]. PSO is a population-based algorithm and the agents (particles) interact with each other to adapt to their environment in an iterative manner within a given neighborhood.

Different attempts have been made to improve the canonical PSO algorithm by adding options such as charged swarms [2], neighborhood operator [16], dynamic neighborhood [7], cluster analysis [9] and niching strategies[3]. Other studies looked at strategies for updating particle's velocity such as velocity relaxation [12] and craziness-based [4]. For a comparison of linear and classical velocity update rules see [17].

On the other hand, the conjugate gradient (CG) method [5] is a highly efficient direct minimization approach, which is currently the method of choice in wide areas of science and engineering. In computational solid state physics, for example, the CG method is used to minimize directly the total energy of the system of electrons, which is usually a function of a very large number of variables, in small number of iterations [13]. The key features behind the great success of the CG approach is the conjugacy property of the search directions, and periodic restart of the iterative minimization procedure each certain number of CG steps.

In this work we will point out first the analogy between the PSO and the CG methods, and then employ the above two key features of the CG approach to improve the performance of the PSO. In this hybrid PSO-CG scheme, no explicit calculations of the true gradients are performed, and hence it is still within the spirit of the PSO framework. Therefore, the present PSO-CG approach is different from previously introduced PSO and CG hybridization [8], in which the CG method is invoked together with numerically calculated gradients at certain circumstances (trapping in a local minima or when the absolute difference between successive gradients are larger than a certain value). The proposed PSO-CG algorithm will be tested on four well known test problems.

2 Background

The PSO method is an evolutionary algorithm that simulates the movement of flocks of birds [10]. It can be employed to minimize a general function *F(x)*, where x is a vector in a multidimensional space. In this approach a population of individuals (potential solutions of *F(x)*, called particles) update their movements to reach the target point [the global minima of *F(x)*] by continuously receiving information from other members of the flock. In the classical [1] PSO [15], the n*th* particle velocity and position are updated according to

$$V_{i+1}^n = wV_i^n + c_1 r_1 (P_L^n - x_i^n) + c_2 r_2 (P_g - x_i^n), \tag{1}$$

and

$$x_{i+1}^n = x_i^n + \alpha V_{i+1}^n. \tag{2}$$

Here, w is inertial weight factor, P_L^n is the local best vector of the n*th* particle, and P_g is the global best vector; c_1 and c_2 are adjustable social factors; r_1 and r_2 are random numbers (between 0 and 1); α is the time step.

On the other hand, the conjugate gradient (CG) method is a highly efficient direct minimization approach. In this approach, *F(x)* is minimized by updating the vector x_i according to

$$x_{i+1} = x_i + b_i h_i, \tag{3}$$

where, h_i is the search direction and b_i is chosen such that x_{i+1} is the minimum point of *F(x)* along h_i. In the original Fletcher-Reeves CG approach [5], the search direction h_{i+1} is constructed (except for the first iteration) as

$$h_{i+1} = \beta_i h_i + g_{i+1}. \tag{4}$$

Here, g_i is the gradient of *F(x)* at x_i, given as

$$g_i = -\partial F/\partial x|_{x=x_i}, \tag{5}$$

and β_i is given as

$$\beta_i = \frac{g_{i+1} \cdot g_{i+1}}{g_i \cdot g_i}. \tag{6}$$

[1] The classical PSO adopted in this work is a modification of the original PSO by adding a weight factor for velocity restriction.

The main idea in the CG scheme is to let each search direction be dependent on the previously traversed search directions, which dramatically enhances its efficiency: the learned experience by traversing the previous directions is used in addition to the gradient of F, at the current position, to find an optimal new search direction. In practice, it is found that neglecting the dependence on the quite old search directions increases the efficiency of the CG method. This is usually done by periodically restarting the iterative minimization procedure each certain number of CG steps: setting $h_i = g_i$ as in the first iteration.

3 The Proposed PSO-CG Algorithm

Before introducing our new PSO scheme, we point out the analogy between the CG and PSO methods, briefly described in Section 2. Such an analogy can be easily seen by considering one particle (say the n*th* one) and think of its velocity V_i^n as the search direction h_i and the second term in Eq. (1) as a '*pseudo*'-gradient. In this case the weight factor w is analogous to the parameter β in the CG method, and the time step α corresponds to b_i (both are set to 1 in this work). Thus, what we suggest here, to further improve the efficiency of the classical PSO, is simply to replace the constant weight factor, w, in Eq. 1, by an adaptive factor for each particle and iteration number, w_i^n, calculated in a similar way as β [Eq. (6)]. Moreover, we make use of the idea of restart, as usually done in conjunction with the CG method.

It is worth stressing again that no explicit calculations of the true gradient are performed, and the main purpose of this work is to provide an improved PSO method. Thus, '*pseudo*'-gradients (G_i^n) are used in our proposed PSO-CG approach. One may think that the most logical choice of G_i^n would be the second term of Eq. (1). However, we found that this is not the case, and a much better choice is given as

$$G_i^n = 2(r_1 P_L^n + r_2 P_g) - x_i^n. \tag{7}$$

These *pseudo*'-gradients are used *only* in the calculation of w_i^n and in the case of restart. For the other PSO-CG steps, we use the usual expression (Eq. 1) with w is replaced by w_i^n.

In the standard CG method, the magnitudes of the true gradients reduce after each step and hence β is expected to be smaller than one. In our proposed PSO-CG method, this may not be the case and it takes several steps to reach a significant reduction in magnitude of the pseudo-gradients. Thus, for the calculation of w_i^n we used the formula

$$w_i^n = \frac{G_{i+1}^n . G_{i+1}^n}{G_{i-j}^n . G_{i-j}^n}. \tag{8}$$

where j can be considered as an adjustable parameter. The other new adjustable parameter is the number of steps (M) after which the CG steps are restarted.

4 Experimental Setup, Results and Discussion

To evaluate the proposed method, we used four well known test problems: one unimodal (Rosenbrock) and three multimodal (Ackley, Griewank and Rastrigin) [1]. In our present experiments the number of variables (dimension the vector x) is set to 30, and the number of particles to 20. We use asymmetric initialization as suggested by [6]. In the classical PSO runs, the constant inertial weight factor is set to 0.72, and in both PSO and PSO-CG we used $c_1 = c_2 = 1.49$. The maximum number of iterations is set to 2000, for the four test problems. The results were averaged over 30 independent runs. These parameters are as those used in similar studies, see for example [1].

We have performed several tests to determine the optimal values for j, in Eq. 7, and the parameter M of the restart procedure, see section 3. We have found the $j = 4$ is a very good value for the four considered test problems. As for M, the reasonable values in the PSO and PSO-CG methods are of 10 and 7 for Ackley, 5 and 3 for Rosenbrock, 25 and 3 for Rastrigin, and 7 and 3 for Griewank. These values for both j and M are used throughout this paper. More tests are required to suggest a reasonable value of M in the PSO-CG scheme for a wide range of problems.

To test the effects of both new features (adaptive weight factor and restart) we performed experiments for the following four configurations: (i) classical PSO with fixed w and no restart, denoted by PSO-NR. (ii) classical PSO with fixed w and restart, denoted by PSO-RM (where M takes the values described above). (iii) PSO-CG with adaptive w and no restart, denoted by PSO-CG-NR. (v) PSO-CG with adaptive w and restart, denoted by PSO-CG-RM. The resulting averaged best fitness as a function of iteration number are shown in Fig. 1, for the four test problems considered. The final averaged best fitness obtained after 2000 iterations are listed in Table I, compared with the hybrid Genetic Algorithm (GA) and PSO approach [14]. It is clear that our approach outperforms the GA-PSO one.

Let us start our discussion with the effects of restart. Fig. 1 and Table 1 show that the it leads to a significant improvement in both the classical PSO and PSO-CG approaches, and that the reasonable value of M depends on the rate of convergence of the adopted method. This can be understood as follows. The velocity (or the search direction in the CG method) gets better with increasing the iteration number. Hence keeping the dependence on old velocities, which are quite far from the current one, is counter productive. Such a dependence is removed by restart. This explains both the improvement of the classical PSO and PSO-CG algorithms by the restart procedure, and the dependence of M on the rate of convergence.

The high efficiency of CG method when compared to the steepest descent method (considering only the gradient as the search direction) is due to the conjugacy property of the search directions. In the PSO method, the inclusion of the previous velocity in Eq. (1) has somehow similar effect: the classical PSO is more efficient than the original one. Thus, allowing w to be adaptive and calculated in a similar way as β in the CG method improves the conjugacy

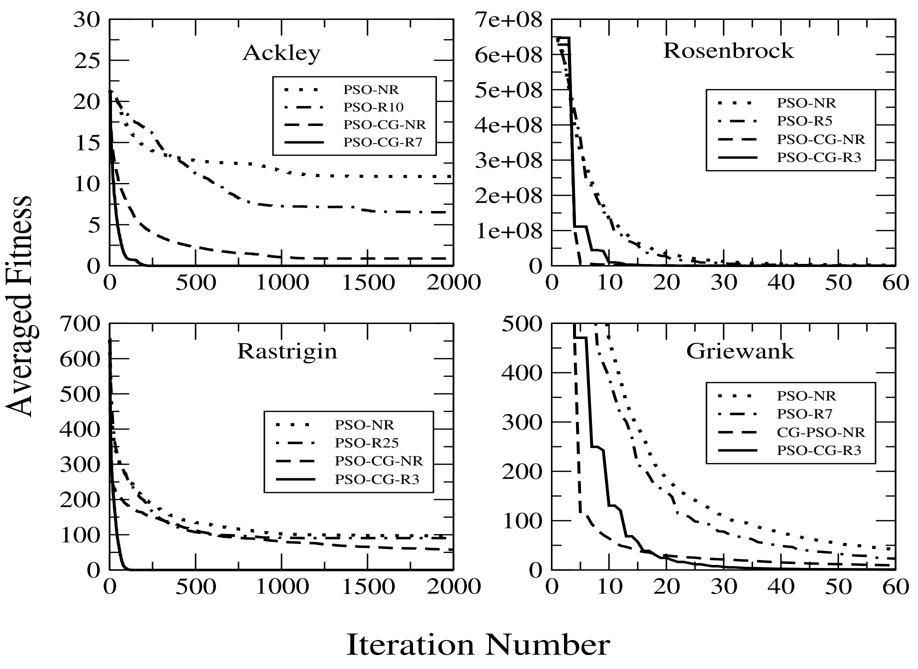


Fig. 1. The average best fitness as a function of iteration number for the four PSO algorithms

Table 1. Summary of mean best value and standard deviation in the last iteration. NC means not computed. WR means with restart [using the reasonable values of M]. The best is in bold.

Problem	PSO-NR	PSO-WR	PSO-CG-NR	PSO-CG-WR	GA-PSO
	Mean Best	Mean Best	Mean Best	Mean Best	Mean Best
	(stdv)	(stdv)	(stdv)	(stdv)	(stdv)
Ackley	10.870	6.518	0.888	**0.388E-14**	NC
	(9.900)	(8.56)	(3.752)	(0.638E-15)	NC
Rosenbrock	103.152	61.241	76.478	**27.197**	28.996
	(154.501)	(64.100)	(30.762)	(0.619)	(0.003)
Rastrigin	97.380	90.508	0.0	**0.0**	0.0
	(30.34)	(32.494)	(0.0)	(0.0)	(0.0)
Griewank	0.021	0.018	0.0	**0.0**	0.005
	(0.027)	(0.023)	(0.0)	(0.0)	(0.001)

of the velocities in the PSO method, which explains the large improvement in performance by going from PSO to PSO-CG algorithms.

5 Summary and Further Work

In this paper, we have pointed out the analogy between the classical PSO and the conjugate gradient methods. This has allowed us to introduce an improved PSO algorithm by employing two key features of the CG method: adaptive weight factors calculated as in the CG method and restart. Preliminary results obtained for four well known test problems have demonstrated the highly improved performance of the proposed PSO-CG algorithm. Further tests, analysis and deeper understanding of proposed method as will as a detailed comparison with the state-of-art standard PSO method are currently under consideration.

Acknowledgment. The work in this paper has been funded by Ministry of Higher Education-Jordan, project number TH/1/01/2008.

References

1. Angeline, P.: Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences. In: Porto, V.W., Waagen, D. (eds.) EP 1998. LNCS, vol. 1447, pp. 601–610. Springer, Heidelberg (1998)
2. Blackwell, T., Bentley, P.: Dynamic search with charged swarms. In: Genetic and Evolutionary Computation Conference 2002 (GECCO), pp. 19–26 (2002)
3. Brits, R.: Niching strategies for particle swarm optimization. Master's thesis, University of Pretoria, Pretoria (2002)
4. Chatterjee, A., Mukherjee, V., Ghoshal, S.: Velocity relaxed and craziness-based swarm optimized intelligent pid and pss controlled avr system. International Journal of Electrical Power and Energy Systems 31(7–8), 323–333 (2009)
5. Fletcher, R., Reeves, C.: Computer Journal 7, 149 (1964)
6. Fogel, D., Beyer, H.G.: A note on the empirical evaluation of intermediate recombination. Evolutionary Computation 3(4), 491–495 (1996)
7. Hu, X., Eberhart, R.: Multiobjective optimization using dynamic neighborhood particle swarm optimisation. In: The IEEE Congress on Evolutionary Computation (CEC), pp. 1677–1687 (2002)
8. Kawakami, K., Meng, A.: Improvement of particle swarm optimization. PIERS ONLINE 5(3), 261–266 (2009)
9. Kennedy, J.: Stereotyping: improving particle swarm performance with cluster analysis. In: The IEEE Congress on Evolutionary Computation (CEC), pp. 1507–1512 (2000)
10. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proc. of the IEEE Int. Conf. on Neural Networks, Piscataway, NJ, USA, pp. 1942–1948 (1995)
11. Kennedy, J., Eberhart, R.: Swarm Intelligence. Morgan Kaufmann Publishers, San Francisco (2001)
12. Liu, Y., Qin, Z., Xu, Z.L., He, X.S.: Using relaxation velocity update strategy to improve particle swarm optimization. In: Proceedings of 2004 International Conference on Machine Learning and Cybernetics, vol. 4, pp. 2469–2472 (2004)

13. Payne, H., Teter, M., Allna, D.: Rev. Mod. Phys., pp. 1045–1097 (1992)
14. Settles, M., Soule, T.: Breeding swarm: A ga/pso hybrid. In: Genetic and Evolutionary Computation Conference 2005 (GECCO), Washington, USA, pp. 161–168 (2005)
15. Shi, Y., Eberhart, R.: A modified particle swarm optimizer. In: Proceedings of the IEEE International Conference on Evolutionary Computation, pp. 67–73 (1998)
16. Suganthan, P.: Particle swarm optimiser with neighbourhood operator. In: The IEEE Congress on Evolutionary Computation (CEC), pp. 1958–1962 (1999)
17. Wilke, D., Kok, S., Groenwold, A.: Using relaxation velocity update strategy to improve particle swarm optimization. International journal for numerical methods in engineering 70 (2007)

A Hybrid of Particle Swarm Optimization and Local Search for Multimodal Functions

Jin Qin[1,2], Yixin Yin[1], and Xiaojuan Ban[1]

[1] College of Information Engineering, University of Science & Technology Beijing, 100083 Beijing, China
[2] College of Computer Science & Technology, Guizhou University, 550025 Guiyang, China
cse.jqin@gzu.edu.cn

Abstract. The standard PSO has problems with consistently converging to good solutions, especially for multimodal functions. The reason for PSO failing to find (global) optima is premature convergence. Also, it has been shown in many empirical studies that PSO algorithms lack exploitation abilities. In this paper, we propose a hybrid of particle swarm optimization and local search, in which a standard PSO algorithm incorporates a local search algorithm. The standard PSO algorithm and the local search algorithm are devoted to exploration and exploitation of solution space, respectively. Particle's current position is updated using update equation of standard PSO and then is refined by local search algorithm. The introduction of a local search improves the capability of exploitation of local region of standard PSO and prevents from premature convergence. The hybrid algorithm can locate multiple solutions without use of specific niching techniques. The hybrid algorithm showed superior performance on a set of multimodal functions.

Keywords: Particle swarm optimization, Local Search, Exploration, Exploitation, Multimodal function.

1 Introduction

Particle swarm optimization (PSO) is a population-based search algorithm, originated from the simulation of the social behavior of birds within a flock. Since particle swarm optimization as a method for optimization of continuous nonlinear problem was introduced by James Kennedy and Russell Eberhart in 1995[1][2], it has been used across a wide range of applications, such as image and video analysis, design and restructuring of electricity networks, control, antenna design, electronics and electromagnetics, and so on [3].

The original PSO, however, has problems with consistently converging to good solutions, especially for multimodal functions. A number of techniques have been developed to improve quality of solution to multimodal functions, including introduction of new social topologies, e.g. Mendes's fully informed particle swarm (FIPS)[4], Suganthan's dynamically increasing neighborhood[5]; sub-swarm strategy by which multiple sub-swarms behave cooperatively, e.g. Clerc's parameter-free particle swarm system called TRIBES[6], Seo's multi-grouped particle swarm optimization (MGPSO)[7];

repelling method by which one particle is repelled from another in order to avoid collision, e.g. Blackwell's collision-avoiding swarms[8]; hybrid algorithms, e.g. Kao's hybrid genetic algorithm and particle swarm optimization (GA-PSO)[9].

In this paper, we propose another hybrid PSO algorithm incorporating a local search. Two features distinguish our hybrid PSO algorithm from other hybrids of PSO. Firstly, PSO and local search are devoted to different aspects of search i.e. exploration and exploitation, respectively. Secondly, local search focuses on exploitation of a region which is centered at the current position of a particle, not its personal best position.

2 Standard Particle Swarm Optimization

Let $S = \{1, 2, ..., N\}$ denote the swarm of particles. At any time, t, the position of a particle, $i (i \in S)$, is denoted as vector in n dimensions, $\mathbf{x}_i(t)$. In the next time step, $t+1$, particle i will move to the position

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1) \tag{1}$$

where $\mathbf{v}_i(t+1)$ is the velocity of particle i at time step $t+1$. The velocity of particle i is updated as follow:

$$\mathbf{v}_i(t+1) = \mathbf{v}_i(t) + c_1 r_1 (\mathbf{p}_i(t) - \mathbf{x}_i(t)) + c_2 r_2 (\mathbf{g}_i(t) - \mathbf{x}_i(t)) \tag{2}$$

where $\mathbf{p}_i(t)$ is the personal best position of particle i which it has visited with the best fitness before, $\mathbf{g}_i(t)$ is the best position with the best fitness found by any of particle in i's topological neighborhood with respect to some social network, c_1 and c_2 are positive acceleration constants, and $r_1, r_2 \sim U(0,1)$ are random values in the range $[0,1]$, sampled from a uniform distribution. Without loss of generality, considering minimization problems, $\mathbf{p}_i(t)$ is calculated as

$$\mathbf{p}_i(t) = \arg\min_{\mathbf{x}_i(s)} \{ f(\mathbf{x}_i(s)) \mid 0 \le s \le t \} \tag{3}$$

where $f : R^n \to R$ is a fitness function.

And $\mathbf{g}_i(t)$ is calculated as

$$\mathbf{g}_i(t) = \arg\min_{\mathbf{p}_j(t)} \{ f(\mathbf{p}_j(t)) \mid j \in Neighbor(i) \subset S \} \tag{4}$$

where $Neighbor(i)$ is i's topological neighborhood with respect to some social network.

In the original version of PSO, each component of $\mathbf{v}_i$ is kept within the range $[-v_{\max}, +v_{\max}]$ to prevent particle speeds out of control. The use of hard bounds on velocity, however, presents some problems since the optimal value of $v_{\max}$ is problem-specific. One of the most widely used improvements is the introduction of inertia weight by Shi and Eberhart [10][11], which is employed to better control the scope of search, reduce the importance of $v_{\max}$, and perhaps eliminate it altogether. Incorporating the new parameter, the update equation of velocity is rewritten as

$$\mathbf{v}_i(t+1) = w\mathbf{v}_i(t) + c_1 r_1(\mathbf{p}_i(t) - \mathbf{x}_i(t)) + c_2 r_2(\mathbf{g}_i(t) - \mathbf{x}_i(t)) \tag{5}$$

where w is the so-called inertia weight.

Another method of balancing global exploration and local exploitation was proposed by Clerc and Kennedy [12]. Similar to the inertia weight method, this method introduced a new parameter χ, known as the constriction coefficient. The new parameter is defined as

$$\chi = \frac{2}{\phi - 2 + \sqrt{\phi^2 - 4\phi}} \tag{6}$$

where $\phi = c_1 + c_2 > 4$.

The constriction coefficient is applied to the entire update equation of velocity:

$$\mathbf{v}_i(t+1) = \chi(\mathbf{v}_i(t) + c_1 r_1(\mathbf{p}_i(t) - \mathbf{x}_i(t)) + c_2 r_2(\mathbf{g}_i(t) - \mathbf{x}_i(t))) \tag{7}$$

When the constriction method is used, ϕ is commonly set to 4.1, $c_1 = c_2 = 2.05$ and $\chi \approx 0.72984$. This becomes the so-called canonical/ standard particle swarm optimization [13].

3 Hybrid of Particle Swarm Optimization and Local Search

The reason for PSO failing to find (global) optima is premature convergence. Formal analyses of particle trajectories proved that each particle converges to a stable point, which is a weighted average of the particle's personal best and neighborhood best positions. However, the point of stagnation may not necessarily coincide with a local minimum.

Also, it has been shown in many empirical studies that PSO algorithms lack exploitation abilities. PSO algorithms have the ability to quick drill down to good, promising regions of the search space, but lack the ability to refine solutions.

The standard PSO is driven by own experience of individual particle which leads it to return to the place that most satisfied it in the past and social interaction with other particles by which individual particle learns from the best neighbor, in other words, it moves close to the best neighbor and to the best itself.

An observation of particles' behavior is that a particle which positions at $\mathbf{x}(t)$ at time step t will directly move to $\mathbf{x}(t+1)$ at next time step. Intuitively, the behavior of a particle is so discontinuous. A particle leaves a region in which it arrives just a moment ago without staying for a while to exploit the region. As a result, two disadvantages are introduced. On the one hand, the particle may miss a better solution; on the other hand, the swarm may gather together so fast that it converges prematurely.

Based on above observation, we propose a hybrid PSO algorithm. The basic idea is described as follows. A particle stops to exploit the new region for a period of time whenever it reaches a new position. A particle exploits a new arriving region through a local search. In our hybrid algorithm, a local search focuses on exploitation of the region visited by a particle and the movement of a particle aims at exploration of promising region. In this way, particles that reach different regions of solution space may find different local optima through a local search, simultaneously. This is very important for optimization of multimodal functions. Our hybrid algorithm is depicted as Algorithm 1.

```
Algorithm 1. Hybrid algorithm of PSO and local search
FOR each particle $i \in S$ DO
  Initialize $\mathbf{x}_i$, $\mathbf{v}_i$;
FOR each particle $i \in S$ DO
     $\mathbf{p}_i \leftarrow \mathbf{x}_i$;
     $y_i \leftarrow f(\mathbf{p}_i)$;
REPEAT
     FOR each particle $i \in S$ DO
            Determine $\mathbf{g}_i$ with respect to given topology;
            $\mathbf{v}_i \leftarrow \chi(\mathbf{v}_i + c_1 r_1(\mathbf{p}_i - \mathbf{x}_i) + c_2 r_2(\mathbf{g}_i - \mathbf{x}_i))$;
            $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$;
            $\mathbf{x}'_i \leftarrow localsearch(\mathbf{x}_i)$;
            IF $f(\mathbf{x}'_i) < y_i$ THEN
                   $\mathbf{p}_i \leftarrow \mathbf{x}'_i$;
                   $y_i \leftarrow f(\mathbf{x}'_i)$;
UNTIL stopping condition is true;
```

Here, local search can be any trajectory method, such as steepest descent, Rosenbrock, Nelder-Mead, Fletcher-Reeves, etc. In the implementation of our hybrid algorithm, we adopt a simplified steepest descent algorithm, depicted in Algorithm 2.

```
Algorithm 2. Simplified Steepest Descent Algorithm
$\mathbf{x}' \leftarrow \mathbf{x}$;
$y \leftarrow f(\mathbf{x}')$;
FOR $i = 1, 2, ..., n_{iter}$ DO
  $\mathbf{x}_{new} \leftarrow \mathbf{x}' - \lambda \nabla f(\mathbf{x}')$;
  $y_{new} \leftarrow f(\mathbf{x}_{new})$;
  IF $y_{new} < y$ THEN
     $\mathbf{x}' \leftarrow \mathbf{x}_{new}$;
     $y \leftarrow y_{new}$;
  ELSE
     $\lambda \leftarrow \lambda / 2$;
```

In Algorithm 2, there are two parameters which are required setting before the algorithm is executed: the number of iterations, n_{iter} and the step size, λ. In the implementation of our algorithm, let $\lambda = (upper - lower) / n_{iter}$, where *upper* and *lower* are the upper bound and lower bound of values of variables in a given function, respectively. As a result, the algorithm becomes a one-parameter one.

4 Experiment and Results

We compared our hybrid algorithm (hPSO) with Passaro's kPSO. We adopted the set of benchmark functions used in [14] plus two high-dimensional functions ($n = 30$). These functions are listed in Table 1. Global optima and optimal solution of each function are presented in Table 2.

Table 1. Benchmark functions

Name	Function	Domain
Branin RCOS	$f_1(x_1,x_2) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1-\frac{1}{8\pi})\cos(x_1) + 10$	$x_1 \in [-5,10]$ $x_2 \in [0,15]$
Six-hump camel-back	$f_2(x_1,x_2) = -4(4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4)$	$x_1 \in [-1.9,1.9]$ $x_2 \in [-1.1,1.1]$
Deb's 1st function	$f_3(x) = \sin^6(5\pi x)$	$x \in [0,1]$
Himmelblau	$f_4(x_1,x_2) = 200 - (x_1^2 + x_2 - 11)^2 - (x_1 + x_2^2 - 7)^2$	$x_1, x_2 \in [-6,6]$
Shubert 2D	$f_5(x_1,x_2) = \sum_{i=1}^{5} i\cos((i+1)x_1 + i)\sum_{i=1}^{5} i\cos((i+1)x_2 + i)$	$x_1, x_2 \in [-10,10]$
Griewank	$f_6(\mathbf{x}) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1$	$x_i \in [-600,600]$
Ackley	$f_7(\mathbf{x}) = -20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - e^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)} + 20 + e$	$x_i \in [-30,30]$

The values of parameters in our algorithm are set as in Table 3. We also adopted the same experimental settings used in [14]. 50 independent simulation runs on each function are conducted for both algorithms. Location of the global optima with an accuracy of $\varepsilon = 0.00001$ served as the criterion of termination. In Table 4, we report the number of function evaluations required to locate the global optima. For our algorithm, the total number of evaluations is the number of objective function evaluations plus the number of gradient evaluations.

Table 2. Global optima and optimal solutions for all functions

Function	Optimum	Solution	Description
f_1	$f_1(x_1,x_2) = 0.397887$	$(-\pi,12.275),(\pi,2.275),(9.42478,2.475)$	3 global minima
f_2	$f_2(x_1,x_2) = 4.126514$	$(0.08983,-0.7126),(-0.08983,0.7126)$	2 global maxima, several deceptive local ones

Table 2. (*Continued*)

f_3	$f_3(x)=1$	$0.1, 0.3, 0.5, 0.7, 0.9$	5 equally spaced global maxima
f_4	$f_4(x_1,x_2)=200$	$(3.0,2.0),(-2.80512,3.13131),$ $(-3.77931,-3.28319),(3.58443,-1.84813)$	4 global maxima
f_5	$f_5(x_1,x_2)=-186.730908$	$(-7.08350,-7.70831),(5.48286,4.85805),\ldots$	18 global minima surrounded by 760 local minima
f_6	$f_6(x_1,x_2,\ldots,x_n)=0$	$(0,0,\ldots,0)$	several local minima, one global minimum
f_7	$f_7(x_1,x_2,\ldots,x_n)=0$	$(0,0,\ldots,0)$	numerous local minima, one global minimum

Table 3. Parameters setting of hPSO

Parameter	χ	c_1	c_2	n_{iter}
Value	0.72984	2.05	2.05	20

Table 4. Average and standard deviation of number of evaluations

Functions	Swarm size	Number of evaluations	
		kPSO	hPSO
f_1	30	**2084±440**	2786±2092
	60	**3688±717**	4192±2052
f_2	30	**1124±216**	1727±43
	60	**2127±341**	3479±49
f_3	30	1207±688	**996±24**
	60	**1654±705**	1986±31
f_4	30	2259±539	**1532±7**
	60	3713±570	**3065±9**
f_5	30	81194±45646	**1330±541**
	60	117503±77451	**2506±15**
f_6	30	—	2029969±277461
f_7	30	—	1287320±151752

5 Observation and Discussion

From the computational results presented in Table 4, we can observe that hPSO showed a competitive performance over kPSO for the former four benchmarks in terms of computational time. For the fifth benchmark, hPSO significantly outperforms kPSO in terms of computational time. For the last two high-dimensional functions, hPSO also found the global minima. Additionally, kPSO has extra computational cost of clustering procedure.

For a deep observation of the process that hPSO finds out the global optima of a give function, several snapshots of hPSO running on the Branin RCOS function are plotted in Figure 1. We can observe that hPSO located all three global minima after several thousand function evaluations. Although hPSO has not incorporated a mechanism of clustering, it also showed superior performance in optimization of multimodal functions. The inherent reason for this is the separation of exploration and exploitation in hPSO. In hPSO, standard PSO algorithm focuses on searching for promising regions of solution space whereas local search algorithm focuses on finding out local optima. Moreover, particle's current position is updated using update equation of PSO and then is refined by local search algorithm. In this way, hPSO prevents from premature convergence and meanwhile locates multiple global/local optima.

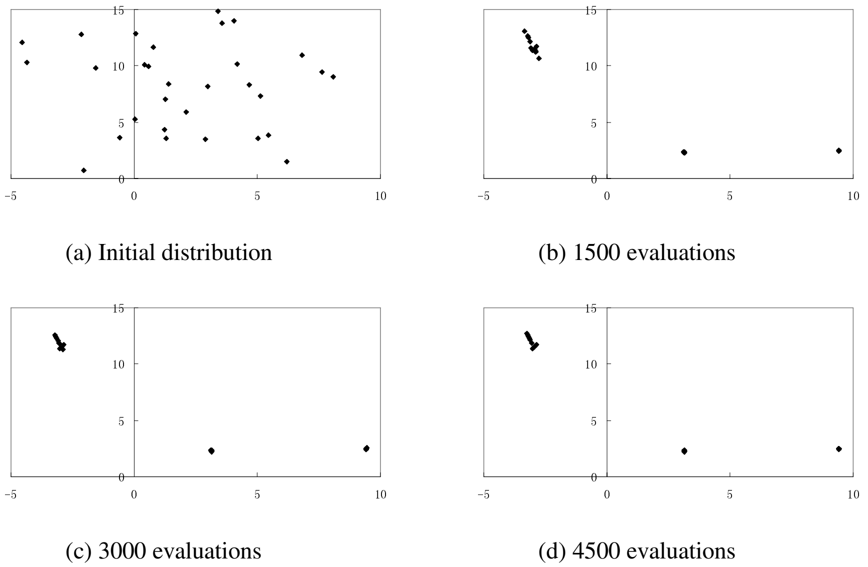


Fig. 1. Distribution of particles' personal best positions at different time steps

6 Conclusion

In this paper, we proposed a hybrid particle swarm optimization method, called hPSO. The basic idea of hPSO is the cooperation of PSO algorithm and local search algorithm. A standard PSO algorithm and a local search algorithm are devoted to exploration and exploitation of solution space, respectively. The introduction of a local search improves the capability of exploitation of local region of standard PSO and prevents from premature convergence.

Experiments on a set of seven benchmark functions were conducted. The computational results show that hPSO can locate multiple solutions although it doesn't adopt any specific niching techniques. The hPSO is suited for optimization of multimodal functions.

For local search, we have just considered a simplified steepest descent algorithm. Further research will investigate the introduction of other local search algorithms, such as Rosenbrock, Nelder-Mead, Fletcher-Reeves, etc.

References

1. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: The 1995 IEEE International Conference on Neural Networks, Perth, Australia, vol. IV, pp. 1942–1948. IEEE Service Center, Piscataway (1995)
2. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: The sixth international symposium on micro machine and human science, Nagoya, Japan, pp. 39–43. IEEE, Piscataway (1995)
3. Poli, R.: Analysis of the Publications on the Applications of Particle Swarm Optimisation. Journal of Artificial Evolution and Applications, Article ID 685175, 2008, 10 pages (2008)
4. Mendes, R., Kennedy, J., Neves, J.: Watch thy neighbor or how the swarm can learn from its environment. In: The IEEE swarm intelligence symposium (SIS), pp. 88–94. IEEE, Piscataway (2003)
5. Suganthan, P.N.: Particle swarm optimiser with neighbourhood operator. In: The IEEE congress on evolutionary computation (CEC), pp. 1958–1962. IEEE, Piscataway (1999)
6. Clerc, M.: Particle swarm optimization. ISTE, London (2006)
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(4), 1095–1098 (2006)
8. Blackwell, T.M., Bentley, P.J.: Don't Push Me! Collision-Avoiding Swarms. In: The IEEE Congress on Evolutionary Computation (CEC), pp. 1691–1696 (2002)
9. Kao, Y.T., Zahara, E.: A hybrid genetic algorithm and particle swarm optimization for multimodal functions. Applied Soft Computing 8, 849–857 (2008)
10. Shi, Y., Eberhart, R.C.: A modified particle swarm optimizer. In: The IEEE international conference on evolutionary computation, pp. 69–73. IEEE, Piscataway (1998)
11. Shi, Y., Eberhart, R.C.: Parameter selection in particle swarm optimization. In: Porto, V.W., Waagen, D. (eds.) EP 1998. LNCS, vol. 1447, pp. 591–600. Springer, Heidelberg (1998)
12. 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)
13. Bratton, D., Kennedy, J.: Defining a Standard for Particle Swarm Optimization. In: The 2007 IEEE Swarm Intelligence Symposium, pp. 120–127 (2007)
14. Passaro, A., Starita, A.: Particle Swarm Optimization for Multimodal Functions: A Clustering Approach. Journal of Artificial Evolution and Applications, Article ID 482032, 2008, 15 pages (2008)

A Cooperative Ant Colony System and Genetic Algorithm for TSPs

Gaifang Dong[1] and William W. Guo[2]

[1] College of Computer and Information Engineering
Inner Mongolia Agricultural University, Hohhot, China
[2] Faculty of Arts, Business, Informatics and Education, Central Queensland University
North Rockhampton QLD 4702, Australia
w.guo@cqu.edu.au

Abstract. The travelling salesman problem (TSP) is a classic problem of combinatorial optimization and is unlikely to find an efficient algorithm for solving TSPs directly. In the last two decades, ant colony optimization (ACO) has been successfully used to solve TSPs and their associated applicable problems. Despite the success, ACO algorithms have been facing constantly challenges for improving the slow convergence and avoiding stagnation at the local optima. In this paper, we propose a new hybrid algorithm, cooperative ant colony system and genetic algorithm (CoACSGA) to deal with these problems. Unlike the previous studies that regarded GA as a sequential part of the whole searching process and only used the result from GA as the input to the subsequent ACO iteration, this new approach combines both GA and ACS together in a cooperative and concurrent fashion to improve the performance of ACO for solving TSPs. The mutual information exchange between ACS and GA at the end of each iteration ensures the selection of the best solution for the next round, which accelerates the convergence. The cooperative approach also creates a better chance for reaching the global optimal solution because the independent running of GA will maintain a high level of diversity in producing next generation of solutions. Compared with the results of other algorithms, our simulation demonstrates that CoACSGA is superior to other ACO related algorithms in terms of convergence, quality of solution, and consistency of achieving the global optimal solution, particularly for small-size TSPs.

Keywords: Ant colony optimization, Ant colony system, Genetic algorithm, Traveling salesman problem, Convergence, Consistency.

1 Introduction

The traveling salesman problem (TSP) is a classic problem in combinatorial optimization research that can be described as follows: given a list of cities and their pair-wise distances, the task of TSP is to find the shortest possible tour that visits each city once and returns to its original city [1]. In computational complexity, TSP belongs to the class of *NP*-complete problems, and thus it is unlikely to find an efficient algorithm for solving TSPs directly [2]. As a result, various heuristics and approximation algorithms have been devised to quickly produce usable solutions for TSPs [3,4].

In the last two decades, many techniques of modern computational intelligence have been successfully adopted as effective tools to produce acceptable solutions for TSPs, such as neural networks [5,6], simulated annealing (SA) [7], genetic algorithm (GA) [8,9], ant colony optimization (ACO) [10–13], and their modifications and/or combinations in various ways [14-23].

ACO is a multi-agent system inspired by the foraging behavior of some ant species. These ants deposit pheromone on the ground in order to mark some favorable path that can be followed by other members of the colony. The shortest path will be eventually taken by all ants through which food can be transported from the food source to the nest efficiently. In ACO, a number of artificial ants build solutions to the considered optimization problem and exchange information on the quality of these solutions via a communication scheme similar to the one adopted by real ants. Although different ACO algorithms have been proposed and produced satisfactory outcomes in solving various TSPs, these algorithms face constantly challenges for improving the slow convergence and avoiding stagnation at the local optima.

This paper takes on these constant battles by combining both GA and ACS together in a cooperative fashion to improve the performance of ACS for solving TSPs. Unlike the previous studies that regarded GA as a sequential part of the whole searching process and only used the selected result from GA as the input to the subsequent ACO iteration or vice versa, our approach is to run both GA and ACS concurrently and mutual information exchange in terms of choosing the best solution is made after each iteration, which accelerates the convergence. This cooperative approach also creates a better chance for reaching the global optimal solution because the independent running of GA provides the assurance of diversity in population (or solutions) during the entire process.

In the remaining of this paper, we firstly describe some widely used ACO algorithms for TSPs, which should point out where our new approach differs from the others. Our proposed algorithm, cooperative ant colony system and genetic algorithm (CoACSGA), will then be described, and followed by the presentation of the results of using this new algorithm for solving some selected benchmarking cases of TSP. Based on these results, comparisons are then made with the outcomes of other GA/ACO algorithms for TSPs. Conclusions are drawn at the last.

2 The Main Ant Algorithms for TSPs

Ant System (AS) is the first ACO algorithm proposed in the literature [10,11]. Its main characteristic is that at each iteration the pheromone values are updated by all the m ants that have built a solution at the end of that iteration. The pheromone τ_{ij}, associated with the edge joining cities i and j, is updated by.

$$\tau_{ij} \leftarrow (1-\rho)\cdot\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} , \tag{1}$$

where ρ is the evaporation rate, and $\Delta\tau_{ij}^{k}$ is the quantity of pheromone laid on edge (i, j) by ant k and expressed as

$$\Delta\tau_{ij}^{k} = \begin{cases} Q / L_k & \textit{if ant k visited edge (i, j) in its tour,} \\ 0 & \textit{otherwise,} \end{cases} \tag{2}$$

where Q is a constant, and L_k is the length of the tour constructed by ant k.

In the construction of a solution, ants select the next city to be visited through a stochastic mechanism. The probability for ant k in city i to move to city j next is given by:

$$p_{ij}^{k} = \begin{cases} \dfrac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum\limits_{l \in S_k} \tau_{il}^{\alpha} \cdot \eta_{il}^{\beta}} & \textit{if } j \in S_k, \\ 0 & \textit{otherwise,} \end{cases} \tag{3}$$

where S_k is the intersection of the candidate list of city i and the set of cities that ant k has not visited yet; α and β control the relative importance of the pheromone versus the heuristic information η_{ij}, which depends on the distance between cities i and j d_{ij}

$$\eta_{ij} = \frac{1}{d_{ij}}, \tag{4}$$

Max-Min Ant System (MMAS) [12,13] is an improvement over the original AS. It only allows the best ant to update the pheromone trails after each iteration and the value of the pheromone is bound between τ_{min} and τ_{max}. This pheromone update can be expressed as

$$\tau_{ij} \leftarrow \left[(1-\rho)\cdot\tau_{ij} + \Delta\tau_{ij}^{best}\right]_{\tau_{\min}}^{\tau_{\max}}, \tag{5}$$

where $\Delta\tau_{ij}^{best}$ is defined as

$$\Delta\tau_{ij}^{best} = \begin{cases} 1 / L_{best} & \textit{if (i, j) is in the best tour,} \\ 0 & \textit{otherwise,} \end{cases} \tag{6}$$

where L_{best} is the length of the tour of the best ant. This may be either the best tour found in the current iteration, or the best solution found since the start of the algorithm, or a combination of both.

Ant Colony System (ACS) [11] introduces a local pheromone update into AS/MMAS, in addition to the pheromone update performed at the end of each construction process. Each ant applies it only to the last edge traversed through

$$\tau_{ij} \leftarrow (1-\rho)\cdot\tau_{ij} + \varphi\cdot\tau_0, \tag{7}$$

where $\varphi \in (0,\ 1]$ is the pheromone decay coefficient, and τ_0 is the initial value of the pheromone.

The main goal of the local update is to diversify the search performed by subsequent ants during an iteration, i.e., by decreasing the pheromone concentration on the traversed edges, subsequent ants are encouraged to choose other edges and, hence, to produce different solutions. This reduces the likelihood for several ants to produce identical solutions during each iteration.

The local pheromone update is defined as

$$\tau_{ij} = \begin{cases} (1-\rho)\cdot\tau_{ij} + \rho\cdot\Delta\tau_{ij}^{best} & \textit{if } (i, j) \textit{ in in the best tour,} \\ \tau_{ij} & \textit{otherwise.} \end{cases} \tag{8}$$

In addition to the differences in the pheromone update procedure, ACS also uses a different transition rule [17] for ant *k* in city *i* to select its next city *j* to visit:

$$p_{ij}^{k} = \begin{cases} q_0 + (1-q_0)\dfrac{\tau_{ij}\cdot\eta_{ij}^{\beta}}{\sum\limits_{l\in S_k}\tau_{il}\cdot\eta_{il}^{\beta}} & \textit{if } j = \arg\max_{l\in S_k} \tau_{il}\eta_{il}^{\beta}, \\ (1-q_0)\dfrac{\tau_{ij}\cdot\eta_{ij}^{\beta}}{\sum\limits_{l\in S_k}\tau_{il}\cdot\eta_{il}^{\beta}} & \textit{otherwise,} \end{cases} \tag{9}$$

where q_0 is a random constant in [0, 1].

Attempts have been made to improve the performance of these ACO algorithms since their introduction. The accumulated experience ant colony (AEAC) introduces a weighting factor based on the results of the current and/or previous iterations into the element selection for the next round [14]. The ACO with multiple ant clans (ACOMAC) allows different groups of ant to exchange information at some stages in pheromone update, incorporating other search strategies during execution [16]. The cunning ant system (cAS) allows an ant to construct a solution according to both the pheromone density and part of a solution from a previous iteration so as to reduce premature stagnation and thus improve the performance of AS algorithms [18]. In minimum spanning tree ant colony optimization (MST-ACO), a dual nearest insertion procedure is adopted to initialize the pheromone; the low bound is computed by 1-minimum spanning tree; the Lin-Kerninghan local search is also utilized for selecting the next city to visit [21]. In an improved MMAS (iMMAS) algorithm [22], the pheromone concentration of a tour is reinitialized using a modified τ_{mim}-τ_{max} scheme to speed up the searching process of MMAS.

The performance of ACO can be much improved by using a pheromone diffusion model (PDACO) to facilitate the intercommunication among nearby ants [15]. The pheromone diffusion model in PDACO was replaced by an ant-constant pheromone diffusion model for ACO (ACPDMACO) that takes the energy consumption of an ant during the tour into consideration of pheromone update in each iteration [20], this involving a complex mathematical model. A multi-direction search rule was introduced into ACO (MSACO) for conducting local search during iteration, and satisfactory outcomes were reported [19]. Pilat and White [24] investigated the performance of embedding GA into ACO (ACSGA and MetaACS) for solving TSPs but the trail did not yield improved results.

3 Cooperative ACS-GA (CoACSGA) for TSPs

Unlike those hybrid intelligent ACO algorithms proposed previously, in which GA, EA, or PSO played only a partial role in fine tuning the parameters for accelerating ACO convergence, we here propose a new algorithm that is designed to execute both

ACS and GA concurrently and cooperatively. This algorithm not only maximizes the advantages offered by both ACS and GA independently, but also makes ACS converge to the global optima faster driven by quick convergence and diversity control provided by GA. A generic expression of CoACSGA is given in Algorithm 1.

Although this generic algorithm looks simple, it requires a good understanding on some key concepts adopted in this algorithm which differ it from others. Firstly, the strategy of selecting the next city for an ant to visit is based on natural ordering and selection. At each city, we create a sorted list of a certain number of cities that are the closest to it. In a natural selection process, the closest cities have a higher probability to be chosen for the next move. To make the use of this list more efficient, this sorted list should only contain some closest cities. This constant (C_0) depends on the city number n of a particular TSP, and a rough guide would be $C_0 = (5\% - 15\%) \times n$.

```
Set parameters; initialize pheromone trails
Construct FirstAntSolutions (FAS) by ACS
Initialize GA using FAS
while termination condition not met do
  Construct NewAntSolutions (NAS) by ACS
  Get NewAntGeneration (NAG) by GA
  Select NewBestSolution (NBS) from NAS and NAG
  Update Pheromones
endwhile
Terminate with NBS as the solution
```

Algorithm 1. The cooperative ACS-GA (CoACSGA) algorithm for TSPs

For ant k at city i to select the next city j to visit, ant k will first consult the sorted list c(i) of city i to select the closest city from it. If this city is in set S_k, it will be the next city to visit. If $c(i)$ has no intersection with S_k, then the traditional ACS mechanism is used to select a city outside the list. This selection process is summarized as:

$$j = \begin{cases} \min c(i) & \text{if } j \in S_k, \\ \arg\ \max \tau_{ij}\eta_{ij}^{\beta} & \text{otherwise.} \end{cases} \tag{10}$$

Since this natural ordering and selection strategy is adopted for choosing the next city to visit after each iteration, local pheromone update is no longer required.

The solution (sequence of cities visited) of each ant's first ACS iteration constitutes a component of the initial population for GA. Afterwards ACS and GA are executed concurrently in the *while* loop. At the end of each iteration, the two best solutions with the shortest length of tour from both ACS (A_{best}) and GA (G_{best}) are compared to determine the shortest path for the round. Whichever is shorter, it is used to replace the other. This keeps the best solution of each round evolving for fast convergence.

4 Simulation Results and Discussion

The proposed CoACSGA was used to simulate the following four well-known TSPs: Berlin52, Eil51, Eil76, and KroA100. The parameters used for individual problems are given in Table 1, and the maximum iteration number was set to 500 for all four cases. The simulation results of running each case 20 times are listed in Table 2.

Our simulation shows that the optimal solution for each case was achieved at least twice in every 20 runs of CoACSGA and each best solution was reached within 120 iterations at the worst.

The performance of an algorithm can be evaluated from many aspects associated with its design, implementation, parameter setting, running environment, and so forth. We here choose the accuracy of the best solutions, computing cost and optimal convergence, and consistency as the major factor to evaluate the performance of CoACSGA, compared with the published results of some other algorithms.

Table 1. Parameters used in CoACSGA for simulating TSPs

TSP	n	m	α	β	q_0	τ_0	ρ	C_0	Q	p_c	p_m
Berlin52	52	50	0.1	2	0.5	20	0.1	6	10000	0.4	0.6
Eil51	51	50	0.1	6	0.5	20	0.1	8	1000	0.4	0.6
Eil76	76	60	0.1	10	0.5	20	0.1	8	1000	0.4	0.6
KroA100	100	50	0.1	5	0.5	20	0.1	8	100000	0.4	0.6

Note: p_c is the probability for crossover and p_m is the probability for mutation in GA.

Table 2. Simulation results for TSPs using CoACSGA (20 runs for each case)

TSP	Solution			Times the best reached	Iterations to reach the best		Optimal solution [3]
	Best	Worst	Mean		Best	Worst	
Berlin52	7542	7869	7632	8	7	116	7542
Eil51	426	437	429	4	31	103	426
Eil76	538	555	544	2	38	64	538
KroA100	21282	21799	21493	2	2	111	21282

The best solutions for Eil51, Eil76, and KroA100 of eight GA/ACO related algorithms are listed in Table 3. Since the results for Berlin52 have been published by different authors, these best solutions are tabulated in Table 4 separately. It should be noted that we exclude a few conference articles reporting some algorithms that achieved either the optimal solutions to these selected TSPs even better than the known benchmarking optimal solutions, or the known optimal solutions but without providing necessary information on how these were reached.

For Eil51, Eil76, and KroA100, our algorithm is the only one that achieved the optimal solutions to all cases among the eight GA/ACO algorithms. The noticeable point for these cases is that the algorithms (ACSGA and MetaACS) proposed by Pilat and White [24] of combining GA and ACS failed to reach the optimal solutions for Eil51 and KroA100. This could be attributed to the sequential design between GA and ACS, which did not encourage either approach to maximize its advantages independently during iteration. Efforts of both GA and ACS might be cancelled out at some stage during the process. This potential conflict is avoided in CoACSGA by adopting a concurrent model during each iteration and hence the optimal solutions are achieved quickly.

Table 3. Best solutions resulted from different GA/ACO algorithms for the selected TSPs

TSP	KOS [3]	CoACSGA	GA [9]	GCGA [9]	MMAS [13]	ACS [16]	ACOMAC [16]	ACSGA [24]	MetaACS [24]
Eil51	**426**	**426**	430	427	427	434	430	432	428
Eil76	**538**	**538**	552	550	N/A	559	555	N/A	542
KroA100	**21282**	**21282**	21554	21292	21320	21684	21457	21544	21513

For Berlin52, in addition to our algorithm, ACPDMACO [20] also achieved the optimal solution to the problem. Among other ACO algorithms, ACS, MSACO, and iMMAS are very close to reach the optimal solution (Table 4).

Table 4. Best solutions resulted from different ACO algorithms for Berlin52

TSP	KOS [3]	CoACSGA	AS [22]	MMAS [22]	iMMAS [22]	ACS [19]	MSACO [19]	PDACO [20]	ACPDMACO [20]
Berlin52	**7542**	**7542**	7683	7665	7545	7544	7544	7664	**7542**

In our experiments, the optimal solutions to all four cases have been found at least twice in 20 runs (10%), particularly 8 times (40%) for Berlin52 and 4 times (20%) for Eil51. Such high repetition rate has not been reported in all the referred publications. This indicates that CoACSGA has a high level of consistency in successful searching optimal solutions for TSPs, particularly for small-size TSPs, such as Eil51 and Berlin52.

5 Conclusion

The proposed new hybrid algorithm, CoACSGA, combines both GA and ACO together in a cooperative and concurrent fashion to improve the performance of ACO for solving TSPs. This is different from the previous studies that regarded GA as a sequential part of the entire searching process and only used the selected result from GA as the input to the subsequent ACO iteration. The mutual information exchange between ACS and GA at the end of each iteration ensures the selection of the best solution for the next round, which accelerates the convergence. The cooperative approach also creates a better chance for reaching the global optimal solution because the independent running of GA will maintain a high level of diversity in producing next generation of solutions. Our simulation not only proves these expectations, but also indicates that this approach has a high level of consistency in successful searching optimal solutions for TSPs, particularly for small-size TSPs.

References

1. Flood, M.M.: The traveling salesman problem. Operation Research 4, 61–78 (1955)
2. Lawer, E.L., Lenstra, J.K., Kan, A.H.R., Shmoys, D.B.: The traveling salesman problem. Wiley, New York (1985)
3. TSPLIB, `http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/tsp/`
4. Reinelt, G.: The traveling salesman: computational solutions for TSP applications. Springer, Heidelberg (1994)

5. Leung, K.S., Jin, H.D., Xu, Z.B.: An expanding self-organizing neural network for the traveling salesman problem. Neurocomputing 6, 267–292 (2004)
6. Masutti, T.A.S., de Castro, L.N.: A self-organizing neural network using ideas from the immune system to solve the traveling salesman problem. Information Sciences 179, 1454–1468 (2009)
7. Lo, C.C., Hus, C.C.: Annealing framework with learning memory. IEEE Transactions on System, Man, Cybernetics - Part A 28, 1–13 (1998)
8. Jayalakshmi, G.A., Sathiamoorthy, S.: A hybrid genetic algorithm – a new approach to solve traveling salesman problem. International Journal of Computational Engineering Science 2, 339–355 (2001)
9. Yang, J.H., Wu, C.G., Lee, H.P., Liang, Y.C.: Solving traveling salesman problems using generalized chromosome genetic algorithm. Progress in Natural Science 18, 887–892 (2008)
10. Dorigo, M., Maniezzo, V., Colorni, A.: The ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics – Part B 26, 29–42 (1996)
11. Dorigo, M., Gambardella, L.M.: Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation 1, 53–66 (1997)
12. Stutzle, T., Hoos, H.: The MAX-MIN ant system and local search for the traveling salesman problem. In: Proceedings of the IEEE International Conference on Evolutionary Computation, Piscataway, USA, pp. 309–314 (1997)
13. Stutzle, T., Hoos, H.: MAX-MIN Ant System. Future Generation Computer Systems 16, 889–914 (2000)
14. Montgomery, J., Randall, M.: The accumulated experience ant colony for the traveling salesman problem. International Journal of Computational Intelligence and Applications 3, 189–198 (2003)
15. Huang, G.R., Cao, X.B., Wang, X.F.: An ant colony optimization algorithm based on pheromone diffusion. Acta Electronica Sinica 32, 865–868 (2004)
16. Tsai, C.F., Tsai, C.W., Tseng, C.C.: A new hybrid heuristic approach for solving large traveling salesman problem. Information Sciences 166, 67–81 (2004)
17. Birattari, M., Pellegrini, P., Dorigo, M.: On the invariance of ant colony optimization. IEEE Transactions on Evolutionary Computation 11, 732–742 (2007)
18. Tsutsui, S.: Ant colony optimization with cunning ants. Transactions of Japanese Society for Artificial Intelligence 22, 9–36 (2007)
19. Cai, Z.Q.: Multi-direction searching ant colony optimization for travelling salesman problem. In: Proceedings of 2008 International Conference on Computational Intelligence and Security, pp. 220–223 (2008)
20. Ji, J.Z., Huang, Z., Wang, Y.M., Liu, C.N.: A new mechanism of pheromone increment and diffusion for solving travelling salesman problems with ant colony algorithm. In: Proceedings of the Fourth International Conference on Natural Computation, pp. 558–563 (2008)
21. Zhang, Y., Li, L.J.: MST ant colony optimization with Lin-Kerninghan local search for the traveling salesman problem. In: proceedings of the 2008 International Symposium on Computational Intelligence and Design, pp. 344–347 (2008)
22. Li, T.K., Chen, W.Z., Zheng, X., Zhang, Z.: An improvement of the ant colony optimization algorithm for solving travelling salesman problem (TSP). In: Proceedings of the 5th International Conference on Wireless Communications, Networking and Mobile Computing, pp. 1–3 (2009)
23. Dorigo, M., Birattari, M., Stutzle, T.: Ant colony optimization: artificial ants as a computational intelligence technique. IEEE Computational Intelligence Magazine 1, 28–39 (2006)
24. Pilat, M.L., White, T.: Using genetic algorithms to optimize ACS-TSP. In: Proceedings of the 3rd International Workshop on ants, Brussels, pp. 282–287 (2002)

Tracking Control of Uncertain DC Server Motors Using Genetic Fuzzy System

Wei-Min Hsieh[1], Yih-Guang Leu[2], Hao-Cheng Yang[2], and Jian-You Lin[2]

[1] Department of Electronic Engineering, Hwa Hsia Institute of Technology
[2] Department of Applied Electronics Technology, National Taiwan Normal University
leuyk@ntnu.edu.tw

Abstract. A controller of uncertain DC server motor is presented by using the fuzzy system with a real-time genetic algorithm. The parameters of the fuzzy system are online adjusted by the real-time genetic algorithm in order to generate appropriate control input. For the purpose of on-line evaluating the stability of the closed-loop system, an energy fitness function derived from backstepping technique is involved in the genetic algorithm. According to the experimental results, the genetic fuzzy control scheme performs on-line tracking successfully.

Keywords: genetic algorithm, DC server motor, control systems.

1 Introduction

Genetic algorithms (GAs) utilize the concept of "survival of the fittest". Based on the procedure of the genetic operations, GAs can find an optimal solution for engineering problems, such as optimization problems and machine learning. Therefore, GAs, which possess the simple implement ability and the capability of escaping from local optimum, are often incorporated into the design of the fuzzy systems [1-16] in order to search the optimal parameters of the fuzzy systems. In [17], to guarantee the system stability for the nonlinear systems, the solution of the Lyapunov condition is solved by the genetic algorithm. An optimal genetic fuzzy controller design with index function obtained by genetic algorithms has been proposed in [18]. Vehicle system control involved in GAs has been proposed in [19-20]. Motor control via GAs has been proposed in [21-24].

Traditionally, GAs require the procedure of off-line learning before they on-line control a plant. For the purposes of avoiding the procedure of off-line learning and evaluating on-line the stability of the closed-loop systems, an energy fitness function derived from backstepping technique is involved in the genetic algorithm. In this paper, a simple genetic algorithm with the energy fitness function is used to tune the parameters of fuzzy system for the control of uncertain DC server motors. The parameters of the fuzzy system are online adjusted by the simple genetic algorithm in order to generate appropriate control input for uncertain DC server motors.

2 Design of Backstepping Controller for DC Server Motors

The equivalent model of DC server motors is shown in Fig. 1:

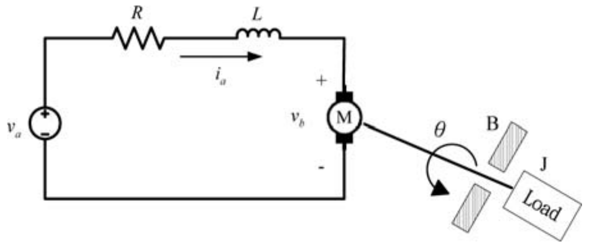


Fig. 1. The equivalent model of DC server motors

where R is the armature resistance, L is the armature inductance, θ is the angular displacement of the motor shaft, B is the friction constant, and J is the armature moment of inertia of the motor.

Let $\mathbf{x}(t)=\begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}=\begin{bmatrix} \theta(t) \\ \dot{\theta}(t) \end{bmatrix}$. Suppose that $L=0$. Then, the representation of the state space for DC server motors is given as

$$\dot{\mathbf{x}}(t)=\begin{bmatrix} 0 & 1 \\ 0 & -\left(K_t K_b + RB\right)/JR \end{bmatrix}\mathbf{x(t)}+\begin{bmatrix} 0 \\ K_t/JR \end{bmatrix} v_a \tag{1}$$

Our control objective is to develop the genetic fuzzy controller so that the uncertain DC server motor system output can asymptotically track a bounded command x_r.

Now, the detail design procedure of the backstepping controller for motor systems (1) is described as follows.

Let the tracking error be $z_1 = x_1 - x_r$. Then, the derivative of z_1 can be expressed as

$$\dot{z}_1 = \dot{x}_1 - \dot{x}_r \tag{2}$$

Define a virtual control as

$$\alpha_1 = \dot{x}_r - c_1 z_1 \tag{3}$$

where $c_1 > 0$ is a design parameter. By using $z_2 = \dot{x}_1 - \alpha_1$, equation (2) can be rewritten as

$$\dot{z}_1 = z_2 - c_1 z_1 \tag{4}$$

The derivative of z_2 can be expressed as

$$\dot{z}_2 = \dot{x}_2 - \dot{\alpha}_1 = \dot{x}_2 - (-c_1 \dot{z}_1 + \ddot{x}_r) \tag{5}$$

Define the control law as

$$v_a = \frac{JR}{K_t}[\ddot{x}_r - c_1\dot{z}_1 - c_2 z_2 - z_1 + f] \tag{6}$$

where $c_2 > 0$ is a design parameter, and $f = (K_t K_b + RB)/JR$. Then, from (6), equation (5) can be rewritten as

$$\dot{z}_2 = -c_2 z_2 - z_1 \tag{7}$$

Consider the Lyapunov function as $V = \frac{1}{2}\sum_{i=1}^{2} z_i^2$. By differentiating Lyapunov function V, and using (4) and (7), we have

$$\dot{V} = \sum_{i=1}^{2} z_i\,\dot{z}_i \le -c_1 z_1^2 \tag{8}$$

Based on the above discussion, the state trajectory x_1 can asymptotically track the bounded reference x_r. Since f is uncertain, the optimal control law (6) cannot be obtained. To solve this problem, the fuzzy system is used to approximate the uncertain continuous function f. First, the uncertain continuous function f in (6) is replaced by fuzzy system, i.e., $\hat{f}$. The resulting control law is

$$v_a = \frac{JR}{K_t}\left[\ddot{x}_r - c_1\dot{z}_1 - c_2 z_2 - z_1 + \hat{f}\right] \tag{9}$$

Then, substituting (9) in (1), we obtain the error equation

$$\dot{z}_2 = f - \hat{f} - c_2 z_2 - z_1 \tag{10}$$

Using (10), we have

$$\begin{aligned}\dot{V} &= \sum_{i=1}^{2} z_i\,\dot{z}_i \\ &= z_1(z_2 - c_1 z_1) + z_2(f - \hat{f} - c_2 z_2 - z_1) \\ &= -\sum_{i=1}^{2} c_i z_i^2 + z_2[f - \hat{f}] \\ &\le -\sum_{i=1}^{2} c_i z_i^2 + |z_2| f^U - z_2\hat{f}\end{aligned} \tag{11}$$

where $|f| \le f^U < \infty$. In order to instantaneously evaluate the stability of the closed-loop system, we define an energy fitness function as

$$F = z_2\hat{f} \tag{12}$$

Note that a chromosome with the largest energy fitness function denotes the optimal solution.

3 Genetic Fuzzy Systems

Generally, the dynamic of DC server motors is uncertain because they suffer from structured and unstructured uncertainties such as load variation, friction, external disturbances, and so forth. Therefore, in order to acquire favorable performance for motor control, it is necessary that a control algorithm should possess robustness to uncertainties. In this paper, fuzzy system is used to approximate the uncertainties. A simplified genetic algorithm is used to adjust the parameters of the fuzzy system in order to instantaneously generate the appropriate control strategy. The simplified genetic algorithm has an energy fitness function which is used to evaluate the real-time stability of the closed-loop systems.

First, we construct the fuzzy inference engine, and use fuzzy IF-THEN rules to perform a mapping from training input data x_q , $q = 1, 2, \cdots, n$, and the output data y . By using product inference, center-averaging, and singleton fuzzification, the output of the fuzzy system y can be expressed as:

$$y = \frac{\sum_{i=1}^{h} w^i \left(\prod_{q=1}^{n} \mu_{A_q^i}(x_q) \right)}{\sum_{i=1}^{h} \left(\prod_{q=1}^{n} \mu_{A_q^i}(x_q) \right)} \tag{13}$$

where $\mu_{A_q^i}(x_q)$ is the membership function of A_q^i , h is the total number of IF-THEN rules, and w^i is the point at which $\mu_{B^i}(w_p^i) = 1$. To evolutionarily obtain the adjustable parameters w^i of the fuzzy system, we define the chromosomes as

$$\Xi = \begin{bmatrix} {}^1w \\ {}^2w \\ \vdots \\ {}^kw \end{bmatrix} = \begin{bmatrix} {}^1w_1 & {}^1w_2 & \cdots & {}^1w_h \\ {}^2w_1 & {}^2w_2 & \cdots & {}^2w_h \\ \vdots & \vdots & \ddots & \vdots \\ {}^kw_1 & {}^kw_2 & \cdots & {}^kw_h \end{bmatrix} \tag{14}$$

where ${}^\ell w = [{}^\ell w_1 \; {}^\ell w_2 \cdots {}^\ell w_h]$ is the ℓth chromosome for $\ell = 1, 2, \ldots, k$. The performance of each chromosome is evaluated according to the defined energy fitness function (12). After generations of evolution, it is expected that the genetic algorithm converges, and a best chromosome with largest fitness representing the optimal solution to the problem is obtained. The crossover [16] uses the single gene crossover operator, where the single gene crossover operator generates new genes, ${}^i\hat{w}_j$, only at the position j-th for all chromosomes. After sorting, the first chromosome is the best one in the population in terms of fitness. As to the operation procedure of the mutation operator, the *(k/2+1)*-th chromosome is replaced by the sum of the first chromosome and a random perturbation according to the mutation rate p_m .

4 Experimental Results

In this section, using practical circuits and a microcontroller to control the DC server motor system verifies the performance of the proposed method. The block diagram of hardware implementation is shown in Fig. 2. The hardware structure is composed of a personal computer, a microcontroller (IC 82G516), a switching DC-DC converter, and SEM MT22R2-24 DC server motor system. The proposed method is implemented by using the personal computer with 5-ms sampling interval, and the DC server motor system is controlled by Pulse-Width Modulation (PWM) method, where switch-duty ratio $D \in [-1,1]$ is varied to adjust the output of the Buck DC-DC converter.

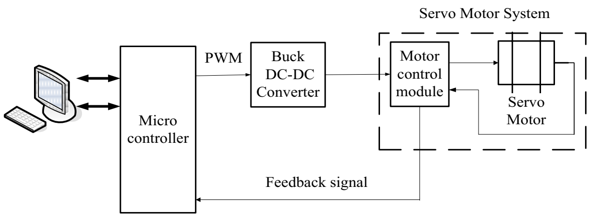
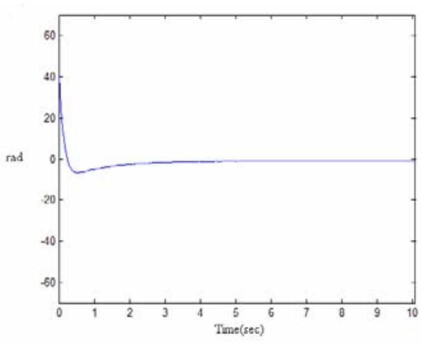


Fig. 2. The block diagram of hardware implementation

It is assumed that the design parameters are set as $c_1 = 10$, $c_2 = 15$, $k = 4$, and $h = 49$. The desired reference signal is set as $x_r = 60 - 100\exp(-t)$. The experimental results are shown in Figs. 3-5. The tracking error response is shown in Fig. 3. The associated switch duty ratio D and the associated control input voltage v_a are shown in Figs. 4 and 5, respectively. From the experimental results, it is shown that the state x_1 of the server motor can track the reference signal x_r well.



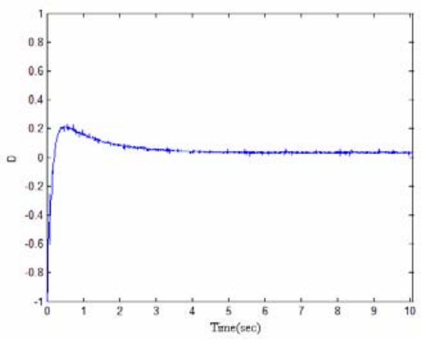


Fig. 3. The tracking error $z_1 = x_1 - x_r$

Fig. 4. The switch duty ratio D

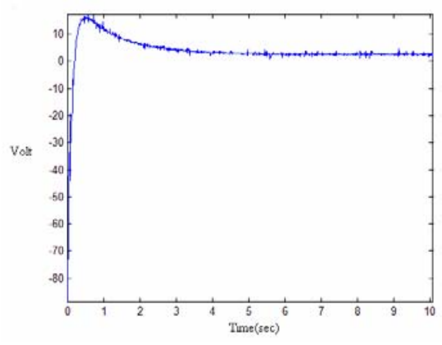


Fig. 5. The control input voltage v_a

5 Conclusions

The genetic fuzzy control scheme has been proposed for uncertain DC server motor. The parameters of the fuzzy controller can be tuned instantaneously via the genetic algorithm without the procedure of off-line learning. The experimental results have shown that the genetic fuzzy control scheme performs on-line tracking successfully.

Acknowledgment

This work was supported by the National Science Council of Taiwan, R.O.C., under Grants NSC 96-2221-E-003-012-MY3.

References

1. Lam, H.K., Leung, F.H., Tam, P.K.S.: Design and stability analysis of fuzzy model-based nonlinear controller for nonlinear systems using genetic algorithm. IEEE Transactions on Systems, Man and Cybernetics, Part B 33(2), 250–257 (2003)
2. Chou, C.H.: Genetic algorithm-based optimal fuzzy controller design in the linguistic space. IEEE Transactions on Fuzzy Systems 14(3), 372–385 (2006)
3. Cai, L., Rad, A.B., Chan, W.L.: A Genetic Fuzzy Controller for Vehicle Automatic Steering Control. IEEE Transactions on Vehicular Technology 56(2), 529–543 (2007)
4. Mirzaei, A., Moallem, M., Dehkordi, B.M., Fahimi, B.: Design of an Optimal Fuzzy Controller for Antilock Braking Systems. IEEE Transactions on Vehicular Technology 55(6), 1725–1730 (2007)
5. Yang, Z., Hachino, T., Tsuji, T.: Model reduction with time delay combining the least-squares method with the genetic algorithm. IEE Proc. Control Theory Appl. 143(3) (1996)
6. Michalewicz, Z.: Genetic Algorithms + Data Structure = Evolution Program. Springer, Heidelberg (1996)
7. Leu, Y.G., Wang, W.Y., Li, Y.H.: RGA-based On-Line Tuning of BMF Fuzzy-Neural Networks for Adaptive Control of Uncertain Nonlinear Systems. Neurocomputing 72, 2636–2642 (2009)

8. Lawrence, D.: Handbook of Genetic Algorithms. Van Nostrand, Reinhold (1991)
9. Iwasaki, M., Miwa, M., Matsui, N.: GA-based evolutionary identification algorithm for unknown structured mechatronic systems. IEEE Transaction on Industrial Electronics 52, 300–305 (2005)
10. Gonzalez, A., Perez, R.: Selection of Relevant Features in a Fuzzy Genetic Learning Algorithm. IEEE Transactions on Systems, Man and Cybernetics, Part B 31(3) (June 2001)
11. Caponetto, R., Fortuna, L., Graziani, S., Xibilia, M.: Genetic algorithms and applications in system engineering: a survey. Trans. Inst. Meas. Control 15, 143–156 (1993)
12. Game, G.W., James, C.D.: The Application of Genetic Algorithms to the Optimal Selection of Parameter Values in Neural networks for Attitude Control System. IEEE Colloquium on High Accuracy Platform Control in Space 3/1–3/3 (1993)
13. Lin, C.T., Jou, C.P.: GA-based fuzzy reinforcement learning for control of a magnetic bearing system. IEEE Transactions on Systems, Man and Cybernetics, Part B 30(2), 276–289 (2000)
14. Wang, C.H., Liu, H.L., Lin, C.T.: Dynamic optimal learning rates of a certain class of fuzzy neural networks and its applications with genetic algorithm. IEEE Transactions on Systems, Man and Cybernetics, Part B 31(3), 467–475 (2001)
15. Brown, M., Harris, C.J.: Neurofuzzy adaptive modelling and control. Prentice-Hall, Englewood Cliffs (1994)
16. Wang, W.Y., Li, Y.H.: Evolutionary learning of BMF fuzzy-neural networks using a reduced-form genetic algorithm. IEEE Transactions on Systems, Man and Cybernetics, Part B 33, 966–976 (2003)
17. Lam, H.K., Leung, F.H., Tam, P.K.S.: Design and stability analysis of fuzzy model-based nonlinear controller for nonlinear systems using genetic algorithm. IEEE Trans. Syst., Man Cybern. Part B 33(2), 250–257 (2003)
18. Chou, C.H.: Genetic algorithm-based optimal fuzzy controller design in the linguistic space. IEEE Transactions on Fuzzy Systems 14(3), 372–385 (2006)
19. Cai, L., Rad, A.B., Chan, W.L.: A Genetic Fuzzy Controller for Vehicle Automatic Steering Control. IEEE Transactions on Vehicular Technology 56(2), 529–543 (2007)
20. Mirzaei, A., Moallem, M., Dehkordi, B.M., Fahimi, B.: Design of an Optimal Fuzzy Controller for Antilock Braking Systems. IEEE Transactions on Vehicular Technology 55(6), 1725–1730 (2007)
21. Cheng, M., Sun, Q., Zhou, E.: New Self-Tuning Fuzzy PI Control of a Novel Doubly Salient Permanent-Magnet Motor Drive. IEEE Transactions on Industrial Electronics 53(3), 814–821 (2006)
22. Lin, F.J., Huang, P.K., Chou, W.D.: Recurrent-Fuzzy-Neural- Network -Controlled Linear Induction Motor Servo Drive Using Genetic Algorithms. IEEE Transactions on Industrial Electronics 54(3), 1449–1461 (2007)
23. Wai, R.J., Tu, C.H.: Lyapunov-based Genetic Algorithm Controlled Linear Piezoelectric Ceramic Motor Drive. In: Wai, R.J., Tu, C.H. (eds.) IEEE Conference on Cybernetics and Intelligent Systems, 2006, June 7-9, pp. 1–6 (2006)
24. Wai, R.J., Tu, C.H.: Design of Total Sliding-Mode-Based Genetic Algorithm Control for Hybrid Resonant-Driven Linear Piezoelectric Ceramic Motor, Power Electronics. IEEE Transactions on Power Electronics 22(2), 563–575 (2007)

Novel Multi-Objective Genetic Algorithm Based on Static Bayesian Game Strategy

Zhiyong Li, Dong Chen, Ahmed Sallam, and Li Zhao

School of Computer and Communication, Hunan University,
Changsha, 10082, P.R. China
jt_lizhiyong@hnu.cn
Chendong0101@qq.com

Abstract. Multi-objective evolutionary algorithms (MOEAs) have been the mainstream to solve multi-objectives optimization problems. In this paper we add the static Bayesian game strategy into MOGA and propose a novel multi-objective genetic algorithm(SBG-MOGA). Conventional MOGAs use non-dominated sorting methods to push the population to move toward the real Pareto front. This approach has a good performance at earlier stages of the evolution, however it becomes hypodynamic at the later stages. In SBG-MOGA the objectives to be optimized are similar to players in a static Bayesian game. A player is a rational person who has his own strategy space. A player selects a strategy and takes an action to realize his strategy in order to achieve the maximal income for the objective he works on. The game strategy will generate a tensile force over the population and this will obtain a better multi-objective optimization performance. Moreover, the algorithm is verified by several simulation experiments and its performance is tested by different benchmark functions.

Keyword: Multi-Objective Optimization, Genetic Algorithm, Static Bayesian Game.

1 Introduction

Many real world problems are multi-objective optimization problems (MOP). A MOP is a problem which has two or more objectives that need to be optimized simultaneously, and there might be constraints imposed on these objectives, even that the objectives of the MOP are in conflict with each other. Generally, the MOPs could be expressed as follows,

$$min\{z_1 = f_1(x), z_2 = f_2(x), \ldots, z_q = f_q(x)\} \tag{1}$$

$$\text{s.t.} g_i(x) \leq 0, i = 1, 2, \ldots, m \tag{2}$$

Where $x \in R^n$is the vector of the decision variables, $f_i(x)$ is one of the q objective functions, and $g_i(x)$ is one of the m inequality constrain functions which constitute the feasible solutions space. The Multi-objective optimization (MO) aims to optimize the vector-valued objective function. Unlike signal objective optimization, the solution of a MOP is not a signal point, but a set of points that represent the best trade-offs between the objective functions. At the time Mathematical programming techniques manifest their limitation to solve Multi-objective optimization problems (MOP), Evolutionary algorithms (EA) such as genetic algorithm have more advantages to solve MOPs. For example EAs deal simultaneously with a set of possible solutions in a single run, and are less susceptible to the shape or continuity of the Pareto front.

During the past two decades, a large number of MOEAs have been suggested, among which there are many excellent and representative algorithms, such as NSGA[4], NPGA[5], MOGA[3], SPEA2[6], PAES[8], NSGA-II[7]. At the beginning, NSGA[4] and NPGA[5] introduced non-dominated sorting method with MOEAs, which demonstrated the capability of the MOEAs in solving MOP. However these approaches did not include elitism. After the present of SPEA[6], elitism mechanism had been an indispensable composition of MOEAs, and in [1] it has been seen as a landmark of the field by the authors. SPEA2[6] proposed by Eckart Zitzler et al, has three main differences with SPEA: 1)An improved fitness assignment scheme was used; 2) A nearest neighbor density estimation technique was incorporated; 3) A new archive truncation methods guarantee the preservation of boundary solutions was created. These improvements give SPEA2 a better performance. NSGA-II[7] usually considered as a completely different algorithm with NSGA, this algorithm has proposed a fast non-dominated sorting approach with O(MN^2) computational complexity.

All these mentioned MOEAs are non-dominated sorting based, which generate force to push the population to move toward Pareto front In this paper we introduce SBG-MOGA which adds static Bayesian game strategy into MOGA. So in SBG-MOGA, besides the force generated by the non-dominated sorting, there will be a tensile force generated by the running game between objectives. These two kinds of forces together act on the population, which will obtain a better performance in terms of finding a diverse set of solutions and drawing near the true Pareto-optimal set.

2 Related Work

Because of the properties of genetic algorithm, it's very fit to solve the multi-objective optimization problems (MOP). The main steps of the signal objective optimization genetic algorithm (GA) are: 1) initialize the original population; 2) compute the fitness of the individuals in the population; 3) put the crossover and mutation operators on the parents population and give birth to a new offspring population; 4) if the algorithm meets the end conditions then stop running, or else jump to step 2). Unlike the signal objective optimization GA, the MOGA have many new properties, such as Pareto Optimal Set, non-dominated sorting, elitism mechanism.

2.1 Non-dominated Sorting

The solutions of MOPs are a set of points known as Pareto Optimal Set that cannot be improved. In order to push the population to move toward Pareto Optimal Set, NSGA[4], NPGA[5] and MOGA[2] introduce the non-dominated sorting into MOEA. First, the definitions of dominate and non-dominate are as follows (without loss the generality, and assuming the minimization problem.): Vector u=(u_1,u_2,...,u_n); vector v=(v_1,v_2,...,v_n)

Definition 1. (dominate)
u is dominate v if

$$u_i \leq v_i, \forall i = 1, \ldots, n \tag{3}$$

$$and \quad u_i \prec v_i, \exists i = 1, \ldots, n \tag{4}$$

Definition 2. (non-dominate)
u and v are said to be non-dominate to one another if u is not dominate to v and v is not dominate to u.

MOGA changes the step 2) of GA as follows: 1) the population is ranked on the basis of non-domination; 2) The fitness assigned to the individuals from the rank 1 to rank n decrease gradually. In many algorithms, nearest neighbor density estimation technique is used with MOGA to allow a more precise guidance of the search process [6]. Later NSGA-II[7] has proposed a fast non-dominated sorting approach, which improves the computational complexity of non-dominated sorting from O(MN^3) to O(MN^2). Firstly, for each individual for example p_i the approach calculates two entities: 1) the domination count n_i, which is the number of solutions that dominates the solution p_i, and 2) S_i, which is a set of solutions that the solution p_i dominates. Secondly, the approach performs the following steps: 1) for each individual p_i(i=1,,n), if n_i=0, put it in a list Q_j (j initialized with 0, then increased by one) and give them fitness; 2) for each individual p_i in Q_j, and for each individual p_t in S_i, decrease n_t by one; 3) if population is not empty, jump to step 1, else stop.

2.2 Elitism Mechanism

The innovation of elitism mechanism enhanced the convergence properties of a MOEA. SPEA[6] suggested an elitism mechanism with the concept of non-domination. The elitism mechanism maintains an external population at every generation storing all non-dominated solutions discovered so far beginning from the initial population. This external population participates in all genetic operations.

3 Proposed Algorithm

In this section, we introduce a novel MOGA based on static Bayesian game strategy (SBG-MOGA). In SBG-MOGA, each generation of evolution is considered as a game and each objective to be optimized is considered as a player.

A player is clever enough to know how to select an appropriate strategy to get the biggest income in the game. The algorithm contains a sequence of rounds, in each round players play with each other, which will provide the population with tensile force to pull them to move toward the true Pareto front. In addition, elitism mechanism is adopted in our algorithm.

3.1 Game Modeling

The game modeling is adopted to decide how a new population is created. First, without loss of generality, the following definitions and assumptions apply:
Assumption (Objective Functions & Individual Vector)
Assuming that there is a minimization problem and the number of objectives is n, and X is the finite search space where the objective function
$F = \{f_1(x), f_2(x), \ldots, f_n(x), x \in X\}$. Also there is a player for each objective, $A = (a_1, a_2, \ldots, a_n)$;And at generation t in MOGA (t is not the last one) the number of individuals is m, the individual vector $B(t) = (b_1, b_2, \ldots, b_m)$;

Definition 3. (Fitness Matrix)
The fitness vector of an individual for all the objectives: $v^{ft} = (t_1, t_2, \ldots, t_n)$*,*
and the fitness matrix of the population $M^{ft} = \left\{\begin{matrix} v_1^{ft} \\ v_2^{ft} \\ \vdots \\ v_m^{ft} \end{matrix}\right\}$

Definition 4. (Profit Matrix)
After all the players in the game perform their actions, there will be a profit matrix: $M^p = \left\{\begin{matrix} p_{10} & p_{11} & \cdots & p_{1n} \\ p_{20} & p_{21} & \cdots & p_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ p_{n0} & p_{n1} & \cdots & p_{nn} \end{matrix}\right\}$ p_{ij} *indicates that the player i's action results a* p_{ij} *profit to player j. To clarify the divergence in* M^p*, we introduce* $M^{p'}$*,where* $p_{ij}' = p_{ij} - \frac{1}{n}\sum_{t=1}^{n} p_{tj}$

Definition 5. (Strategy Vector)
All players have a Strategy space, defined as strategy vector $v^s = (s_1, s_2, \ldots, s_c)$*;*

Definition 6. (Friendly Degree Matrix)
During the game a player (for example ***a****) perform a strategy to maximize its profit, however this may damage or increase another player's (for example* ***b****) profit. So if* ***b****'s profit increased because of* ***a****'s action,* ***b*** *will increase his friendly degree to* ***a****, also he will have a bigger probability to take a friendly strategy for* ***a****. Thus the friendly degree indicates the extent of reduction or increment of* ***b****'s profit and the probability for* ***b*** *to take a certain strategy (malicious or friendly)*

*to **a**. The friendly degree matrix* $M^{fd} = \begin{Bmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ d_{21} & d_{22} & \cdots & d_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ d_{n1} & d_{n2} & \cdots & d_{nn} \end{Bmatrix}$ *Where* d_{ii} *is the friendly degree player "i" to himself, thus* d_{ii} *is assigned with a max-value (for example 100). On the other hand,* d_{ij} *is assigned with a mid-value at first, and during the algorithm run time it will change based on the* p'_{ji}.

The game modeling can be described as follows: each player takes an action to carry out his best strategy which is selected from v^s to maximize his profit. The selection of the best strategy is affected by friendly degree matrix M^{fd}, and the profit matrix M^p, so in order to select the best strategy players must take all the factors including the effects of other players into account. Then the most important is what the action and the strategy exactly are. According to the definitions and assumptions given above, an action and a strategy vector defined as:

Weight Vector. for player "i", there is a weight vector $v_i^w = (w_{i1}, w_{i2}, \cdots, w_{in})$,which reflects the player's reference on each objective. An individual (for example b_k) has a fitness mapping on player "i", which is calculated by $\sum_{j=0}^{n} w_{ij}t_{kj}$.It's obvious that w_{ii} will be the max-value(100), and w_{ij},$i \neq j$ is assigned according to the strategy that the player adopts.

Strategies. The strategy space in our algorithm defined as $v^s = (S_1 = "weight\ w = 75", S_2 = "weight\ w = 25", S_S = "weight\ w = 100")$; where S_S is the strategy that chosen by the players for themselves. A player also chose a strategy to each other player. In other words all the strategies chosen by the player "i" are $(S_{x1}, S_{x2}, \ldots, S_{xn}), where S_{xi} = S_S$.

Action. After the best strategy has been selected, each player "i" 1) gets the weight vector v_i^w based on the strategy, and then 2) calculates the fitness of the population mapped on player "i", using $v^f = M^{ft} \bullet v_i^{wT}$; 3) selects m/n individuals from the population to build up a subpopulation according to sorted fitness $v^{f'}$, puts the crossover and mutation operators on the subpopulation independently. And then the algorithm groups n subpopulations together to build up a new offspring population, then calculates $M^{p'}$, and updates friendly degree matrix M^{fd}.

Take a simple example: there are 3 players in the game model (use 1, 2, 3 to represent the three players respectively) and the friendly degree matrix M^{fd} is the common knowledge. So in order to maximize his benefit, player "1" must consider the strategies chosen by "2" and "3". The strategy (S_S, S_2, S_2) may be not the final best choice for "1", because that will damage the friendship with the other two players and as a result the other players will consider a bigger probability to choose a strategy that harm 1's benefit. Thus "1" must take a comprehensive consideration of friendly degree matrix M^{fd}, profit matrix M^p and the reflection of other players.

3.2 Description of SBG-MOGA and Convergence Properties

According to the analysis above, we propose the general flow of our algorithm as follows:

Step 1. Initialize population B(0) and M^{fd}, initialize an elitism archive set A(0) to store all non-dominated solutions discovered in the algorithm, t=0;
Step 2. Evaluate the fitness matrix, and put non-dominated solutions into elitism archive, of course the elitism mechanism is open, mechanisms such as crowding distance[7] can be adopted;
Step 3. If terminate condition is satisfied, then stop;
Step 4. Each player publish his strategy, and takes an action as described in section 3.1.
Step 5. Increase t such that t:=t+1, then return to step 2.

The step 4 is the generation from B(t) to B(t+1). Let F be a set of objective space of MOGA, the non-dominated set in F is denoted as M($F, \prec$). Let X be the finite search space; f:$X \to F$ be a mapping from X to F. For some $A \subseteq X$,

$$M_f(A, \prec) = \{a \in A : f(a) \in M(f(a)), \prec\}$$

represents the non-dominated individuals in generation A. So our algorithm can be described as figure 1. In literature [2] the authors proved the sufficient convergence condition of multi objective optimization algorithm, and according to their classification our algorithm belongs to the base algorithm VV.

```
1: Initialize B(0) random;
2: $A(0) = M_f(B(0), \prec)$;
3: t=0;
4: repeat:
5: generate B(t+1) from B(t)
6: $A(t+1) = M_f(B(t+1) \cup A(t), \prec)$
7: t=t+1;
8: until stop condition satisfied
```

4 Experiment Results

To verify our algorithm, six benchmark functions were used to make a comparison with NSGA-II [7]. All experiments have been executed on a PC (Pentium IV-3GHz CPU, 512M DDR, WinXP OS, VC++ 6.0). We follow the NSGA-II's authors' suggestion to set the parameters. Population size and generations are set to 100 and 250 respectively. The probability of crossover is set to 0.9, and probability of the mutation is 1n. Both η_c and η_m are set to 20. Moreover, We adopt two variables *C(A, B)* and "Δ" to compare the algorithms accurately. where *C(A, B)* denote the ratio of solutions in A dominates that in B, and "Δ" reflects the performance of solutions' distributing. A good uniform distribution can make "Δ" value be zero.

All test functions are computed 30 times, and then calculate an average value of the results for each performance parameter. These test functions listed in the following formulae are all minimal value problem chosed from a classical study in MOPs field.

Test case 1

$$f_1(x) = x^2$$
$$f_2(x) = (x-2)^2$$

Where $x \in [-10^3, 10^3]$

Test case 2

$$f_1(x) = \begin{matrix} -x & if & x \leq 1 \\ -2+x & if & 1 < x \leq 3 \\ 4-x & if & 3 < x \leq 4 \\ -4+x & if & x > 4 \end{matrix}$$

Where $x \in [-500, 500]$

Test case 3

$$f_1(x) = (x^2 + y^2)^{\frac{1}{8}}$$
$$f_2(x) = ((x-0.5)^2 + (y-0.5)^2)^{\frac{1}{4}}$$

Where $x, y \in [-100, 100]$

Test case 4

$$f_1(x,y) = x$$
$$f_2(x,y) = (1+10y).[1 - (\tfrac{x}{1+10y})^a$$
$$-\tfrac{x}{1+10y}. \sin(2\pi q x)]$$

Where $x, y \in [0,1], a = 2, q = 4$

Test case 5

$$f_1(x) = x$$
$$f_2(x) = g(x)[1 - \sqrt{x_1/g(x)}]$$
$$g(x) = 1 + 9(\textstyle\sum_{i=2}^{n} x_i)/(n-1)$$

Where $x \in [0,1], n = 30$

Test case 6

$$f_1(x) = x_1$$
$$f_2(x) = g(x)[1 - (x_1/g(x))^2]$$
$$g(x) = 1 + 9(\textstyle\sum_{i=2}^{n} x_i)/(n-1)$$

Where $x \in [0,1], n = 30$

Table 1. Experiment Comparison

Distribution comparison of different algorithms using Δ

Test function	1	2	3	4	5	6
SBG-MOGA	0.416	0.712	0.513	0.594	0.483	0.368
NSGA-II	0.450	0.830	0.621	0.613	0.593	0.593

Convergence comparison of different algorithms using C

Test function	1	2	3	4	5	6
C(S,N)	0	0	0.08	0.04	0.1	0
C(N,S)	0	0.03	0.11	0.17	0	0

The results in table 1 shows that our new algorithm has a better performance over NSGA-II for the parameter "Δ", and that our method is approximate with NSGA-II however our proposal has shown a relative disadvantage in convergence in the obtained non-dominated front.

5 Conclusions

Currently, all MOEAs are based on non-dominated sorting approaches, which push the populations to move toward the true Pareto front. The non-dominated sorting approach has a good performance at earlier stages of the evolution, however it becomes a hypodynamia at later stages. In this article, based on analyzing the static Bayesian game model, we proposed a novel multi-objective optimization genetic algorithm. The experiments are based on six difficult problems, it shows that the proposed SBG-MOGA can achieve a better distribution than NSGA-II, and SBG-MOGA is approximate to NSGA-II in convergence in the obtained non-dominated set.

Acknowledgments. This work is sponsored by the National Basic Research Program of China (973) with grant No.2006CB and Science Technology Plan of ChangSha with grant No. K0901022-11.

References

1. Coello Coello, C.A.: Evolutionary multi-objective optimization: a historical view of the field. Computational Intelligence Magazine 1, 28–36 (2006)
2. Agapie, A., Rudolph, G.: Convergence Properties of Some Multi-Objective Evolutionary Algorithms. In: The 2000 Congress on Evolutionary Computation (CEC 2000). IEEE Press, Piscataway (2000)
3. Carlos, M., Peter, J.: Genetic algorithms for multiobjective Optimization: Formulation, Discussion and Generalization. In: Forrest, S. (ed.) Proceedings of the Fifth International Conference on Genetic Algorithms, pp. 416–423 (1993)
4. Srinivas, N., Deb, K.: Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation 2(3), 221–248 (1994)
5. Horn, J., Nafpliotis, N., et al.: A niched pareto genetic algorithm for multiobjective optimization. In: Proceedings of the First IEEE Conference on Evolutionary Computation, Piscataway, New Jersey. IEEE World Congress on Computational Intelligence, vol. 1, pp. 82–87 (June 1994)
6. Zitzler, E., Laumanns, M., et al.: SPEA2: Improving the strength pareto evolutionary algorithm. In: Giannakoglou, K., et al. (eds.) EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, Athens, Greece, pp. 95–100 (2002)
7. Deb, K., Agrawal, S., et al.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: Deb, K., Rudolph, G., Lutton, E., Merelo, J.J., Schoenauer, M., Schwefel, H.-P., Yao, X. (eds.) PPSN 2000. LNCS, vol. 1917, pp. 849–858. Springer, Heidelberg (2000)
8. Knowles, J., Corne, D.: Approximating the nondominated front using the pareto archived evolution strategy. Evolutionary Computation 8(2), 149–172 (2000)
9. Zhiyong, L., Rudolph, G.: A framework of quantum-inspired multi-objective evolutionary algorithms and its convergence condition. In: Proceedings of the 9th annual conference on Genetic and evolutionary computation, London, England, pp. 908–908 (2007)

A Hybrid Pareto-Based Tabu Search for Multi-objective Flexible Job Shop Scheduling Problem with E/T Penalty

Junqing Li[1], Quanke Pan[1], Shengxian Xie[1], and Jing Liang[2]

[1] School of Computer, Liaocheng University, Liaocheng, 252059
[2] School of Electrical Engineering, Zhengzhou University, Zhengzhou, 450001
lijunqing@lcu.edu.cn

Abstract. In this paper, we propose a Pareto-based tabu search algorithm for multi-objective FJSP with Earliness/Tardiness (E/T) penalty. In the hybrid algorithm, several neighboring structure based approaches were proposed to improve the convergence capability of the algorithm while keep population diversity of the last Pareto archive set. In addition, an external Pareto archive was developed to record the non-dominated solutions found so far. In the hybrid algorithm, dynamic parameters were introduced to adapt to the searching process. Experimental on several well-known benchmark instances show that the proposed algorithm is superior to several existing approaches in both solution quality and convergence ability.

Keywords: Flexible job shop scheduling problem, tabu search, Multi-objective optimization, Pareto archive set, Earliness/Tardiness penalty.

1 Introduction

The flexible job-shop scheduling problem (FJSP), as a branch of the classical job-shop scheduling problem (JSP), has been researched in very recent years. The research on the multi-objective FJSP is much less than the mono-objective FJSP. Kacem et al. (2002a, 2002b) [5,6] developed an effective evolutionary algorithm controlled by an assigned model based on the approach of localization (AL). Xia and Wu (2005) [7] presented a practical hierarchical solution approach by making use of PSO to assign operations on machines and simulated annealing (SA) algorithm to schedule operations on each machine. Zhang et al. (2009) [8] introduced a hybrid algorithm combining PSO algorithm with TS algorithm for solving the multi-objective FJSP problems.

To our best knowledge, the research on the multi-objective FJSP tardiness criterion is much less than the other objectives such as the total completion time, the critical machine workload, and the total workload of all machines. The total tardiness criterion was mainly considered in the single machine problem (Chu [10] and Souissi et al. [11]). Few references studied this criterion in a job-shop configuration. Kanet and Hayya [12] developed a priority rule for solving tardiness criterion in job-shop scheduling problems, while Vinicius and Cintia [13] developed a heuristic approach based on tabu search (TS) for the problem. Zribi [14] is the first researcher to solve the FJSP problems with total tardiness constraint. The minimization of the total tardiness on a single machine is an NP-hard problem (Du et al [15]). The FJSP problem is also an NP-hard problem. As a consequence, the FJSPs with tardiness constraint are also NP-hard.

2 Problem Formulation

In this study, we study the FJSP with Earliness/Tardiness (E/T) penalty. The FJSP with E/T penalty can be formulated as follows.

- Let $J = \{J_i\}_{1 \le i \le n}$, indexed *i*, be a set of *n* jobs to be scheduled. Let n_i be total number of operations of job J_i.
- Let $M = \{M_k\}_{1 \le k \le m}$, indexed *k*, be a set of *m* machines.
- Each job J_i consists of a predetermined sequence of operations. Let $O_{i,j}$ be operation *h* of J_i.
- Each operation $O_{i,j}$ can be processed without interruption on one of a set of machines $M(O_{i,j})$. Let $p_{i,j,k}$ be the processing time of $O_{i,j}$ on machine M_k.

Let r_i and C_i be the release date and the completion date of job J_i. W_k is the workload of machine M_k, which is the total processing time of operations that are operated on machine M_k. Three objectives are used in this study, namely [3]:

1) minimization of maximum completion time (makespan):

$$F_1 = \max\{C_i \mid i = 1,.....,n\} \tag{1}$$

2) minimization critical machine workload:

$$F_2 = \max\{W_k \mid k = 1,.....,m\} \tag{2}$$

Let m_j be the number of operations processed on M_j, k_{ji} is the *i*th job number being processed on M_j, $p(k_{ji})$ be the processing time of the operation k_{ji}, *d* is the due date, then following formulation can be given:

1) the completion time of operation k_{ji} is $\sum_{l=1}^{i} p(k_{j,l})$;

2) the earliness of operation k_{ji} is:

$$T[k_{j,i}] = \max\left\{0, \sum_{l=1}^{i} p(k_{j,l}) - d\right\} \ge 0 \tag{3}$$

3) the earliness of operation k_{ji} is:

$$E[k_{j,i}] = \max\left\{0, d - \sum_{l=1}^{i} p(k_{j,l})\right\} \ge 0 \tag{4}$$

In this study, we consider the E/T constraint as an objective for the problem. Therefore, the E/T penalty in this study can be described as follows:

$$F_3 = \sum_{j=1}^{m} \sum_{i=1}^{m_i} \left\{ \alpha d + \beta E[k_{j,i}] + \gamma T[k_{j,i}] \right\} \tag{5}$$

where, β and γ is the penalty coefficient weight for the earliness and tardiness, respectively. α is the coefficient weight for the due date.

3 Pareto-Based Tabu Search

3.1 The Pareto Archive Set *AS*

To provide a set of solutions with good diversity, a Pareto archive set (*AS*) was introduced in this study, which is used to maintain a limited number of non-dominated solutions found so far. During the optimization process, the archive set is iteratively updated with adding some non-dominated solutions and removing some dominated solutions to get closer to the Pareto-optimal front. Once a new non-dominated solution is found, it will be added to *AS* and any solution which is dominated by the added one will be removed from *AS*. In case *AS* becomes overfull, its member which is in the crowded domain is eliminated to maintain the diversity of the Pareto archive set.

3.2 The Storage Structure of *AS*

To reduce the computational time complexity consumed on the update process of the archive set, the members of the *AS* firstly sequence in an ascending order according to their first objective function value (Pan, 2009) [17].

3.3 Population Division

For the population, we should sequence each solution according to a certain criteria. For multi-objective optimization problems, we can not use one objective function value to determine the solution quality. In this study, a non-dominated sort algorithm (Deb et al., 2002 [16]) was first introduced to the population, which divides the solutions into several levels according to their dominated solutions number. The lower the solution level, the better the solution quality. For the solutions in the same level, the solution with smaller crowding distance is considered better than the one with larger crowding distance value [18].

3.4 The Update Process of *AS*

To identify a new non-dominated solution *x*, a naïve and direct approach is to compare it with each member in *AS*. The computational time complexity of this approach is about $O(wh)$, where h denotes the total number of the member solutions in *AS*, w represents the objective number of the problem.

3.5 The Framework of P-TS

The details steps of the proposed P-TS algorithm are as follows:

Step 1: Initialization phase
Step 2: Evaluate the objective function value of each solution in the population, and then record the best solution as the current solution.
Step 3: If the stopping criterion is satisfied, output the best solution; otherwise, select the best solution as the current solution, then perform steps 4-5.
Step 4: Perturbation in the machine assignment component phase.

Step 4.1 Produce a population of neighboring solutions by applying the function neighboring structure approaches proposed in [18] to the current solution.
Step 4.2 Applying non-dominated sort algorithm to the current neighboring population.
Step 4.3 Update the tabu list by adding the best neighboring solution and removing the oldest solution.
Step 4.4 Update the Pareto archive set by using the speed-up Pareto archive set update function to each solution in the Pareto level 1.

Step 5: Perturbation in the operation scheduling component phase.

Step 5.1 Produce a population of neighboring solutions by applying the critical block neighboring structure [18] to the current solution.
Step 5.2 Applying non-dominated sort algorithm to the current neighboring population.
Step 5.3 Update the tabu list by adding the best neighboring solution and removing the oldest solution.
Step 5.4 Update the Pareto archive set by using the speed-up Pareto archive set update function to each solution in the Pareto level 1.

Step 6: go to step 3.

4 Experiment Results

This section describes the computational experiments to evaluate the performance of the proposed algorithm. The test samples come from Kacem instances set [5]. The current instantiation was implemented in C++ on a Pentium IV 1.8GHz with 512M memory.

4.1 Setting Parameters

Each instance can be characterized by the following parameters: number of jobs (n), number of machines (m), and the number of operations (op_num). Followings are the detail parameters value:

- Population size P_{size}: 1000;
- Maximum number of generations $gen_{\max}$: $n \times m$;

- Maximum number of iteration with no improvement of the best solution during the local search $iter_{\max}$: $op\_num/2$;
- $\beta=1$ and $\gamma=1$

4.2 Results Comparisons

The five test instances come from Kacem [5], which range from 4 jobs × 5 machines to 15 jobs × 10 machines. In the five instances, four ones are total FJSP (T-FJSP) while the instance 8×8 is partial FJSP (P-FJSP). Four present algorithms are used to make comparison with the proposed algorithm, i.e., AL+CGA from Kacem [5, 6], PSO+SA from Xia et al. [7], PSO+TS algorithm from Zhang et al. [8], and XING algorithm from Xing et al. [20]. The due date of are 11 for 4 × 5 instance, 14 for 8 × 8 instance, 11 for 10 × 7 instance, 1 for 10 × 10 instance, and 11 for 15 × 10 instance.

Table 1. Comparison of the instance 4*5

	Makespan	E/T penalty	Max Workload
AL+CGA	16	-	10
PSO+TS	11	-	10
XING	12	-	8
P-TS	11	1	10
	11	0	11
	11	2	9
	12	4	8

Table 2. Comparison of the instance 8*8

	Makespan	E/T penalty	Max Workload
AL+CGA	15	-	13
PSO+SA	15	-	12
	16	-	13
PSO+TS	14	-	12
	15	-	12
XING	14	-	12
	15	-	12
P-TS	14	9	12
	14	3	13
	14	2	14
	15	3	12
	16	11	11

Table 3. Comparison of the instance 10*7

	Makespan	E/T penalty	Max Workload
XING	11	-	11
	11	-	10
P-TS	11	36	11
	11	37	10
	12	26	11
	12	30	10
	12	24	12
	13	17	10
	13	14	11
	14	4	11
	14	3	12
	14	11	10
	15	3	11
	16	10	10

Table 4. Comparison of the instance 10*10

	Makespan	E/T penalty	Max Workload
AL+CGA	7	-	5
PSO+SA	7	-	6
PSO+TS	7	-	6
XING	7	-	6
	8	-	5
	8	-	6
P-TS	7	1	7
	7	2	6
	7	4	5
	8	3	5

To our best knowledge, there not exit any literature which has given the experiment results of the above five instances with E/T penalty. Therefore, we list our experimental results paralleled with the other existing algorithms which have given the results with minimization of the total workload instead of the E/T penalty. Table 1 to table 5 give the comparison results of the above five instances. From these tables, we can see that our proposed algorithm can always get the optimal two objective values. In addition, the proposed P-TS algorithm can also obtain more non-dominated solutions in very short time. It can be seen from the above five tables that our proposed P-TS algorithm can either obtain rich Pareto optimal solutions or dominated solutions than the other existing algorithms.

Table 5. Comparison of the instance 15*10

	Makespan	E/T penalty	Max Workload
AL+CGA	23	-	11
	24	-	11
PSO+SA	12	-	11
XING	11	-	11
	12	-	11
PSO+TS	11	-	11
P-TS	11	9	11
	12	7	11

5 Conclusion

In this paper, we propose a Pareto-based TS algorithm for multi-objective FJSP with Earliness/Tardiness penalty. In the hybrid algorithm, several neighboring structure based approaches were proposed to improve the convergence capability of the algorithm while keep population diversity of the last Pareto archive set. In addition, an external Pareto archive was developed to record the non-dominated solutions found so far. In the hybrid algorithm, dynamic parameters were introduced to adapt to the searching process. Experimental on several well-known benchmark instances show that the proposed algorithm is superior to several existing approaches in both solution quality and convergence ability.

Acknowledgments. This work supported by National Science Foundation of China under Grants 60874075, 70871065, 60905039, Science Research and Development of Provincial Department of Public Education of Shandong under Grant J08LJ20, J09LG29, J08LJ59, Soft Science Foundation of Shandong under Grant 2009REB125.

References

1. Brandimarte, P.: Routing and scheduling in a flexible job shop by tabu search. Annals of Operations Research 22, 158–183 (1993)
2. Gao, L., Peng, C.Y., Zhou, C., Li, P.G.: Solving flexible job shop scheduling problem using general particle swarm optimization. In: The 36th CIE Conference on Computers & Industrial Engineering, pp. 3018–3027 (2006)
3. Liouane, N., Saad, I., Hammadi, S., Borne, P.: Ant Systems & Local Search Optimization for Flexible Job-Shop Scheduling Production. International Journal of Computers, Communications & Control 2, 174–184 (2007)
4. Gao, J., Sun, L., Gen, M.: A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems. Computers & Operations Research 35(9), 2892–2907 (2008)

5. Kacem, I., Hammadi, S., Borne, P.: Pareto-optimality approach for flexible job-shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic. Mathematics and Computers in Simulation 60, 245–276 (2002a)
6. Kacem, I., Hammadi, S., Borne, P.: Approach by localization and multi-objective evolutionary optimization for flexible job-shop scheduling problems. IEEE Transactions on Systems, Man and Cybernetics, Part C 32(1), 408–419 (2002b)
7. Xia, W.J., Wu, Z.M.: An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems. Computers and Industrial Engineering 48(2), 409–425 (2005)
8. Zhang, G.H., Shao, X.Y., Li, P.G., Gao, L.: An effective hybrid swarm optimization algorithm for multi-objective flexible job-shop scheduling problem. Computers and Industrial Engineering 56(4), 1309–1318 (2009)
9. Ho, N.B., Tay, J.C.: Solving Multiple-Objective Flexible Job Shop Problems by Evolution and Local Search. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 38(5), 674–685 (2008)
10. Chu, C.: A branch and bound algorithm to minimize total tardiness with different release dates. Naval Research Logistics 39, 265–283 (1992)
11. Souissi, A., Kacem, I., Chu, C.: New lower bound for minimizing total tardiness on a single machine with sequence dependent setup times. In: PMS 2004, Nancy, April 26-28 (2004)
12. Kanet, J.J., Hayya, J.C.: Priority dispatching with operation dues datas in job-shop. Journal of operations Management 2, 167–165 (1982)
13. Vinicius, A., Cintia, R.: Tabu search for minimizing total tardiness in a job-shop. International journal of Production Economics 63, 131–140 (2000)
14. Zribi, N., Kacem, I., El-Kamel, A., Borne, P.: Ecole Centrale de Lille, France, `http://www.wacong.org/wac2006/allpapers/isiac/isiac_45.pdf`
15. Du, J., Leung, J.Y.-T.: Minimizing total tardiness on one machine is np-hard. Mathematics of Operations Research 15, 483–495 (1990)
16. Deb, K., Paratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-Π. IEEE Transactions on Evolutinary Computation 6(2), 182–197 (2002)
17. Pan, Q.K., Wang, L., Qians, B.: A novel differential evolution algorithm for bi-criteria no-wait flow shop scheduling problems. Computers & Operations Research 36(8), 2498–2511 (2009)
18. Li, J.Q., Pan, Q.K., Xie, S.X., Li, H., Jia, B.X., Zhaos, C.S.: An effective hybrid particle swarm optimization algorithm for flexible job-shop scheduling problem. MASAUM Journal of Computing 1(1), 69–74 (2009)
19. Pezzella, F., Morganti, G., Ciaschetti, G.: A genetic algorithm for the Flexible Job-shop Scheduling Problem. Computers and Operations Research 35, 3202–3212 (2008)
20. Xing, L.N., Chen, Y.W., Yang, K.W.: An efficient search method for multi-objective flexible job shop scheduling problems. Journal of Intelligent Manufacturing 20, 283–293 (2009b)

Research on Multi-objective Optimization Design of the UUV Shape Based on Numerical Simulation

Baowei Song, Qifeng Zhu, and Zhanyi Liu

Northwestern Polytechnical University, Xi'an, 710072, China

Abstract. The numerical simulation research of the Underwater Unmanned Vehicle (UUV) shell shape is carried out by applying the modern fluid dynamics numerical simulation technology. In the numerical simulation process, while focusing on the characteristics of the UUV shape, this treatise work properly with its computational model, computational domain, computational grid and boundary conditions. Based on numerical simulation techniques, the UUV shape is multi-objectively optimized by integrated fluid dynamics simulation software-fluent on iSIGHT optimization design platform. Multi-island genetic algorithm is adopted as the optimization algorithm, and Reynolds-averaged Navier-Stokes equation and turbulence model are as well applied in the optimization design process. The results show that by reducing the drag coefficient and the flow noise of the UUV shape, the optimized design has made a significant improvement in its comprehensive performance, which provides a new way for today's optimization design of UUV shape.

Keywords: Underwater Unmanned Vehicle (UUV) shape, numerical simulation, multi-objective optimization, iSIGHT, optimization design.

1 Introduction

Resistance and flow noise are two key technical performance indicators of underwater craft. Therefore, drag and noise reduction technology is one of the key technologies in underwater vehicles researches [1]. Theoretically, there are many ways to reduce the drag and flow noise of UUV, but they have limitations in engineering applications [2]. As the resistance and noise are two different mechanical phenomena, comprehensive optimal solution is not able to be achieved if we set both as the optimization objective at the same time. In traditional research of optimal design of underwater vehicle shape, one of them is set as the objective function while the other as the constraint condition. In this way, it is difficult to get the UUV shape integrated with best resistance performance and best noise performance. Some scholars have set to study the multi-objective optimization design of the UUV shape in these years. However, based on empirical calculating formula, the accuracy of calculation can not be guaranteed [3]. With the development of computational fluid dynamics, numerical simulation software has been widely applied in simulating calculation of fluid dynamics, and its accuracy has been verified [4-5]. Fusing optimization techniques and numerical simulation techniques, the optimal design of UUV shell shape based on numerical simulation techniques is the developing trend of the unmanned underwater vehicle (UUV) shell shape design.

In this paper, the numerical simulation technology has been applied in the multi-objective optimization design of the UUV shape for the first time, which helps to improve the dynamics performance and the acoustic performance of the UUV, thus extensively enhances its comprehensive performances. And this optimization method provides a new way for today's optimization design of UUV shape.

2 UUV Shell Shape Design

The outline of the UUV shell is shaped by the bus rotating around the axis. As its bus is called linear in geometry, so UUV shape design is linear design. UUV linear is generally composed of four sections: head curve segments, parallel middle, tail curve segments and caudal segment. The specific geometry and parameters are in Figure 1.

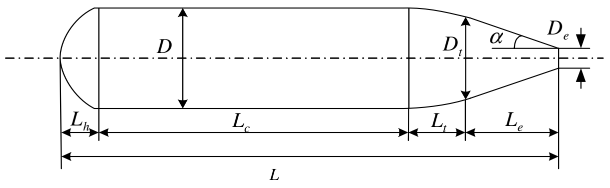


Fig. 1. Specific Geometry and Parameters of the UUV Shape

In this treatise, all of the UUV linear adopt the Granville linear [6]. Its head linear adopts the two-parameter square root of polynomial curve segments linear, and the linear is expressed as follows:

$$y = k_{s1}\sqrt{x} + \frac{x}{16}k_{s1}(5x^3 - 21x^2 + 35x - 35) + k_{s2}\frac{1}{6}x(x-1)^3 + 1 - (x-1)^4 \tag{1}$$

Parallel middle linear expression as follows:

$$y = D/2 \tag{2}$$

Using Granville Sharp-tailed linear as its tail curve segments linear, expression is as follows:

$$y = \frac{D}{2}\left[x^3\left(6x^2 - 15x + 10\right) - k_{s3}x^2\left(x-1\right)^3 - k_{s4}x^3\left(x-1\right)^2\right]^{1/2} \tag{3}$$

Caudal segment linear can be expressed as follows:

$$y = \frac{D_t}{2} - x\tan\alpha \tag{4}$$

3 Numerical Simulations

3.1 Control Equations

Taking available computing resources into account, in order to simulate the micro-flow field on the surface of the UUV shell and improve the accuracy of simulation as well; this paper adopts Reynolds-averaged Navier-Stokes equation and RNG $k-\varepsilon$ turbulence model method [7]. The specific mathematical model of the method is as follows:

Continuity equation:

$$\frac{\partial \overline{u_i}}{\partial x_i} = 0 \tag{5}$$

Reynolds-averaged Navier-Stokes equation:

$$\rho \overline{u_j} \frac{\partial \overline{u_i}}{\partial x_j} = \rho \overline{f_i} - \frac{\partial \overline{p}}{\partial x_i} + \frac{\partial}{\partial x_j}(\mu \frac{\partial \overline{u_i}}{\partial x_j} - \rho \overline{u_i^{'} u_j^{'}}) \tag{6}$$

Turbulent fluctuation kinetic energy equation (k equation) as follows:

$$\frac{\partial}{\partial t}(\rho k) + \frac{\partial}{\partial x_i}(\rho k u_i) = \frac{\partial}{\partial x_i}\left[\left(\mu_0 + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + P_k - \rho \varepsilon \tag{7}$$

Turbulent energy dissipation rate (ε equation) as follows:

$$\frac{\partial}{\partial t}(\rho \varepsilon) + \frac{\partial}{\partial x_i}(\rho \varepsilon u_i) = \frac{\partial}{\partial x_j}\left[\left(\mu_0 + \frac{\mu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right] + C_{\varepsilon 1}\frac{\varepsilon}{k}P_k - C_{\varepsilon 2}\rho\frac{\varepsilon^2}{k} - R_\varepsilon \tag{8}$$

3.2 Grid Generation

Considering the symmetry of the UUV, this paper selects a two-dimensional space computational domain along the flow direction, the length of the rectangular computational domain is three times of the shell length, and the width is 20 times of the shell radius. In order to avoid the effects of the entrance and the exit, 3.0m space before the entrance and 3.21m space before the exit are reserved. The final calculation region is shown in Figure 2.

In this paper, the computational grid is generated by the method of limited volume discretion. The UUV shell surface has an influence on the water downstream market mainly concentrated in the near wall region. In order to improve the calculation accuracy and save computing resources, the defined size function method is applied in the near wall region to make the grid around the rotating surface turn intensive and gradually become sparse outward.

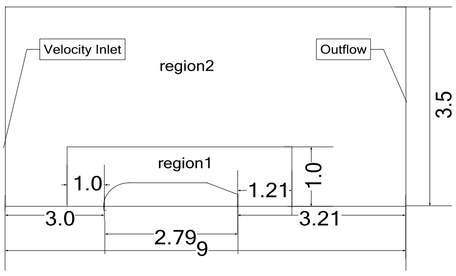


Fig. 2. Computational Domain

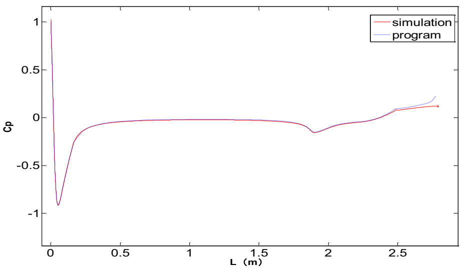


Fig. 3. Pressure Coefficient Distribution

3.3 Conditions of Solution Setting

The specific calculation conditions are set as follows:
Solver: steady-state segregated 2D implicit solver, using axial symmetry method. Turbulence equations: second-order equation.

Fluid medium: water, density of 998.2kg/m^3, dynamic viscosity coefficient is 0.001003Pa•s.

Boundary conditions: the entrance is velocity inlet with flow rate 20m/s, along the x-axis; the exit is outflow.

Discrete equation method: pressure-correction method uses Numerical SIMPLEC calculation method [8], and the discrete method of parameters uses the second-order accuracy upwind scheme.

Convergence criteria: the continuity equation residual is set to 1e-4, and 1e-5 for others.

3.4 The Numerical Simulation Results

After 2000-step iterative calculation, the equation convergence. To verify the accuracy of this numerical simulation method, this treatise makes a comparison of the pressure coefficient distribution calculated by numerical simulation with that of program in reference [9], which is shown in Figure 3.

Table 1. Comparison of Hydrodynamic Performance between Simulation and Program

Hydrodynamic parameters	Program results	Simulation results
Drag coefficient of the largest surface-area	0.00288	0.00282
Maximum decompression coefficient of the head curve	0.91624	0.91618
Location of the largest decompression coefficient of the head curve	0.05066	0.05132
Maximum decompression coefficient of the tail curve	0.15292	0.15701
Location of the largest decompression coefficient of the tail curve	1.89908	1.89796

From figure 3 and Table 1, we can see that this numerical simulation results are basically consistent with the theoretical calculations, so numerical simulation results are credible.

4 Multi-objective Optimization Design Model of UUV Shape

4.1 Design Variables

There are many design parameters of UUV shell shape, in this article, five design parameters are selected for the design variables as follows: adjustable parameters of the head linear: k_{s1} and k_{s2}, the adjustable parameters of tail linear: k_{s3} and k_{s4}, and, half-angle of caudal segment: α.

Value range of design variables is as follows: $0.001 \le k_{s1} \le 4.5$, $0.001 \le k_{s2} \le 12.0$, $0.001 \le k_{s3} \le 8.0$, $0.001 \le k_{s4} \le 2.9$, $8 \le \alpha \le 12$.

4.2 Constraints

The optimization design constraints of the UUV shell shape are mainly composed of geometric constraints and performance constraints.

The head shell should have a larger volume in order to meet the needs of ammunition and other internal devices installment, This requirement can be put forward by the head fullness factor, that is: $0.7 \le \Phi \le 1$.

In order to have a better hydrodynamic characteristic, the linear should not have cavitations phenomenon, that is to say $|C_{p\min h}| < \sigma_k/\eta$, σ_k is the Vacuole number of UUV sailing, η is 1.2.

In order to reduce the noise impact on the self-guided system, the largest coefficient location of the head should enlarge as much as possible, that is: $\xi X_{\min h0} \le X_{\min h}$, in which $X_{\min h0}$ is the design required minimum value for the largest coefficient location of the head, ξ is 1.2.

4.3 Objective Function

UUV shape design is an integrated multi-objective optimization design problem. Its objectives are: the minimum drag coefficient the UUV shape C_x, the maximum decompression coefficient in the head curve segment $|C_{p\min h}|$, the maximum location $X_{\min h}$ of the maximum decompression coefficient in head curve segment, the minimum decompression coefficient in the tail curve segment $|C_{p\min t}|$, and the maximum location $X_{\min t}$ of the maximum decompression coefficient in head curve segment.

The solving methods to the optimization problem of multi-objective function include: unification-object method, main object method, and coordination curve method, in this paper, we apply unification-object method to turn it into a single object, and use empirical estimation method to determine weighting factors, so that every target is in the same magnitude, the obtained objective function is as follows:

$$f = \min\left\{1000C_x + \frac{1}{X_{\min h}} + 10\left|C_{p\min h}\right| + \frac{20}{X_{\min t}} + 10\left|C_{p\min t}\right|\right\} \tag{9}$$

5 Optimization

5.1 Optimization Process

According to the requirements, we compile the generating program shapecreat.exe of the UUV linear coordinates. Then, introduce the file generated from the compiled program into GAMBIT, set the model, generate and output the Grid file. FLUENT reads the grid file, gets the solution through the parameter setting, and output the hydrodynamics report file. iSIGHT[10] analyzes the hydrodynamics report file generated, gets the resistance of the UUV shape, wet surface area, the maximum head decompression coefficients and their location and the maximum tail decompression coefficients and their locations. Then the value of the unified multi - target objective function calculate is calculated out and output. iSIGHT is applied to search and optimize the design variable space based on the optimization algorithm. The flow diagram is shown in Figure 4.

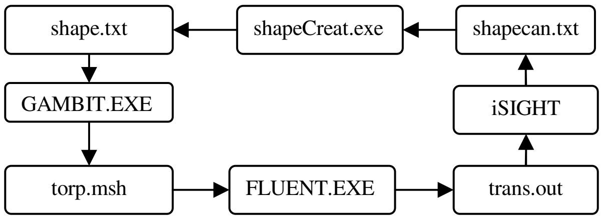
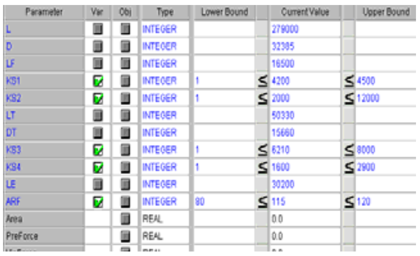


Fig. 4. Optimization Process

5.2 Optimal Design

According to mathematical model of optimal design, the initial conditions are shown in Figure 5, and the data transfer process is shown in Figure 6.



Parameter	Var	Obj	Type	Lower Bound		Current Value		Upper Bound
L			INTEGER			279000		
D			INTEGER			32385		
LF			INTEGER			16500		
KS1	✓		INTEGER	1	≤	4200	≤	4500
KS2	✓		INTEGER	1	≤	2000	≤	12000
LT			INTEGER			50330		
DT			INTEGER			15660		
KS3	✓		INTEGER	1	≤	6210	≤	8000
KS4	✓		INTEGER	1	≤	1600	≤	2900
LE			INTEGER			30200		
ARF	✓		INTEGER	80	≤	115	≤	120
Area			REAL			0.0		
PreForce			REAL			0.0		

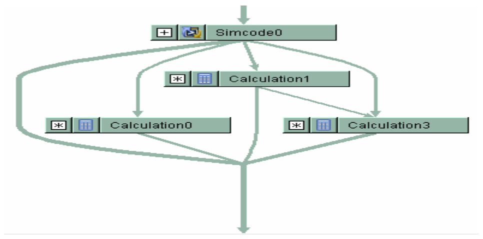


Fig. 5. Initial Design Variable Settings

Fig. 6. Data Transfer Process

UUV shape optimization design has five design variables, which belongs to the high-dimensional design space. Since it is difficult to search the optimal value, exploring optimization techniques should be adopted in order to facilitate the preliminary search of the optimal solution in the design space within global scope.

5.3 Comparative Analysis of the Results

In this paper, multi-island genetic algorithm is used for multi-objective optimal design of UUV shape, and through 1000-step optimization calculating, the optimal solution is finally obtained.

Table 2. Comparison of Design Variables and Optimization Objects

Design variable	k_{s1}	k_{s2}	k_{s3}	k_{s4}	α	f
Original shape	4.2	2.0	6.21	1.60	11.5	43.575
Optimized shape	3.2	3.8	7.96	2.55	8.2	34.406

Table 3. Comparison of Hydrodynamic Performance Parameters

Hydrodynamic parameters	Original shape	Optimized shape
Drag coefficient of the largest surface-area	0.00282	0.00276
Maximum decompression coefficient of the head curve	0.91618	0.69455
Location of the largest decompression coefficient of the head curve	0.05132	0.07014
Maximum decompression coefficient of the tail curve	0.15701	0.17467
Location of the largest decompression coefficient of the tail curve	1.89796	2.30011

As it can be seen from Table 2 and Table 3, the optimal design of the UUV shape has achieved the blow effects: with the drag coefficient significantly reduced, the hydrodynamic performance of the UUV is improved; the maximum decompression coefficient of the head curve significantly reduces and its location has a certain shift backward; in addition, the maximum decompression coefficient of tail curve has very slight increase and its location has a evident shift; as a result, the noise performance of the UUV is also improved.

6 Conclusions

Numerical simulations are carried out on fluid dynamic characteristics of the UUV shape, and through theoretical calculations, the accuracy of the numerical simulation results has been verified. Focusing on the multi-objective characteristics of UUV shape design, this treatise has established the mathematical model for the UUV shape multi-objective optimization design aiming at resistance and noise. Based on numerical simulation technology, the UUV original shape is optimal designed with its resistance performance and acoustic performance improved and its comprehensive performances remarkably enhanced, which indicates the feasibility of the multi-objective optimization design method of the UUV shape in this paper.

References

1. Huang, J.G., Zhang, Y.W.: Torpedo hydrodynamics. Northwestern Polytechnical University Press, Xi'an (1989)
2. Walsh, M.J.: Riblets as a viscous drag reduction technique. AIAA(S0001-1452) 21(12), 485–486 (1983)

3. Du, X.D., Shi, X.H., Liu, C.D.: Optimal Design of Cylindrical Shell of Underwater Unmanned Vehicle (UUV). Mechanical Science and Technology 24(8), 988–990 (2005)
4. Li, X.H., Pan, G., Song, B.W., Hu, H.B., Li, J.W.: A study of autonomous underwater vehicle hull form using computational fluid dynamic. Machinery Design And Manufacture 8, 24–26 (2006)
5. Hu, H.B., Pan, G., Song, B.W., Liu, Z.Y.: Study on Hydrodynamic Performance Simulation of Underwater Gyroidal Objects Shape. Fire Control and Command Control 33(9), 143–145 (2008)
6. Granville, P.S.: Geometrical characteristics of streamlined shapes. Journal of Ship Research 13(4), 12–20 (1969)
7. Xu, J.G.: Algebraic Stress Model with RNG Equation for Simulating Confined Strongly Swirling Turbulent Flows. Journal of Thermal Science 15(1), 14–19 (2001)
8. Wang, F.J.: Computational Fluid Dynamics Analysis. Tsinghua University Press, Beijing (2004)
9. Zhang, Y.W.: Torpedo shape design. Northwestern Polytechnical University Press, Xi’an (1998)
10. Yu, X.Q., Yao, W.X., Xue, F., Mu, X.F., Liu, K.L., Huang, A.F.: A Study on the Requirements for the Framework of Multidisciplinary Design Optimization. Mechanical Science and Technology 23(3), 286–289 (2004)

Multi-Objective Optimization for Massive Pedestrian Evacuation Using Ant Colony Algorithm

Xinlu Zong[1], Shengwu Xiong[1,*], Zhixiang Fang[2], and Qiuping Li[2]

[1] School of Computer Science and Technology,
Wuhan University of Technology, Wuhan 430070, China
xiongsw@whut.edu.cn
[2] State Key Laboratory of Information Engineering in Surveying,
Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China

Abstract. Evacuation route planning is one of the most crucial tasks for solving massive evacuation problem. In large public places, pedestrians should be transferred to safe areas when nature or man-made accidents happen. A multi-objective ant colony algorithm for massive pedestrian evacuation is presented in this paper. In the algorithm, three objectives, total evacuation time of all evacuees, total routes risk degree and total crowding degree are minimized simultaneously. Ants search routes and converge toward the Pareto optimal solutions in the light of the pheromone. The experimental results show that the approach is efficient and effective to solve massive evacuation problem with rapid, reasonable and safe plans.

Keywords: Multi-objective optimization, Evacuation, Ant colony optimization.

1 Introduction

In recent years, frequent natural disasters and man-made catastrophic events, such as floods, fires, hurricanes or terrorist attacks, have lead to serious results to human beings, especially for those large public places, such as gymnasiums, stations. Thus, massive evacuation has been attached great importance for handling emergency situation. The modeling and simulation of emergency evacuation is attracting widespread attention of researchers [1]–[2].

Evacuation planning is a very complex problem involving both features of individuals and environment. A series of models [3]–[4] were proposed for evacuation simulation. For massive evacuation, several factors need to be taken into account simultaneously. Most of relevant research works took single factor as the most important objective to achieve and ignored other factors which influence on evacuation results. Few studies [5]–[6] on evacuation route planning considered optimizing multiple objectives. In fact, evacuation route planning is a multi-objective optimization problem (MOP) which is a very important research topic due to the multi-objective nature of most real-world problems.

[*] Corresponding author.

In this paper, a multi-objective ant colony algorithm for massive emergency evacuation is presented. In this algorithm, three objectives, total evacuation time of all evacuees, total risk degree and total crowding degree of all routes are optimized simultaneously. The experimental results show that the proposed approach is efficient and effective to solve massive evacuation problem with rapid and safe plans.

2 Problem Formulation

2.1 Definition of Variables and Parameters

An emergency evacuation network is defined as a directed graph $G(N, A)$, where $N = \{1,2,...,n\}$ is the set of nodes and $A \subseteq N \times N$ is the set of arcs. The number of pedestrians is M . Denote t_{ij}^k as the traveling time through arc (i, j) of pedestrian k under emergency situation. r_{ij} is the risk degree on (i, j). c_{ij}^k is the crowding degree when pedestrian k passes (i, j). l_{ij} is the length of (i, j), and d_{ij} is the distance form (i, j) to dangerous spot. P_k is the evacuation path of pedestrian k , and i_k is the initial node of pedestrian k , where $i_k \in N$. $v_{ij}^k(t)$ is the passing speed of pedestrian k through (i, j) , $v_{ij}(0) = 1\, m/s$ is pedestrians' speed under normal conditions. N_{ij}^k is the number of pedestrians on (i, j) when pedestrian k passes. C_{ij} is the passing capacity of (i, j) .

2.2 Mathematical Formulation

The multi-objective model to solve massive evacuation problem is described below:

$$\min f_1 = \sum_{k=1}^{M} \sum_{i=i_k}^{j \in P_k} t_{ij}^k \tag{1}$$

$$\min f_2 = \sum_{k=1}^{M} \sum_{i=i_k}^{j \in P_k} r_{ij} \tag{2}$$

$$\min f_3 = \sum_{k=1}^{M} \sum_{i=i_k}^{j \in P_k} c_{ij}^k \tag{3}$$

Subject to:

$$r_{ij} = \frac{1}{d_{ij}} \tag{4}$$

$$c_{ij}^k = \frac{N_{ij}^k}{l_{ij}} \tag{5}$$

$$v_{ij}^k(t) = v_{ij}(0) \cdot e^{-(k_1 c_{ij}^k + k_2 r_{ij}) \cdot t} \tag{6}$$

$$\int_0^{t_{ij}^k} v_{ij}^k(t)dt = l_{ij} \tag{7}$$

$$\frac{N_{ij}^k}{C_{ij} t_{ij}^k} \le 1 \tag{8}$$

The objective (1) is to minimize total evacuation time, objective (2) is to minimize total risk degree of all routes, objective (3) is to minimize total crowding degree of all routes. Constraint (4) is the formula of risk degree along a road segment. Constraint (5) is the equation of crowding degree along a road segment. Constraint (6) is the decrease function of the evacuation speed on (i, j) of pedestrian k. Constraint (7) describes the relationship between passing speed and length form node i to j. Constraint (8) ensures that the number of pedestrians on (i, j) at time t will not exceed its capacity.

3 Multi-objective Optimization for Massive Evacuation

The ant colony optimization algorithm was first proposed by Dorigo [7]. It is used to solve combinatorial optimization problems. The ant colony optimization imitates the behavior of real ants when finding food. During the process, ants will collect and store information in pheromone. Pheromone will be released by ants while finding shorter path. In addition, the pheromone will be evaporated over time. As a result, all the ants will be attracted to the shortest path due to the positive feedback mechanism.

The multi-objective ant colony optimization algorithm for massive pedestrian evacuation problem can be described as follows:

Step 1: Initialization.

Step 1.1: Set the values of fundamental parameters of the algorithm, including the maximum number of iterations $T\_\max$, the number of ants M.

Step 1.2: For every arc set an initial pheromone value $\tau_{ij}(0) = C$ (C is a small positive number) and set the initial pheromone updating value $\Delta\tau_{ij} = 0$.

Step 1.3: Set non-dominated solutions set $ND - set = \Phi$.

Step 1.4: Randomly select a node as the point at which emergency event occurs. Place all the M ants on initial nodes randomly.

Step 2: Construct a path for each ant from the initial node to one of any exits.

Step 2.1: Determine the current allowed list of the ant k, here $allowed_k$ is denoted as the set of nodes that the ant k can visit.

Step 2.2: At time t, ant k move from node i to node j with the probability:

$$P_{ij}^{k}(t) = \begin{cases} \dfrac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum\limits_{s\in allowed_k} \tau_{is}^{\alpha}(t)\eta_{is}^{\beta}(t)} & if\ \dfrac{N_{ij}(t)}{C_{ij}t} \le 1\ and\ j \in allowed_k \\ 0 & Otherwise \end{cases} \tag{9}$$

Where α, β are parameters that weight the relative importance of pheromone and heuristic information. $\tau_{ij}(t)$ is the amount of pheromone on arc (i,j), and $\eta_{ij}(t)$ is a heuristic function, it can be calculated as follows:

$$\eta_{ij}(t) = \frac{C_{ij}}{t_{ij} \cdot r_{ij}} \tag{10}$$

Here, t_{ij} is the required time passing through (i,j) under normal conditions.

Step 3: Repeat Step 2 until all the M ants have constructed routes.

Step 4: ND-set determining.

Step 4.1: Denote $s_t = \{p_1, p_2, ..., p_M\}$ as the set of routes for all ants by Step 2, and one set represents a feasible solution, here p_k is a path constructed by ant k.

Step 4.2: For feasible solution s_t, calculate its objective functions f_1, f_2, f_3, if s_t is a non-dominated solution of the current non-dominated set, $\mathrm{ND\text{-}set} = \mathrm{ND\text{-}set} \cup \mathrm{s_t}$, and remove the solutions that dominated by s_t.

Step 5: Pheromone updating.

The pheromone on each arc is updated as follows:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \rho\Delta\tau_{ij} \tag{11}$$

$$\Delta\tau_{ij} = \begin{cases} \dfrac{Q}{\sum\limits_{k=1}^{M} t_{ij}^{k} \sum\limits_{k=1}^{M} r_{ij} \sum\limits_{k=1}^{M} c_{ij}^{k}} & if\ ant\ k\ passed\ (i,j)\ in\ s_t \\ 0 & Otherwise \end{cases} \tag{12}$$

Where ρ $(0 \le \rho \le 1)$ is the evaporation rate of pheromone, Q is a given constant.

Step 6: Stopping Criteria.

If $T < T\_\max$, every ant returns to its initial node, $T+1 \to T$,and go to Step2, else the algorithm is terminated.

4 Experimental Results

In order to solve massive evacuation problem, this multi-objective optimization model was applied to an area of Wuhan, China. Fig.1 shows the study area in the form of network with 476 nodes in total. The center of the area is a stadium which has 42 bleachers and 157 nodes in total including bleachers, stairs, exits and passages. The other 319 nodes represent roads outside. Suppose the emergency event has occurred in the node as marked in Fig.1, at beginning all pedestrians are placed in the center area, and aim to search their routes to one of the 8 exits.

Experiments were done to analyze if the presented approach can tackle and optimize routes for massive evacuation problem effectively. The multi-objective ant colony optimization algorithm was implemented using MATLAB coding and run on a PC with 3.06GHz, 1 GB RAM. In the algorithm, the parameters used were set as follows: $\alpha = 1$, $\beta = 3$, $\rho = 0.7$, $Q = 100$, $k_1 = 1$, $k_2 = 1$.

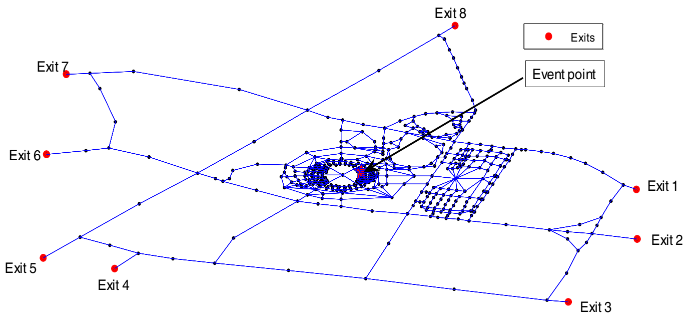


Fig. 1. The road network of study area

Table 1 lists the results obtained by experiments with different number of ants and iterations respectively. It can be seen that with the same number of ants, values of the three objective functions are decreased as the number of iteration increases, thus indicating that the algorithm is achieving the approximate global optimums.

Table 1. Results of the proposed approach

Ants	Iterations	f_1	f_2	f_3	Solutions
500	100	1.355e+006	5.2965	2638.7	9
500	200	1.3479e+006	4.7093	2632.7	13
1000	200	3.7835e+006	10.209	11402	20
2000	200	1.2003e+010	16.878	37625	15

Fig. 2 shows the distribution of non-dominated solutions with 2000 ants for 200 iterations. Each non-solution represents an evacuation routing plan. Comparing with single objective optimization, a set of non-dominated solutions can be obtained instead of single optimal solution. The trade-off among different objectives in Fig.2 can help decision makers choose an optimum evacuation routes plan according to different evacuation demands.

Table 2 lists the number of pedestrians evacuated from each exit with different total pedestrian number, 500, 1000 and 2000 respectively. The average percentage (AP) of total pedestrians for each exit is shown in Table 2. According to Fig. 1, most of pedestrians select Exit 1 to evacuate because it is the nearest exit, while only 3.62% pedestrians choose Exit 5 due to the long distance from initial positions. For the other exits, the percentage ranges from 5.63% to 20.73%, which means all exits are made fully use of during the evacuation process.

Table 2. Distribution of pedestrian number for each exit

Exits No.	Number of people			AP (%)
1	160	271	509	28.18
2	27	63	234	7.8
3	35	54	90	5.63
4	41	93	194	9.07
5	19	38	65	3.62
6	53	143	237	12.25
7	65	123	257	12.72
8	100	215	414	20.73
Total number	500	1000	2000	100

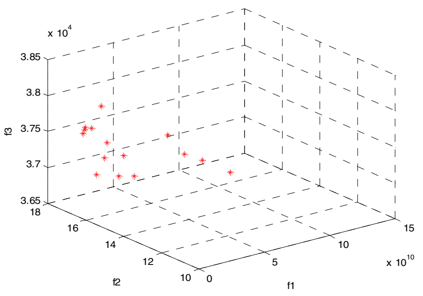


Fig. 2. The non-dominated solutions

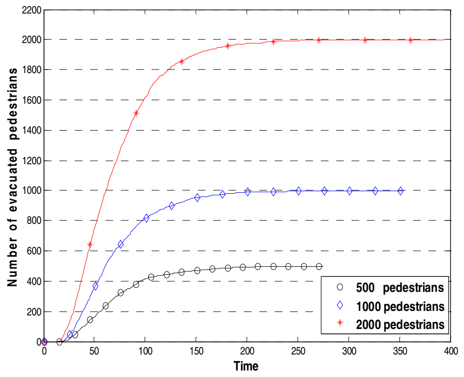


Fig. 3. Number of evacuated pedestrians versus time

Fig.3 depicts curves of the varying number of evacuated pedestrians over time with different number of evacuees. As the total pedestrian number increase, the evacuation time required increases. It is shown that at the first several time periods, there are no pedestrian reach exits of the area. There have been pedestrians evacuated after time period 15 and then the curves rise rapidly. The results demonstrate that the proposed approach is suitable for solving massive evacuation problem.

5 Conclusions

Massive pedestrian evacuation is a very complex problem because many aspects need to be considered in order to make a proper plan. Thus, different objectives should be optimized simultaneously while satisfy some constrains.

In this paper, an emergency evacuation model and a multi-objective ant colony algorithm for the model are presented. In this model, three objectives, total evacuation time of all evacuees, total routes risk degree and total crowding degree are optimized simultaneously. The experimental results show that the approach is effective to solve massive pedestrian evacuation problem with rapid and safe plans.

Acknowledgments. This work was supported in part by the National Science Foundation of China under grant no. 40701153, 40971233.

References

1. Parisi, D.R., Dorso, C.O.: Microscopic Dynamics of Pedestrian Evacuation. Physica A:Statistical Mechanics and Its Applications 354, 606–618 (2005)
2. Parisi, D.R., Dorso, C.O.: Morphological and Dynamical Aspects of the Room Evacuation Process. Physica A: Statistical Mechanics and Its Applications 385(1), 343–355 (2007)
3. Yuan, W.F., Tan, K.H.: An Evacuation Model Using Cellular Automata. Physica A 384, 549–556 (2007)
4. Li, X., Chen, T., Pan, L., Shen, S., Yuan, H.: Lattice Gas Simulation and Experiment Study of Evacuation Dynamics. Physica A 387, 5457–5465 (2008)
5. Saadatseresht, M., Mansourian, A., Taleai, M.: Evacuation Planning Using Multiobjective Evolutionary Optimization Approach. European Journal of Operational Research 198(1), 305–314 (2009)
6. Izquierdo, J., Montalvo, I., Pérez, R., Fuertes, V.S.: Forecasting Pedestrian Evacuation Times by Using Swarm Intelligence. Physica A 388, 1213–1220 (2009)
7. Dorigo, M., Caro, G.D., Gambardella, L.M.: Ant Algorithms for Discrete Optimization. Artificial Life 5(2), 137–172 (1999)

An Improved Immune Genetic Algorithm for Multiobjective Optimization

Guixia He[1], Jiaquan Gao[1], and Luoke Hu[2]

[1] Zhijiang College, Zhejiang University of Technology, Hangzhou 310024, China
[2] College of Mechanical Engineering, Zhejiang University of Technology, Hangzhou 310014, China
hegx_1022@163.com

Abstract. The study presents a novel weight-based multiobjective immune genetic algorithm(WBMOIGA), which is an improvement of its first version. In this proposed algorithm, there are distinct characteristics as follows. First, a randomly weighted sum of multiple objectives is used as a fitness function, and a local search procedure is utilized to facilitate the exploitation of the search space. Second, a new mate selection scheme, called tournament selection algorithm with similar individuals (TSASI), and a new environmental selection scheme, named truncation algorithm with similar individuals (TASI), are presented. Third, we also suggest a new selection scheme to create the new population based on TASI. Simulation results on three standard problems (ZDT3, VNT, and BNH) show WBMOIGA can find much better spread of solutions and better convergence near the true Pareto-optimal front compared to the elitist non-dominated sorting genetic algorithm (NSGA-II).

Keywords: immune genetic algorithm, multiobjective optimization, similar individuals.

1 Introduction

For multiobjective optimization problems (MOOPs), evolutionary algorithms (EAs) seem to be particularly suitable because they process a set of solutions in parallel, possibly exploiting similarities of solutions by recombination. The first step towards treating objectives separately in EAs was given by Schaffer(1985) [1]. His approach is known as the vector evaluation genetic algorithm (VEGA). Although there are some disadvantages for VEGA, it plays a great role in facilitating the development of multiobjective evolutionary algorithms (MOEAs). Many MOEAs have been presented since 1985, such as the weight-based genetic algorithm (WBGA) proposed by Hajela and Lin [2], Ishibuchi and Murata's random weight genetic algorithm (RWGA) [3], Srinivas and Deb's non-dominated sorting genetic algorithm (NSGA) [4], the strength Pareto evolutionary algorithm presented by Zitzler et al. [5], the elitist non-dominated sorting genetic algorithm (NSGA-II) by Deb et al. [6], and Knowles's Pareto archived evolutionary strategy (PAES) [7] and so on.

Among MOEAs, some approaches called the weight-based approaches use the weighted sum of objective function values as the fitness of a solution. To the weight-based MOEAs such as WBGA and RWGA, their advantages are low computation complexity. However, their main disadvantages are as follows: (1) difficulty in finding Pareto-optimal solutions in nonconvex problems, (2) lack of elitism for most cases, and (3) difficulty in generating uniformly distributed Pareto-optimal solutions. At present, we find that the weight-based MOEAs can alleviate or even overcome their disadvantages by improving their mate selection and environmental selection schemes. Thus, we present a novel weight-based MOEA (MOIGA) based on GA and immune theory in the literature[8]. Numerical experimental results show that the proposed algorithm can alleviate the above difficulty.

In this paper, we furthermore improve the proposed algorithm with the following aspects. (1) For the local search algorithm, the dominated relationship is added to determine whether the current solution v is replaced by a neighbourhood solution v^* besides utilizing their fitness. In particular, the dominated relationship is first considered. (2) TSASI is improved. (3) A new selection operator is presented to create the new population. Compared to the random selection approach, the proposed selection operator can better ensure diversity along the current non-dominated front. (4) The improved algorithm gives an approach to handle constrained conditions for MOOPs.

2 The Proposed Algorithm

Here, we will present the weight-based MOEA based on an immune operator and GA, which is an improvement of its first version. The main procedures are listed as follows.

(1) A random initial population P_0 of size N is created. Set the external archived set $Q_0 = \emptyset$ and its maximum size is $\bar{N}$. Set a counter $t = 0$.
(2) Combine P_t and Q_t and create a set $R_t = P_t \cup Q_t$. Calculate non-dominated solutions in R_t and then save them to the external archived set Q_{t+1}. If $|Q_{t+1}| > \bar{N}$, then perform TASI to reduce the size of Q_{t+1} to $\bar{N}$. If $|Q_{t+1}| < \bar{N}$, then select $\bar{N} - |Q_{t+1}|$ solutions from the dominated solutions in R_t by using the selection scheme based on TASI and add them to Q_{t+1}.
(3) Repeat the following procedures to select N pairs of parent solutions.
  (a) Specify the weight values $w_1, w_2, \cdots, w_m$ with a random uniform distribution, where $w_1 + w_2 + \cdots + w_m = 1$, and $w_i \in [0, 1], i = 1, 2, \cdots, m$.
  (b) Use TSASI to select a pair of parent solutions.
(4) Apply the simulated binary crossover operator (SBX) [9] to each of the selected N pairs of parent solutions with the predefined crossover probability. Two new solutions are generated from each pair of parent solutions. A new solution is selected from the two generated new solutions as the offspring according to their fitness. Then apply the polynomial mutation operator (PB) [9] to the generated new offspring with the predefined mutation probability, and add them to the population P_{t+1}.

(5) Apply the local search algorithm to all N solutions in the population P_{t+1}. The search direction of the local search for each solution is specified by the weight values in the fitness function by which its parent solutions were selected. The current population is replaced with the N solutions improved by the local search.
(6) If t can be exactly divided by N_{in} (number of inner loops), then apply the immune operator to the set $R_{t+1} = P_{t+1} \cup Q_{t+1}$.
(7) If $t > T$ (the maximum number of generations), then terminate the algorithm and output non-dominated solutions in the set R_{t+1}. Otherwise, $t = t + 1$ and return to (2).

In this improved algorithm, we only introduce the modified local search algorithm, the modified TSASI, the approach of handling constrained conditions, and a new selection operator. For the other aspects similar to its first version, please refer to the original literature [8].

2.1 Selection Scheme Based on TASI

A selection operator is important to a MOEA because it is responsible for guiding the selection process at the various stages of the algorithm toward a uniformly spread-out Pareto-optimal front. Here we present a new selection scheme based on TASI in order to preserve diversity of the created external archived set Q_{t+1}. The detailed procedures are listed as follows.

(1) If $|Q_{t+1}| > \bar{N}$, copy cells of Q_{t+1} to a set ϕ_{t+1}, and use TASI to reduce the size of ϕ_{t+1} to $\bar{N}$, and then let the external archived set $Q_{t+1} = \phi_{t+1}$. Terminate the procedure.
(2) If $|Q_{t+1}| < \bar{N}$, then let a set R_t be composed of dominated individuals among the external archived set Q_t and the population P_t in the t-th generation. Next, the set R_t is sorted for non-domination. Let us say that the number of non-dominated fronts in R_t is K. The maximum number of individuals allowed in the i-th front ($i = 1, 2, \ldots, K$) in the new external archived set Q_{t+1} of size $\bar{N}$ is calculated according to the following equation.

$$N_i = (\bar{N} - |Q_{t+1}|)\frac{1 - r}{1 - r^K} r^{i-1}, \tag{1}$$

where $0 < r < 1$.
(3) Assume that N_i^t is the number of individuals of the i-th front in R_t. If $N_1^t > N_1$ (that is, there are more individuals than allowed), we copy individuals of the first front in R_t to a set ϕ_{t+1} , and use TASI to truncate ϕ_{t+1} until its size is equal to N_1, and then add individuals in ϕ_{t+1} to Q_{t+1}. On the other hand, if $N_1^t \leqslant N_1$ (that is, there are less or equal number of individuals in R_t than allowed), we choose all N_1^t individuals and count the number of remaining slots $\delta_1 = N_1 - N_1^t$. The maximum allowed number of individuals in the second front is now increased to $N_2 + \delta_1$. Thereafter, the actual number of solutions N_2^t present in the second front is counted and is compared with N_2 as above. This procedure is continued until $\bar{N} - |Q_{t+1}|$ individuals are selected.

2.2 Tournament Selection Algorithm with Similar Individuals

Here we present the tournament selection algorithm with similar individuals (TSASI). The detailed procedures are listed as follows.

(1) For a given weight vector $\mathbf{w} = [w_1, w_2, \cdots, w_m]^T$, create two subpopulation $Pop1 = \emptyset$ and $Pop2 = \emptyset$. Their maximum sizes are $Popsize1$ and $Popsize2$, respectively.
(2) Use the binary tournament selection procedure to select an individual from Q_{t+1} and add it to $Pop1$.
(3) If $|Pop1| < Popsize1$, then return to (2). Otherwise, continue.
(4) Similarly, an individual is selected from Q_{t+1} by using the binary tournament selection procedure, and is added to $Pop2$.
(5) If $|Pop2| < Popsize2$, then return to (4). Otherwise, continue.
(6) The average objective vector in $Pop1$ is calculated, and then the most dissimilar solution among $Pop1$ from the average objective vector is chosen as **Parent A**.
(7) Respectively calculate the Euclidean distance in objective space and in decision space from **Parent A** to any solution in $Pop2$.
(8) Select the most similar solution from $Pop2$ as **Parent B**, whose Euclidean distance in decision space from **Parent A** must exceed a given small positive value δ_d.

It can be obviously seen that a pair of similar parent individuals, **Parent A** and **Parent B**, are chosen using TSASI, and TSASI can overcome the drawback that the same two individuals are chosen as a pair of parent individuals. The similarity in decision space between **Parent A** and **Parent B** can be controlled by modifying the parameter δ_d.

2.3 Local Search Algorithm

In this section, we discuss a local search algorithm that is adopted in order to quickly improve the quality of the population. The search direction of the local search for each solution is specified by the weight values in the fitness function. The local search algorithm is similar to that mentioned in the literature [3]. Here we modify it, and add the dominated relationship to determine whether the current solution v is replaced by a neighborhood solution v^* besides utilizing their fitness. Specially, the dominated relationship is firstly considered. The detailed procedure is as follows.

For each solution v in the population, do

(1) Examine a neighborhood solution v^* of the current solution v.
(2) Calculate objective values of v^* and v. If v^* dominates v, then $v = v^*$. Otherwise, evaluate the fitness of v^* and v. If the fitness of v^* is no worse than v, then $v = v^*$.

(3) If randomly chosen N_{neighbor} neighborhood solutions of the current solution v have been already examined (i.e., if there is no better solution among the examined N_{neighbor} neighborhood solutions of v), then end this procedure. Otherwise, return to (1).

A neighborhood solution $\mathbf{x}^*$ of a solution $\mathbf{x} = (x_1, \cdots, x_n)$ is defined as $\mathbf{x}^* = (x_1 + \Delta x_1, x_2 + \Delta x_2, \cdots, x_n + \Delta x_n)$, where Δx_i $(i = 1, 2, \cdots, n)$ is a random small number in the interval $[-(x_i^U - x_i^L), (x_i^U - x_i^L)]$ (with uniform probability distribution). x_i^L and x_i^U are lower and upper bounds of x_i, respectively. This algorithm is terminated if no better solution is found among N_{neighbor} neighborhood solutions that are randomly selected from the neighborhood of the current solution.

2.4 Constraint Handling

In this study, we mainly adopt the idea of the constrained tournament method to handle constraints. The following constrain-domination conditions are utilized to decide the relationship of domination for any two solutions $\mathbf{x}_p$ and $\mathbf{x}_q$.

Definition 1. ***(Constrained Domination):*** *A solution $\boldsymbol{x}_p$ is said to 'constrain-dominate' a solution $\boldsymbol{x}_q$ ($\boldsymbol{x}_p \prec \boldsymbol{x}_q$), if any of the following conditions are true:*

(i) *Solution $\boldsymbol{x}_p$ is feasible and Solution $\boldsymbol{x}_q$ is not.*
(ii) *Solutions $\boldsymbol{x}_p$ and $\boldsymbol{x}_q$ are both infeasible, but solution $\boldsymbol{x}_p$ has a smaller constraint violation.*
(iii) *Solutions $\boldsymbol{x}_p$ and $\boldsymbol{x}_q$ are both feasible and solution $\boldsymbol{x}_p$ dominate solution $\boldsymbol{x}_q$ in the usual sense (see Definition 1).*

The effect of using this constrained-domination principle is that any feasible solution has a better non-domination rank than any infeasible solution. All feasible solutions are ranked according to their non-domination level based on the objective function values. However, among two infeasible solutions, the solution with a smaller constraint violation has a better rank.

Based on this constrained-domination principle, the set of non-constrain-dominated solutions can be obtained. In addition, the constrained-domination principle can also be applied to TSASI. However, we modify the binary tournament selection approach used to choose individuals for creating subpopulation in TSASI as follows.

Definition 2. ***(Binary Constrained Tournament Selection):*** *Given two solutions $\boldsymbol{x}_p$ and $\boldsymbol{x}_q$, choose solution $\boldsymbol{x}_p$ if any of the following conditions are true:*

(i) *Solution $\boldsymbol{x}_p$ belongs to a better non-constrain-dominated set.*
(ii) *Solutions $\boldsymbol{x}_p$ and $\boldsymbol{x}_q$ belong to the same non-constrain-dominated set, but solution $\boldsymbol{x}_p$ has a better fitness value than solution $\boldsymbol{x}_q$.*

By the above definitions, it can be seen that this constraint handling strategy does not require any extra computational burden in addition to the constraint violation computations.

3 Numerical Experiment

In this section, we compare the performance of WBMOIGA with NSGA-II. All experiments are conducted on an IBM computer, which is equipped with a Pentium IV 2.8G processor and 512 MB of internal memory. The operating system is Windows 2000 server and the programming language is C++. The compiler is Borland C++ 6.0.

3.1 Experimental Setting

The specification of parameters for all two algorithms is listed as follows. For WBMOIGA, the population size $N = 100$, the size of external archived set $\bar{N} = 200$, $Popsize1 = Popsize2 = 15$, the crossover possibility $p_c = 0.9$, the mutation possibility $p_m = 1/N_{\text{vars}}$, where N_{vars} is the number of decision variables of the problem, and the reduction rate $r = 0.95$.

For NSGA-II, we maintain the same parameters reported in its original literature [6], which include a population size of 100, a crossover rate of 0.9, and a mutation rate of $1/N_{\text{vars}}$, where N_{vars} = number of decision variables of the problem.

3.2 Performance Measures and Test Problems

Here two different measures (Spacing (**S**) and Generational Distance (**GD**)) are chosen [9]. In addition, we choose three test problems from a number of significant past studies in this area, which include ZDT3 suggested by Zitzler et al., Viennet's VNT, and Binh and Korn's BNH.

3.3 Experimental Results and Analysis

For all problems, the comparison of the pareto fronts produced by WBMOIGA and NSGA-II is shown in Figs.1-3, respectively. The values of the two metrics for all algorithms are respectively presented in Table 1.

For Figs.1-3, we can see that WBMOIGA shows better behavior than NSGA-II for all test problems. WBMOIGA can not only obtain better Pareto-optimal front, but also have a much better spread of solutions. All these observations are also confirmed by analyzing the Table 1. From Table 1, it can be found that WBMOIGA has a smaller value than NSGA-II for each metric. Therefore, we can affirm that WBMOIGA outperforms NSGA-II for all test problems.

Table 1. Results for test problems: mean value and standard deviation (σ)

Test problems	Algorithms	S(mean)	$S(\sigma)$	GD(mean)	$GD(\sigma)$
ZDT3	WBMOIGA	3.87E-03	1.14E-04	8.27E-05	1.62E-06
	NSGA-II	7.83E-03	7.21E-04	1.26E-04	1.03E-05
VNT	WBMOIGA	2.91E-03	1.35E-04	7.21E-04	4.16E-05
	NSGA-II	4.74E-02	9.63E-03	6.67E-03	5.32E-04
BNH	WBMOIGA	3.66E-01	3.27E-02	3.99E-03	7.00E-04
	NSGA-II	6.37E-01	3.29E-02	7.98E-03	1.47E-03

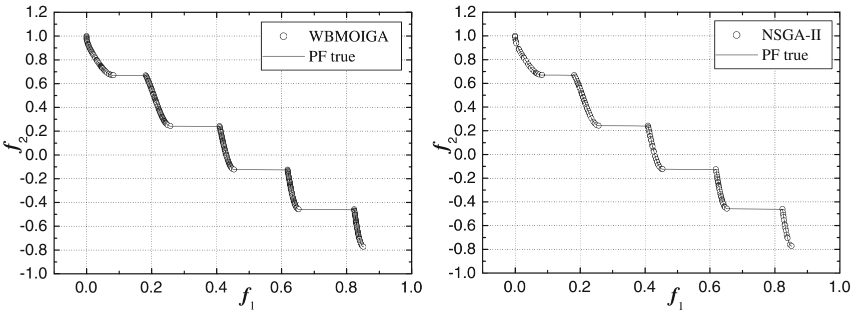


Fig. 1. Pareto-optimal front generated by WBMOIGA and NSGA-II for ZDT3

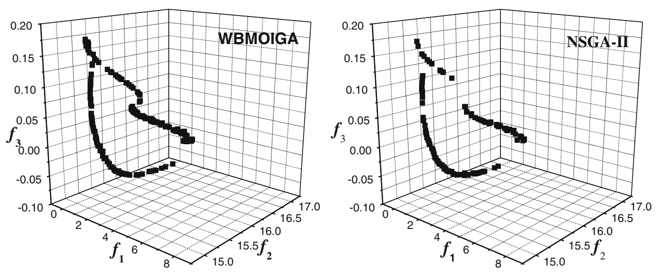


Fig. 2. Pareto-optimal front generated by WBMOIGA and NSGA-II for VNT

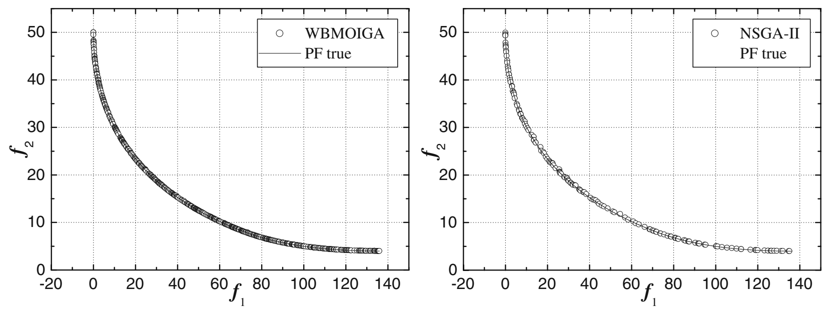


Fig. 3. Pareto-optimal front generated by WBMOIGA and NSGA-II for BNH

4 Conclusion

This study improves an immune genetic algorithm suggested in the literature[8]. Numerical results show that the modified algorithm WBMOIGA shows better behavior for all two metrics than NSGA-II for the test problems ZDT3, VNT, and BNH. Therefore, we can affirm that WBMOIGA outperforms NSGA-II, and can be applied to solve multiobjective optimization problems with constraints or without constraints.

References

1. Schaffer, J.D.: Multiple objective optimization with vector evaluated genetic algorithm. In: Proc. 1st Int. Conf. Genetic Algorithm, pp. 93–100. Hillsdale, New Jersey (1985)
2. Hajela, P., Lin, C.Y.: Genetic search strategies in multicriterion optimal design. Struct. Optimiz. 4, 99–107 (1992)
3. Ishibuchi, H., Murata, T.: A multi-objective genetic local search algorithm and its application to flowshop scheduling. IEEE Trans. Sys. Man Cy. 28(3), 392–403 (1998)
4. Srinivas, N., Deb, K.: Multiobjective optimization using nondominated sorting in genetic algorithms. Evol. Comput. 2(3), 221–248 (1994)
5. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. IEEE Trans. Evol. Comput. 3(4), 257–271 (1999)
6. Deb, K., Pratap, A., Agarwal, S., et al.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. 6(2), 182–197 (2002)
7. Knowles, J.D., Corne, D.W.: Approximating the nondominated front using the pareto archived evolution strategy. Evol. Comput. 8(2), 149–172 (2000)
8. He, G.X., Gao, J.Q.: A novel weight-based immune genetic algorithm for multiobjective optimization. In: Yu, W., He, H., Zhang, N. (eds.) ISNN 2009. LNCS, vol. 5552, pp. 500–509. Springer, Heidelberg (2009)
9. Deb, K.: Multi-objective Optimization Using Evolutionary Algorithms. John Wiley & Sons, Ltd., New York (2001)

Enhanced Mapping of Multi-robot Using Distortion Reducing Filter Based SIFT

Kyung-Sik Choi[1], Yoon-Gu Kim[2], Jinung An[2], and Suk-Gyu Lee[1]

[1] Department of Electrical Engineering, Yeungnam University
[2] Daegu Gyeongbuk Institute of Science and Technology
214-1 Daedong Gyongsan Gyongbuk Republic of Korea
robotics@ynu.ac.kr, {ryankim9,robot}@dgist.ac.kr, sglee@ynu.ac.kr

Abstract. This paper proposes an enhanced mapping of multi-robot using a DSIFT to reduce the mapping calculation time. In this approach, the master robot transmits each robot's mapping information in SLAM by DSIFT, which incorporates an additional step on the SIFT. The DSIFT uses a keypoint to reduce the distortional information throughout the Gaussian filter after the step of the image descriptor. The master robot calculates the slave robot's pose using picture images, and serves the results to all the robots. Simulation results are presented based on DSIFT showing better performance than using the SIFT in multi-robot mapping situations.

Keywords: Multi-robot, SIFT, DSIFT, Mapping.

1 Introduction

A multi-robot system can provide redundancy, and can contribute by the robots working cooperatively to solve an assigned task in a more reliable, faster or cheaper way. Since the cooperation is based upon the interchange of information relating to position and force sensing of the robots, the operative systems must integrate several characteristics to make this communication easier and effective. Also, the force feedback must come from a reliable source, so the sensors should be robust enough to provide this information. To enhance its performance, the fusing of the sensor data or sensors with high performances need to be employed.

In mobile robotics, simultaneous localization and mapping (SLAM) is essential for navigation, to perform given tasks by observing internal and external states. Various researchers have investigated the navigation of mobile robots based on vision systems, which include methods of observation and the recognition of beacons and obstacles in the real environment [1]-[4]. As used in this paper, a vision sensor has advantages over other sensors for detecting beacons, obstacles and other robots, in terms of high performance and low cost.

In a multi-robot system, each robot should recognize various features in its environment, including the other cooperative robots for performing the specified tasks. In this paper the SLAM, based on the scale invariant feature transform (SIFT) approach is utilized for enhancing the mapping performance of the multi-robots, using the vision sensor. An independent robot may have a heavy burden for the calculation time to perform its navigation based on the SLAM results. However, multi-robot can reduce this processing time through one robot serving the mapping information of the other robots using a distortion reducing filter based SIFT (DSIFT). In this case, the master robot must be equipped with a high performance computing system, but the other robots can manage with more minimal systems for performing the localization and navigation tasks, along with a communicating device. One of the most important contributions of this paper is the development of an enhanced mapping algorithm for multi-robot, which can be used for exploring unknown environments.

The next section discusses some related works described in the literature where the DSIFT method developed in this work is described and this is followed by section 4 which presents simulation studies and discusses the details of the new proposed algorithm. Section 5 presents the conclusions of the paper and what the future work in this area is likely to be.

2 Related Works

For landmark based navigation of a mobile robot, it is fundamental that the robot can recognize the features of the landmarks in its operational environment. In a natural environment, the detection of natural landmarks using sensors is very difficult, partly because of complicated and possibly indistinguishable features. Vision sensors have been most popular for this purpose, although they are subject to noise and variations of the light. Se and Lowe (2002) have used high-level image features, which are scale invariant, for greatly facilitating the feature correspondence. The features are distinctive and therefore their maps allow efficient algorithms to tackle the "kidnapped robot" problem [1]. Joo and Lee (2008) attempt object recognition based on the SIFT and binary searching tree for reducing the searching time of recognizing a required object, since the SIFT has to perform so many iterative operations [3]. Ahn *et al* (2008) have proposed a VR-SLAM (Vision and Range sensor-SLAM) algorithm to combine ultrasonic sensors with a stereo camera [5]. Gwak et al (2009) have used the SIFT and developed a feature point matching filter for rejecting mismatched points in real-time for the inertial navigation system and for the vision based SLAM in unstructured environments [6]. Finally, Gao *et al* (2008) proposed an approach of road crossing scene recognition, based on the SIFT and its color features. This method can reduce the processing time through cross checking scene images classified by threshold. The SIFT features of images in a road crossing image database use the K-D trees algorithm and the Bhattacharyya distance for calculating with the color histogram [7].

3 Distortion Reducing Filter Based SIFT

The SIFT was developed by Lowe for image feature generation in object recognition applications This is invariant to image translation, scaling, rotation, and is also partially invariant to illumination changes and is affine. Recently, the SIFT has become popular in VSLAM. Landmark detection is very important in the process of SLAM because robots navigate under varying environmental conditions using landmark detection with different distances, angles and levels of illumination. The features are highly distinctive, which allow a single feature to be correctly matched, with high probability against a large database of features, providing a basis for object and scene recognition [8]-[10]. However, SIFT has drawbacks, including wrong keypoints in the complex frames, although it is excellent in scaling and rotating images. So, the DSIFT is proposed, with keypoint reduced distortional information using Gaussian filters after the image descriptor step. The DSIFT can be used more effectively in case of more peculiar matches, and DSIFT method has added a step for removing the distortion to enhance the original SIFT method.

3.1 Detection of Scale Space Extrema

The first step in the SIFT is the detection of scale space extrema, which search over all scales and locations in the image. It is implemented efficiently by using a difference of Gaussian (DOG) function which identifies the potential interest points, which are invariant to scaling and orientation. This step can be repeatedly assigned under different views of the same object. The scale space can present the Gaussian as a function, $L(x, y, \sigma)$ which is produced from the convolution of a variable-scale Gaussian, $G(x, y, \sigma)$, with an input image, $I(x, y)$, as shown in Eqs. (1) and (2).

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{1}$$

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \tag{2}$$

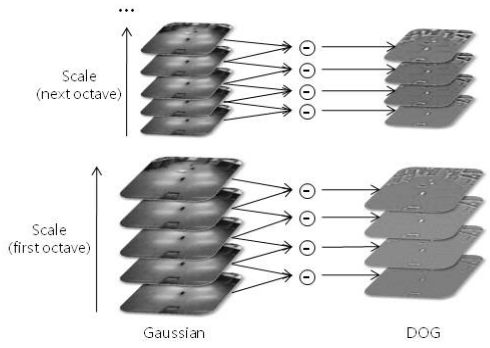


Fig. 1. For each octave of the scale space, the initial image is repeatedly convolved with Gaussians to produce the set of scale space images shown on the left. The adjacent Gaussian images are subtracted to produce the DOG images on the right.

This method is able to efficiently detect stable keypoint locations in the scale space, using scale-space extrema with the convolved DOG function and the input image, as in Eq. (3).

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \tag{3}$$

The $D(x, y, \sigma)$ can be computed from the difference of two nearby scales, separated by a constant multiplicative factor k, which usually uses a $\sqrt{2}$ multiplier. It searches for maxima and minima, which are the candidate points, from the DOG using the scale normalized Laplacian of the Gaussian, $\sigma^2\nabla^2 G$.

$$\frac{\partial G}{\partial \sigma} = \sigma\nabla^2 G \approx \frac{G(x, y, k\sigma) - G(x, y, \sigma)}{k\sigma - \sigma} \tag{4}$$

3.2 Keypoint Localization

A keypoint is chosen by determining a stable point of a location, scale and contrast. If keypoints have points with a low contrast they need to be removed because they are sensitive to noise. For the origin shift on the sample point, the Taylor series expansion is used, as proposed by Lowe [8], and the result is shown in Eq. (5).

$$D(X) = D + \frac{\partial D^T}{\partial X} X + \frac{1}{2} X^T \frac{\partial^2 D}{\partial X^2} X \tag{5}$$

where D and its derivatives are evaluated at the sample point, and $X = (x, y, \sigma)^T$ is the offset from this point. The extremum location, $\hat{X}$ is determined by taking the derivative of this function with respect to X, and setting it to zero.

$$\hat{X} = -\frac{\partial^2 D}{\partial X^2} \frac{\partial D}{\partial X} \tag{6}$$

For the stability of the keypoints, the method needs to remove keypoints with low contrast and have poorly defined peaks in the DOG, because the DOG has a strong response along its edges, although the location along the edge is poorly determined and unstable to small amounts of noise. A calculation of a large principal curvature can be used as a 2×2 Hessian matrix, H.

3.3 Orientation Assignment

For DSIFT, image data is transformed relative to the assigned orientation, the scale, and the location of each feature so it needs to make one or more orientations with invariable keypoints on any transformation. The scale of the keypoint is used to select the Gaussian smoothed image, L, with the closest scale, using each keypoint localization based σ with 1.5 times, so that all the computations are performed in a scale invariant manner. Thereby, the gradient magnitude and the orientation can be derived as shown in Eqs. (8) and (9).

$$m(x, y) = \sqrt{(L(x+1, y) - L(x-1,))^2 + (L(x, y+1) - L(x, y-1))^2} \tag{7}$$

$$\theta(x, y) = \tan^{-1} \frac{L(x, y+1) - L(x, y-1)}{L(x+1, y) - L(x-1, y)} \tag{8}$$

An orientation histogram is formed from the gradient orientations of the sample points within a region around a keypoint. In addition, the largest value of an orientation histogram indicates the dominant directions of the local gradients.

3.4 Image Descriptor

The descriptor is used to divide the keypoints around which the local image gradients are measured at the selected scale in the region. A Gaussian weighting function with σ is used to assign a weight to the magnitude of each sample point and this Gaussian window is useful to avoid a sudden change in the descriptor, with small changes in the position of the window, and to give less emphasis to gradients which are far from the center of the descriptor. The keypoint descriptor allows for a significant shift in the gradient positions by creating orientation histograms over 4x4 sample regions. A gradient sample on the left can shift up to 4 sample positions, while still contributing to the same histogram on the right, thereby achieving the objective of allowing for larger local positional shifts. The descriptor is formed from a vector containing the values of all the orientation histogram entries.

3.5 Distortion Reduction

The SIFT contains noise in a complex environment. In order to apply SLAM on a multi-robot, the distortion reducing filter with σ_D is proposed here. The DSIFT search the matched keypoints to select the Gaussian smoothed image, L after finishing the matching step of the SIFT, so that the standard deviation is calculated as

$$\sigma_D = \sqrt{\frac{\sum_{k=1}^{n}(L(x,y) - L_{mean})^2}{n}} \tag{9}$$

where n is the quantity of the matched keypoints, $L(x,y)$, and L_{mean} are about the pose information of the keypoints in the image L and the mean of the image L, respectively. As a result, the distortions of the matching information are found, which have the position information in the L with a larger value than $2\sigma_D$ as in Eq. (11).

$$L_{conp}(x,y,\sigma) = L_{match}(x,y,\sigma) - L_{dist}(x,y,\sigma_D) \tag{10}$$

4 Experimental Results

To show the enhancement of the DSIFT form of the SIFT, the simulation source developed by Lowe is modified [8][9]. The simulations are performed for two robots under a fluorescent lamp. In the experiments, we assumed that the master robot only maps its environment in the SLAM, and transfers the results to the slave robot. In this paper, three images are used; the reference robot image, and two images taken by the vision sensor with several seconds of delay. All the images have pixel sizes as indicated in Table 1, and Fig. 3 show the SLAM results of the slave robot.

Table 1. Standard deviation and the change matched keypoints for each case

Case	Image	Axis	Standard deviation			Match	
			SIFT	DSIFT	Enhance	SIFT	DSIFT
1	First	x	3.029677	3.179746	0.047195	4	3
		y	6.100511	1.305966	3.671265		
	Second	x	4.3277	3.351941	0.291103	6	3
		y	5.406841	1.402692	2.854618		
2	First	x	46.96441	11.31786	3.149585	36	34
		y	96.6231	10.51131	8.192298		
	Second	x	87.0773	20.31388	3.286593	34	30
		y	129.9269	32.26318	3.027096		
3	First	x	59.99569	13.11619	3.574171	51	45
		y	128.1745	14.04402	8.126625		
	Second	x	49.36058	16.12468	2.061181	77	70
		y	103.8622	16.96237	5.123096		
4	First	x	6.974377	2.089724	2.337463	8	3
		y	5.802162	4.695387	0.235715		
	Second	x	7.226168	1.400735	4.158839	9	4
		y	8.180572	3.25509	1.513163		

The cases 1 and 4 have similar results in the heading angles, although they have different errors of pose as an enhancement of the standard deviations which show that each average is 1.72 and 2.06. In contrast, cases 2 and 3 show a higher enhancement as each enhanced average of the standard deviation is 4.41 and 4.72, and the result is dependent on the number of matched keypoints so that if there are more matched keypoints, a more accurate pose of the robots will be obtained.

The results show better performance using the DSIFT compared with the SIFT in terms of the slave robot's heading angle. In fact the heading angle using the DSIFT is very similar to the actual images, as shown Fig. 3. Case 4 shows the experimental

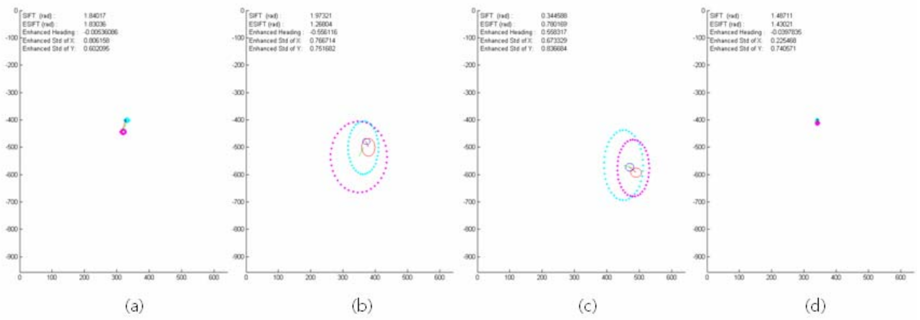


Fig. 2. (a)~(d) images are the navigational results for each case of slave robot. The dotted and solid lined ellipses are the cases of the SIFT and the DSIFT, respectively where the arrow head describes the movement of the slave robot.

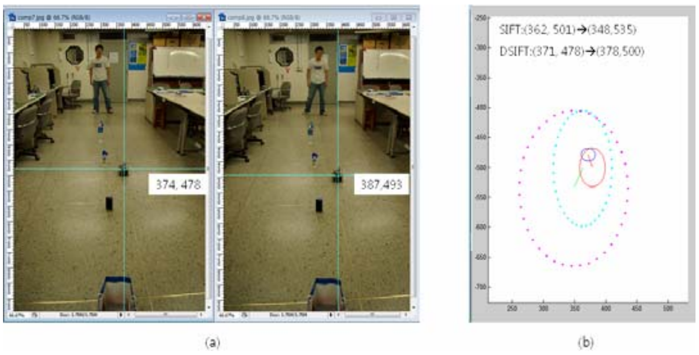


Fig. 3. (a) is show a movement of the slave robot and (b) is the result of the pose of the slave robot in case 2

results of the movement of two input images to confirm the effectiveness of the input time. According to the results, the master robot using more keypoints show more effective components than the input terms of the images taken. In this case, the master robot has a very similar result in terms of the heading angle; the first image has a 2.19% error while the second has a 1.64% error and the reduced errors of the slave robot's pose can be shown to be due to the proposed algorithm.

The number in Fig. 3(a) states the robot's position as a pixel unit in each image. The robot's pose in Case 4 is shown in Fig. 3 (b), where the dotted and solid ellipses depict the cases of the SIFT and the DSIFT approaches, respectively, and the arrow head denotes the motion direction of the robot. When the master robot uses the DSIFT, the heading angle of the slave robot is estimated more reliably, and with a lower standard deviation compared with the SIFT approach. This shows an enhancement of about 79.55% in the standard deviation, which is about 2 times more accurate in the heading angle compared with the SIFT approach. In addition, the accuracy of the robot's pose for the DSIFT and the SIFT methods are about 99.30% and 96.8%, respectively.

5 Conclusions

This paper has proposed an enhanced mapping for a multi-robot team with a vision system using a DSIFT to reduce the processing time in performing the mapping task. In this approach, the master robot transmits each robot's mapping in SLAM by the DSIFT, which has an additional step from the SIFT. The DSIFT uses a keypoint to reduce the distortion throughout the Gaussian filter after the image descriptor step. The master robot calculates the slave robot's pose using picture images, and the serves the results. In contrast to the SIFT, the proposed DSIFT incorporates an additional step in the SIFT. To verify the effectiveness of the proposed algorithm, simulations, based on SIFT and DSIFT robots have been performed for two robots. In the proposed algorithm, the master robot calculates the slave's pose based on the vision information that transmits the mapping information to slave robot with which it estimates the localization task. The simulation results show higher performance for the DSIFT because of the reduced distorted of the information for the input images. This is confirmed in that the master robot can serve lower errors in the other robots' pose and gives a more accurate heading angle to the slave robot, even though only two simulations have been carried out. Comparison for an EKF-SLAM, SIFT, and DSIFT based cases are underway in real indoor environments.

Acknowledgment. This research was carried out under the General R/D Program of the Daegu Gyeongbuk Institute of Science and Technology (DGIST), funded by the Ministry of Education, Science and Technology(MEST) of the Republic of Korea.

References

1. Se, S., Lowe, D., Little, J.: Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks. The International Journal of Robotics Research 21(8), 735–759 (2002)
2. Newman, P., Sibley, G., Smith, M., Cummins, M., Harrison, A., Mei, C., Posner, I., Shade, R., Schroeter, D., Murphy, L., Churchill, W., Cole, D., Reid, I.: Navigating, Recognizing and Describing Urban Spaces With Vision and Lasers. The International Journal of Robotics Research 28(11-12), 1406–1433 (2009)
3. Joo, J.K., Lee, H.C.: Object Recognition using SIFT and Tree Structure. Korean Society of Computer Information 16(1), 33–38 (2008)
4. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Van Gool, L.: A comparison of Affine Region Detectors. International Journal of Computer Vision 65(1), 43–72 (2005)
5. Ahn, S.H., Choi, J.W., Doh, N.L., Chung, W.G.: A practical approach for EKF-SLAM in an indoor environment fusing ultrasonic sensors and stereo camera. Auto. Robot. 24, 315–335 (2008)
6. Gwak, M.G., Sung, S.K., Yun, S.C., Won, D.H., Lee, Y.J.: Integrated SIFT Algorithm with Feature Point Matching Filter for Relative Position Estimation. Journal of the KSAS 37(8), 759–766 (2009)
7. Gao, Q., Li, J., Yang, G.: Vision Based Road Crossing Scene Recognition for Robot Localization. In: 2008 International Conference on Computer Science and Software Engineering, pp. 62–68 (2008)
8. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60, 91–110 (2004)
9. Lowe, D.G.: Object recognition from local scale-invariant features. In: International Conference on Computer Vision, pp. 1150–1157 (1999)
10. Wang, Y.Q., Xia, G.H., Zhu, Q.D., Zhao, G.L.: Monte Carlo Localization of Mobile Robot with Modified SIFT. In: 2009 International Conference on Measuring Technology and Mechatronics Automation, pp. 400–403 (2009)

Study on Improved GPGP-Based Multi-agent Semiconductor Fabrication Line Dynamic Scheduling Method

Xin Ma[1,2] and Ying He[3]

[1] College of Computer Science and Technology, Jilin University, Changchun, China
[2] Key Laboratory of Symbol Computation and Knowledge Engineering of the Ministry of Education, Changchun, China
[3] College of Information Economy, ChangChun Taxation College, Changchun, China
{chairmanccit,floral_bb}@hotmail.com

Abstract. A semiconductor fabrication line dynamic scheduling(SFLDS) model combining MAS(Multi-Agent System) with multi-intelligence algorithms is presented in this paper. The proposed model is based on the improved generalized partial global planning(GPGP) and utilizes the advantages of static intelligence algorithms with dynamic MAS. A scheduling process from 'macro-scheduling to micro-scheduling to repeated- scheduling' is designed for large-scale complex problems to enable to implement an effective and widely applicable prototype system for SFLDS. Under this scheme, a set of limitation and improvement of GPGP about its structure are proposed. The improved GPGP and its model are simulated by using simulation software eM—plant. A case study is provided to examine the practicability, flexibility and robustness of the proposed scheduling approach.

Keywords: semiconductor fabrication line dynamic scheduling problem, multi-intelligence algorithm, multi-agent system, improved generalized partial global planning.

1 Introduction

Today semiconductor wafer fabrication is one of the most complex manufacturing processes, with uncertainty, can be re-inflow, mix processing, equipment out of load balance, etc. Which is significantly different from other industries, is a typical discrete event dynamical system(DEDS)[1][2][3]. Because the scheduling problem of semiconductor wafer fabrication systems has been proved to be NP complete problems, and also has the typical multi-constraint, multi-objective, uncertainties and other features, such problems using the traditional methods are often difficult to work. Therefore, a tool designed for solving the practical problems for the scheduling, control and performance analysis that will not only change the inaccurate status quo of current manual scheduling and control, but also have a great significance to achieve computer-controlled production process.

A multi-agent system (MAS) is a system composed of multiple interacting intelligent agents. Multi-agent systems can be used to solve problems which are difficult or impossible for an individual agent or monolithic system to solve. Some researchers have applied MAS to the SWF problem and obtained some beneficial results[4].

Partial global planning (PGP) is a flexible approach to distributed coordination that allows agents to respond dynamically to their current situation[5]. In 1995, K.Decker and V.Lesser of UMASS made GPGP[6][7] cooperation mechanism in 1995 for cooperating many agents. In order to facilitate model, the two learners also proposed a task analysis, environment modeling and simulation (TAEMS[7][8]) that based on a domain independent scheduling model. GPGP mainly composed of five coordination mechanisms: exchanges private views of task, transmits communication, avoids redundant, handles hard coordination,and handles soft coordination. So the GPGP approach consists of an extendable set of modular coordination mechanisms.

In the semiconductor fabrication line dynamic scheduling, the schedulings of single piece equipment and batch processing equipment are existen jointly and couplingly. As domain-independent modeling, TAEMS had been unable to give a valid description of semiconductor wafer fabrication scheduling, so Zhai Wenbin[9] et al proposed ETAEMS(extended TAEMS) to meet the requirements of multi-agent quantitative description. Meanwhile, Zhai has also developed four relationships on GPGP corresponding ETAEMS: Delay-enable, Co-batched, Downstream WIP on the lot scheduling to promote, and Upstream WIP on the lot scheduling to promote relations.

Focused on the complicated characters of emiconductor wafer fabrication, a MAS and improved GPGP based SFLDS is developed. in this paper. Muti-intelligent computing methods are also applied in the system. To validate the effectiveness of our proposed method, it is simulated by using the eM-plant software, and applied to a real problem. Simulation results show that our method is effective and better than the existing methods.

2 Limitations and Improvement of GPGP

Although Zhai et al developed the ability of the Agents and optimized the scheduling program by imporving the GPGP in the limitations of the application, GPGP also has some limitations in the structure: 1)as for a large-scale Agent composition system, GPGP can't describe all the properties of each Agent in a given team task; 2) TAEMS doesn't provide an explicit way to express objectives of tasks performed. The lack of capability to express objectives of all tasks might result that agent trying to complete their part in an already achieved task; 3)GPGP mechanisms neither support synchronization of agent nor handle redundant teamwork. As the limitations mentioned above, we proposed the improved GPGP coordination mechanism, its double expansion mainly reflected in the extension of TAEMS and GPGP mechanism itself.

2.1 TAEMS Extensions

The proposed extensions to TAEMS are two: modeling organizational information and defining monitoring conditions.

Modeling organizational information. We suggest a new TAEMS object—Agent class $Class_{agent}$. $Class_{agent}$ is any TAEMS object defined over a whole team not only a single agent, namely: $Class_{agent} =< \alpha, \beta, ..., \chi >, \alpha = \alpha_1 \cap \alpha_2 \cap ... \cap \alpha_n, \beta = \beta_1 \cap \beta_2 \cap ... \cap \beta_n, \chi = \chi_1 \cap \chi_2 \cap ... \cap \chi_n, Agent_i =< \alpha_i, \beta_i, \chi_i >$, $i \in n \subseteq N$. In general, this extension drops the number of objects required to $Class_{agent}$ define Which provide possibility for Bid Proposal of task structure's description.

Defining completeness and Un-completeness. In standard TAEMS, there is only one measurement applied to all tasks, namely: $task\_quality$. In a given system, the lack of certainty degree for tasks completed and not completed, then reusability of collaborative mechanisms of the system itself will drop substantially. Therefore, we defined a variable for monitoring the state: $Unachievable(T) = f(q(T), c(T), dl(T))$, when $Unachievable(T) > \alpha$ task T can't complete the job.

2.2 GPGP Mechanisms Extensions

Except for 5 GPGP coordination mechanisms mentioned above, imporved GPGP also has three other mechanisms as follows:

Mechanism1: Synchronization mechanism.
<*coordination methods*>: first, differentiate and confirm the role of commitment $leader$ and $common$; second, Timing test $leader$ and replacing invalid $leader$ .i.e.: if $r(T_1, T_2) \in$ PCR and $T_1 \in$ P and Timeout>αthen then replace $leader$ by choosing functioning agent A to the set of agents . $\{a \mid B_A(B_a(T_2))\}$.

Mechanism2: Monitoring Framework mechanism.
<*coordination methods*>: first, timing test $Unachievable(T)$;second, if $Unachievable(T) > \alpha$, then terminate T ; third, executing other assignments.

Mechanism3: "Communicating results "mechanism.
<*coordination methods*>: first, differentiate and confirm the role of commitment $leader$ and $common$; second, pass the result to the other Agents who satisfy the commitments, namely: $\{A \in \mathrm{A}$ and $A = leader \mid B(B_A(C(T))\}$.

3 The MAS-SFLDS Based on Improved GPGP

The MAS-SFLDS based on improved GPGP is mainly consisted of management Agent, Executive Agent, task Agent and resource Agent, all the agents can communicate with each other, as shown in Figure 1:

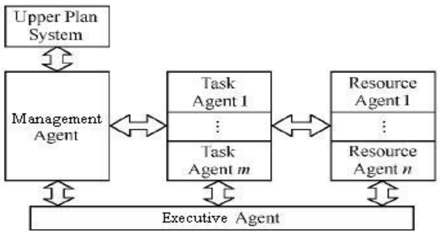


Fig. 1. Improved GPGP-based scheduling model of MAS

Management Agent. Management Agent is non-entities Agent, which is responsible for generating task Agent according to the arrangement from upper-layer; terminates the task agent when the piece is completed, supervises system information of all the Agents. Upper arrangement used to control the semiconductor manufacturing line, cast material, and get the current situation of task agent charged by management agent, then control the cast material time.

Executive Agent. As an intermediary between task side and service side, the objective of the Executive Agent is: complete the task Agent's working efficient in use of resources Agent. Executive Agent maintains a regedit to be scheduling and a regedit of available resources. After received the scheduling instructions, it investigates the collection of all tasks to make sure that they have been decomposed at first, and then follows a certain rule to sort the decomposed working procedure, and updates the procedure-registry and resource-registry, finally, reason out the allocation plan to achieve the "macro-scheduling". The module in the Executive Agent achieveing the scheduling is inference engine by using of the embedded muti-intelligence algorithm.

Task Agent. Agent task with the informations of work piece lot. Task Agent will modify their own properties in according to the informations that from management Agent and resource Agent, and then break down the tasks into sub-tasks or processes through task decomposition modules. At the same time, in order to meet the requirements of improved GPGP, the task Agent that under the task environment[8] as well as the actual situation will construct a global view of the task in use of ETAEMS.

Resource agent. A resource Agent corresponds to an equipment. Rresource Agent will construct a local task view in according to the structural informations; communicate with the relevant agents, access to the relationships with the other Agents and establish a part-global task view. And then apply the improved GPGP again to accomplish the commitments between tasks and form a local virtual task. Then, resource Agent will schedule their own methods and tasks in use of scheduling module. After the implementation of the scheduling program, resource Agent sent the completed informations and results to task Agent to achieve the "micro-scheduling" process. For the failure of the methods or tasks, resource Agent will select the replaced plan in according to a part-global task view, or will send the failure messages to the executive Agent and task Agent by the executive Agent for "repeated-scheduling", until the system completed the task.

4 System Scheduling

4.1 Macro Scheduling—Intelligence Inference Engine

The main function of executive Agent is to achieve the "macro scheduling". With the muti-objective optimization of the inference engine in the executive Agent, the system will choose the appreciated intelligence algorithm of the semiconductor fabrication produce scheduling from table 1 as shown below in according to the characteristic and occasion of a intelligence algorithm.

Table 1. List of characteristics and applicable situation for optimization methods

Algorithmic name	Operation speed	Optimized efficiency	Scheduling size	Appropriate situation
Integer programming	Slow	Perfect	Small	Small
NN	Middle	Middle	Large/middle	Large
GA	Middle	Middle	Middle	Universal
local search and GA	Speedier	Middle	Large/middle	Large
SAGA	Speedier	Perfect	Large/middle	Large

4.2 Micro-scheduling—Improved GPGP Coordination Mechanism

In this system, the “micro-scheduling” mainly happens in the resource Agent. The specific “micro-scheduling” are as follows:

StepA1. View from the global task, read the local task view;
StepA2. Exchange relevant views and form local-global task view;
StepA3. Detect coordination relationships;
StepA4. To determine whether CR is empty. If it is not, goto StepA1;
StepA5. Apply improved GPGP, make commitment and accept a virtual task;
StepA6. Use the scheduling module of the resource Agent for local task scheduling;
StepA7. Resource equipment executive the scheduling, goto StepB4.

In StepA3, improved GPGP is divided into two categories: one is the single piece of resources Agent coordination mechanism, it gives priority to single piece equipment in order to maintain uniform distribution; the other is a batch processing of resource Agent coordination mechanism. It should be combined with the same technical upstream wafer in an early period of the batch processing.

4.3 The Overall Scheduling Steps of System

As mentioned before, the system's scheduling algorithm is as follows:

StepB1. Update the scheduling procedures registry information and available resources registry information of executive Agent;
StepB2. Task Agent decomposition process, and construct the global task of view;

StepB3. Executive Agent allocated procedures to specific resource device in using of the inference engine, to achieve "macro-scheduling";

StepB4. Executive improved GPGP coordination control in resource Agent and achieve "micro-scheduling", goto StepA1;

StepB5. Inform the Executive Agent update task sets, goto StepB1.

4.4 Repeated-rescheduling

Rescheduling policy is used to determine what caused rescheduling; its key judgments are two: The first is the existence of critical equipment failures; the second is the existence of match point. The specific methods are as follows:

StepC1. To determine whether there is a critical equipment failure, or else, goto StepC3;

StepC2. Resource Agent reporting to management Agent, management Agent sets resource equipment abnormal;

StepC3. if there is a match point or not, if it is, goto Step C4;

StepC4. oto Step B1.

5 System Simulation and Results Comparison

The paper used eM-plant simulation software, adopted simulation model that based on the actual production line, set up the corresponding MAS model, and achieved the Agent design and improved GPGP. The actual production data of a semiconductor wafer fabrication used for testing. The semiconductor wafer fabrication is composed of 10 bottleneck equipments, there are 28 sets of key equipment in all. Meamwhile, we apply GASA algorithm(genetic algorithms-simulated annealing) to the inference engine. Tested data including three kinds of wafer products a,b,c, a total of 79 construction sections. Before the "macro-scheduling", the work procedure sorted by the following rules: single piece of equipment FCFS, batch processing equipment MBS. Three kinds of wafer products' rate of charge is 1:1:1 per day, each product is expected to batch charging according to uniform manner, the simulation system is running 650 days.

In order to investigate the stability of wafer etch processing, we have collected parts quantity (as shown in Figure 2,3,4)of wafer etching subsystem model in 134 days-138 days, 384 days-389 days and 604 days-608 days respectively, simulation results basically stable at between 20-30, and cyclical changes. Therefore, tests show that the system not only has a lower redundancy but also has a more stable working performance.

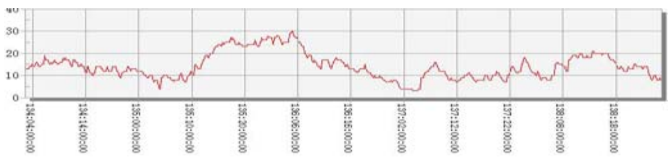


Fig. 2. Parts quantity of wafer etching subsystem model in 134 days- 138 days

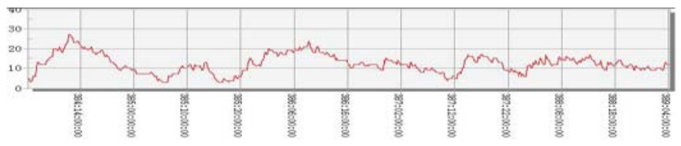


Fig. 3. Parts quantity of wafer etching subsystem model in 384 days- 389 days

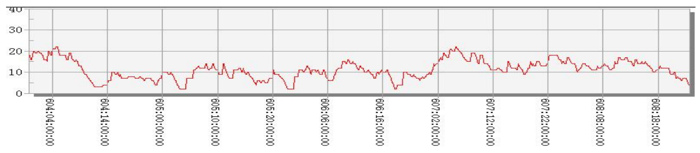


Fig. 4. Parts quantity of wafer etching subsystem model in 604 days- 608 days

At last, to test the performance of the whole SFLDS, the output of the system is compared with those of GPGP-CN[9] and FCFS-MBS[10] methods. Table 2 show the comparison results. From Table 2, it is easy to draw that the output of SFLDS is about 19% more than that of GPGP-CN, and 44.6% more than that of FCFS-MBS. So it can be concluded that the performance of our proposed method is superior to the existing two methods.

Table 2. The output numbers of different methods

Day	100	200	300	400	500	600	ave
SFLDS	50	58	52	49	60	65	55.7
GPGP-CN	50	48	45	44	46	48	46.8
FCFS-MBS	50	40	35	31	36	39	38.5

6 Conclusion

Focused on the scheduling problem for SFLDS, this paper develops an efficient system based on multi-agent system and an improved GPGP, genetic algorithm and simulated annealing methods are also applied in the system. Numerical simulation is performed to test the effectiveness of the proposed method. The results show that our proposed method is effective and robust. The comparisons with other existing results indicate that the performance of improved GPGP is better than those of others.

References

1. Kim, Y.D., Shim, S.O., Choi, B., et al.: Simplification methods for accelerating simulation-based real-time scheduling in a semiconductor wafer fabrication facility. J. IEEE Transactions on Semiconductor Manufacturing 290 (2003)

2. Hsieh, B.W., Chen, C.H., Chang, S.C.: Scheduling semiconductor wafer fabrication by using ordinal optimization-based simulation. J. IEEE Transactions on Robotics and Automation 599 (2001)
3. Hwang, T.K., Chang, S.C.: Design of a lagrangian relaxation-based hierarchical production scheduling environment for semiconductor wafer fabrication. J. IEEE Transactions on Robotics and Automation 566 (2003)
4. Wu, J.W., Xiao, Y.S.: Multi2agent technology in scheduling of semiconductor production line. Computer Integrated Manufacturing Systems 9(8), 641–644 (2003)
5. Durfee, E.H., Lesser, V.R.: Partial global planning: A coordination framework for distributed hypothesis formation. IEEE Transactions on Systems, Man, and Cybernetics 21(5), 1167–1183 (1991)
6. Decker, K., Lesser, V.R.: Generalizing the partial global planning algorithm. J. Intelligent and Cooperative Information Systems, 319–346 (1992)
7. Decker, K., Lesser, V.R.: Designing a family of coordination algorithms. In: Proceedings of the First International Conference on Multi-Agent Systems, pp. 73–80. AAAI Press, San Francisco (1995)
8. Decker, K., Lesser, V.R., Carver, N., et al.: Evolution of the GPGP/TAEMS domain - independent coordination framework. J. Autonomous Agents and Multi-Agent Systems, 87–143 (2004)
9. Zhai, W.B., Zhang, J., Yan, J.Q., Ma, D.Z.: Research on ETAEMS/GPGP-CN based Dynamic Scheduling Technology of Semiconductor Fabrication Line. J. Mechanical Engineering, 53–58 (2005)
10. Gurnani, H., Anupindi, R., Akella, R.: Control of batch processing systems in semiconductor wafer fabrication facilities. IEEE Trans. Semiconduct. Manufact. 5, 319–328 (1992)

Multi-robot Formation Control Using Reinforcement Learning Method

Guoyu Zuo, Jiatong Han, and Guansheng Han

School of Electronic Information & Control Engineering,
Beijing University of Technology, Beijing 100124, China
hjt925@163.com

Abstract. Formation is a good example of the research for multi-robot cooperation. Many different ways can be used to accomplish this task, but the main drawbacks of most of these methods are that robots can't self-learn. In Brooks' behavioral opinion, this paper is to verify that the reinforcement learning method can be used for robots to select different behaviors in various different situations. Experiments are performed to illustrate the team robots' capability of self-learning and autonomy. The results show that the robots can get a self-formation in a barrier environment after learning.

Keywords: Multi-robot, Formation control, Reinforcement Learning.

1 Introduction

There are different levels of needs in maintaining the robot formation's completeness under various mission requirements. To obtain the goal of the robot formation moving to a specified point, motion control should take into accounts the requirements of formation, such as avoiding obstacles and other factors. In this paper we mainly use Brooks' behavioral inhibition method, in which, at every moment, the formation task is specified to be a series of actions, which are here defined as act l, act 2, ..., act n. Using these defined behaviors, multi-robot system can perform the self-formation in an obstacle environment. Each robot's control system has a reinforcement learning module to achieve the upper behavior control. Our aim is that through learning multi-robot system can autonomously construct some good pairs from environment to behavior. This mapping relation can make robots choose appropriate behaviors at each step in the environment to ensure the formation task's achievement without colliding with the obstacles.

2 The Leader-Follower Control Method

The Leader-Follower control method is often used in multi-robot formation control, in which the team leader decides where the formation moves while avoiding obstacles, and the other robots, i.e., the followers, follow the leader with a certain speed and angular velocity [1, 2].

Figure 1 is the $l-\varphi$ control methods. The follower always maintains following its leader, ensuring that $l - l_{12}{}^{d}$ and $\varphi - \varphi_{12}{}^{d}$, where l is the distance between two robots, φ is the angle between them, and d is the distance between the centres of robot's two wheel axes and the robot's rotation.

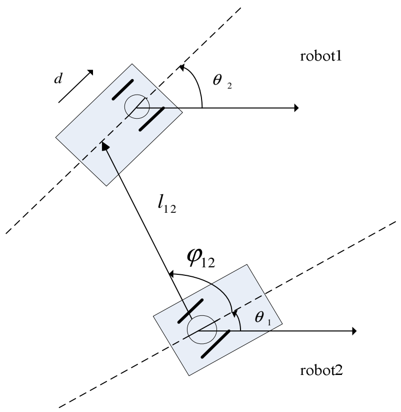


Fig. 1. Model of $l-\varphi$

The kinematics equations of the leader robot are:

$$\begin{aligned} \dot{x} &= v_i \cos\theta_i \\ \dot{y} &= v_i \sin\theta_i \\ \dot{\theta}_i &= \omega_i \end{aligned} \tag{1}$$

here v_i and ω_i, i=(1,2) represent the line speed and angular velocity of the two robots, respectively. The kinematics equations of the follower robot are shown as follows:

$$\begin{aligned} \dot{l}_{12} &= v_2 \cos\gamma_1 - v_1 \cos\varphi_{12} + d\omega_2 \sin\gamma_1 \\ \dot{\varphi}_{12} &= \frac{1}{l_{12}}\{v_1 \sin\varphi_{12} - v_2 \sin\gamma_1 + d\omega_2 \cos\gamma_1 - l_{12}\omega_1\} \\ \dot{\theta}_2 &= \omega_2 \\ \gamma_1 &= \theta_1 + \varphi_{12} - \theta_2 \end{aligned} \tag{2}$$

For the follower, the control output (ω_2, v_2) is:

$$\begin{aligned} \omega_2 &= \frac{\cos\gamma_1}{d}\{a_2 l_{12}(\varphi_{12}{}^{d} - \varphi_{12}) - v_1 \sin\varphi_{12} + l_{12}\omega_1 + \rho_{12}\sin\gamma_1\} \\ v_2 &= \rho_{12} - d\omega_2 \tan\gamma_1 \end{aligned} \tag{3}$$

here,

$$\rho_{12} = \frac{a_1(l_{12}^d - l_{12})}{\cos\gamma_1} + \frac{v_1\cos\varphi_{12}}{\cos\gamma_1} \tag{4}$$

So the $l-\varphi$ closed-loop control is expressed as:

$$\begin{aligned} \dot{l}_{12} &= a_1(\dot{l}_{12}^d - l_{12}) \\ \dot{\varphi}_{12} &= a_2(\varphi_{12}^d - \varphi_{12}) \end{aligned} \tag{5}$$

where a_1 and a_2 are the coefficients of the proportional control method.

3 Behavioral Design of Multi-robot Formation Using Reinforcement Learning

If there are obstacles in the environment, the robot can not pass through the barriers while maintaining the original formation. In general, they need to change their formation in different types. When passing through a narrow obstacle, the formation usually converts into a line formation. That is, the followers need to change the following angle with the leader in order to pass through the obstacles. After the followers move through the obstacles and come to a spacious environment, they change the angle again to return to the original formation. Here we use a reinforcement learning method to design the behaviors of the followers.

Q learning is a reinforcement learning algorithm. The main idea of it is not to learn each state of the evaluation function, but to learn each state-action pairs' evaluation function *Q(s, a)*. *Q(s, a)* signifies the state's cumulative discounted value after performing actions. When the unknown or dynamic enviroment changes, *Q* learning has a great advantage. It does not need to know the next state after actions, but need to rely on the current state-action pairs's *Q* values to determine the optimal strategy in state s [3, 4].

$$Q(s_t, a_t) = r_t + \gamma \max_{a\in A} Q(s_{t+1}, a) \tag{6}$$

Equation (6) is set up only under the conditions of the optimal strategy. In the reinforcement learning process, the two ends of equation (6) is not strictly equal, and the error is expressed as:

$$\Delta Q_t(s_t, a_t) = r_t + \gamma \max_{a\in A} Q_{t-1}(s_{t+1}, a) - Q_{t-1}(s_t, a_t) \tag{7}$$

$$\begin{aligned} Q_t(s_t, a_t) &= Q_{t-1}(s_t, a_t) + \alpha_t \Delta Q_t(s_t, a_t) \\ &= Q_{t-1}(s_t, a_t) + \alpha_t[r_t + \gamma \max_{a\in A} Q_{t-1}(s_{t+1}, a) - Q_{t-1}(s_t, a_t)] \\ &= (1-\alpha_t)Q_{t-1}(s_t, a_t) + \alpha_t[r_t + \gamma \max_{a\in A} Q_{t-1}(s_{t+1}, a)] \\ &= (1-\alpha_t)Q_{t-1}(s_t, a_t) + \alpha_t[r_t + \gamma V_t(s_{t+1})] \end{aligned} \tag{8}$$

where, $V_t(s_{t+1}) = \max_{a\in A} Q_{t-1}(s_{t+1}, a)$

α_t: learning rate in the moment of *t*,

γ: discount factor for $V(s_{t+1})$,

r_t: the reinforcement signal at the moment of *t* to imply the action a_t.

The interactions between the robots and environment are as follows:

1) The robot perceives the current environment state s_t,
2) Obtain a timely reward for the current state of the environment, and perform the appropriate action according to a certain strategic selected,
3) Perform the selected actions, and the environment changes,
4) Get the next state of the environment s_{t+1},
5) Calculate the reward r_t timely,
6) $t \leftarrow t+1$, move to step 7 if the learning objectives have been achieved, otherwise, turn step 2,
7) The end of the learning process.

The reinforcement learning process is as follows: each robot gets the information of environment through the by its own sonar information coming from the other robots. This type of sensor information is sent to the reinforcement learning module (Q-learning). The reinforcement learning module decides to select which action to behave in accordance with all of the robot's sensor signal: act 1, act 2, ..., act n, and the robots take the appropriate actions in the environment. The robot's environmental system will give every action different enhanced signal values based on the role of behavior. The system will decide the trend of such actions, i. e., whether they will be strengthened or weakened in such an environment. And eventually the system will learn the different circumstances to take appropriate actions to achieve a self-formation in the obstacles space [5, 6].

Reinforcement learning is based on the current state and acts a behavior with a random value. For the action selection, we use the Boltzmann distribution to realize choice of random acts.

4 Experiment and Analysis

In the following experiments, two Pioneer3-DX mobile robots are used as the experimental platform to research multi-robot formation. The robot is equipped with 16 sonar sensors, which can cover the 0 ~ 360 ° range around.

4.1 Robot's State Space

The state space of robot is expressed as:

$$s = \{d_l, d_f, d_r\}$$

d_l: The distance between the left side of the robot and the obstacle,

d_f: The distance between the front side of the robot and the obstacle,

d_r: The distance between the right side of the robot and the obstacle.

The above three parameters of state space are shown in Figure 2.

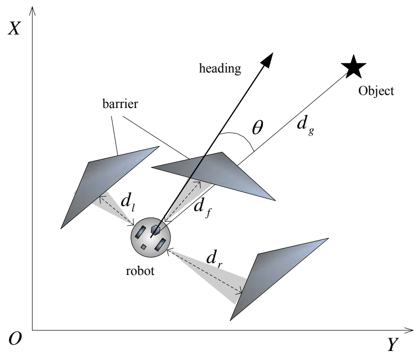


Fig. 2. Position relationships between the robot, the obstacle and the goal

As the maximum distance detected by the sonar sensors is 5000mm, the minimum effective distance is 100mm, so $100 < d_f, d_r, d_l < 5000$mm, that is, we will not consider the distance to obstacles of more than 5000mm or less than 100mm. using following formula to calculate the distance-weighted average.

$$range = 0.5d_f + 0.25d_r + 0.25d_l \quad (9)$$

The distances between the robots and the obstacles are divided into three discrete states as shown in Table 1:

Table 1. Division of robot states (mm)

State	Small	Middle	Large
range	*0<range<500*	*500<range<2000*	*range>2000*

Table 2. Q- Table of states-action pairs

	Small(s_1)	Middle(s_2)	Large(s_3)
Action1(a_1)	$Q(s_1, a_1)$	$Q(s_2, a_1)$	$Q(s_3, a_1)$
Action2(a_2)	$Q(s_1, a_2)$	$Q(s_2, a_2)$	$Q(s_3, a_2)$

At the same time, we define two kinds of robot behaviors:

Action1: keeping the original formation (keep the angle between leader and follower to be $\theta = 0^o$);

Action2: transforming the original formation into a line formation (adjust the angle between leader and follower to be $\theta = 180^o$).

The state -action pairs are shown in Table 2:

4.2 Reinforcement Signals

The reinforcement signal selection is a very important for reinforcement learning method. Here, we use both internal and external reinforcement signals to reflect the individual's interests and the interests of the whole.

1) Internal reinforcement signal: The internal reinforcement signal is used to evaluate the individual interests of the robots, which is defined by the distance between the robot and obstacles. l_{min} is the Minimum distance. When $l < l_{min}$, we think that the robot gets into a dangerous place, so give it a punishment. l_{max} is considered as a safe distance. When $l < l_{max}$, the robot is relatively safe, we give it awards.

$$r_{in} = \begin{cases} -1 & l < l_{min} \\ 1 & l > l_{max} \\ f(l) & l_{min} < l < l_{max} \end{cases} \tag{10}$$

where $f(l)$ is a linear function defined as:

$$f(l) = \frac{2}{(l_{\max} - l_{\min})} \cdot (l - l_{\min}) - 1 \tag{11}$$

2) External reinforcement signal: The signal is used to regulate the group action for the overall interests. The group actions planned by each robot in each step of reinforcement learning may not be the inconsistent. In order to make each follower robot keep the same behavior as the leader, an election approach which uses the expected behavior of most of the robots as a whole action, and then each robot implements the overall team behavior. Here we define the external reinforcement signal for each robot as follows. If the robot behavior is consistent with the team behavior, the robot will be rewarded for this behavior; otherwise, it will be punished. The external reinforcement signal is denoted as:

$$r_{out} = \begin{cases} 1 & \text{the robot's behavior is consistent with the team's ultimately behavior} \\ -1 & \text{otherwise} \end{cases} \tag{12}$$

The overall reinforcement signal is expressed as the weighted sum of internal and external reinforcement signal:

$$r = \alpha \cdot r_{in} + \beta \cdot r_{out} \tag{13}$$

where $\alpha + \beta = 1$.

Q function is defined as:

$$Q(s_t, a_t) = r_t + \gamma \max_{a \in A} Q(s_{t+1}, a) \tag{14}$$

The update rules are:

$$\Delta Q_t(s_t, a_t) = r_t + \gamma \max_{a \in A} Q_{t-1}(s_{t+1}, a) - Q_{t-1}(s_t, a_t) \tag{15}$$

$$Q_t(s_t, a_t) = Q_{t-1}(s_t, a_t) + \alpha_t \Delta Q_t(s_t, a_t) \tag{16}$$

At the beginning of the reinforcement learning, the main task of learning is to explore the environment, thus the randomness of action selection should be greater, in the latter stage of reinforcement learning, learning should converge, so the randomness of action selection should be smaller. Boltzmann machine is used here to carry out anneal operation. The probability of action a_i was chosen as follows:

$$P(a_i) = \frac{e^{Q(s,a_i)/T}}{\sum_{a_n \in A} e^{Q(s,a_n)/T}} \qquad T = T_0 t^{-1/\beta} \tag{17}$$

T_0: the initial temperature value,

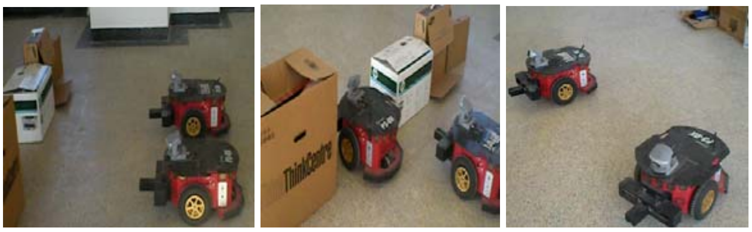
t: Time,

T: the current temperature value, obtained from T_0's decay with time,

β: Constant, used to control the rate of annealing.

The ε-greedy strategy is used to select action a_i. In each action selection, use $p(a_i) = \varepsilon$ to randomly select actions, and use 1-ε to select the action which has the final largest Q value [7].

Figure 3 shows the formation walking conditions after learning.



(a) (b) (c)

Fig. 3. The robots (a) maintain in a formation in spacious environment, (b) get into line formation when encountering obstacles, and (c) get back to the original formation after moving through the obstacles

In the experiment, the leader plans out a path from the beginning to the end. The follower changes the angular with the leader in accordance with the changes of environment and is gradually adapted to a new environment. The robots maintain the original formation in a spacious environment. As they go closer to the obstacle, they get into linear formation to pass through the obstacles. When the environment becomes more spacious again, the robots gradually adjust to form the formation like the original in the first stage.

Acknowledgments. This work is supported by the National Natural Science Foundation of China (No. 60873043) and the Funding Program for Academic Human Resources Development in Institutions of Higher Learning under the Jurisdiction of Beijing Municipality of China (No. PHR201008006).

References

1. Shao, J.Y., Xie, G.M., Yu, J.Z., Wang, L.: Leader-following Formation Control of Multiple Mobile Robots. In: Proeeedings of the 2005 IEEE International Symposium on Intelligent Control Limassol, June 27-29, pp. 808–813 (2005)
2. Jaydev, J., Desai, P., Ostrowski, J.P., Kumar, V.: Modeling and Control of Formations of Nonholonomic Mobile Robots. IEEE transactions on robotics and automation 17(6), 905–908 (2001)
3. Ruan, X.G., Cai, J.X., Chen, J.: Learnning to Control Two-Wheeled Self-Balanced Robot Using Reinforcement Learning Rules and Fuzzy Neural Network. In: Fourth International Conference on Natural Computation, pp. 395–398 (2008)
4. Ishikawa, M., Hagiwara, T., Yamamoto, N., Kiriake, F.: Brain-Inspired Emergence of Behaviors in Mobilr Robots by Reinforcement Learning with Internal Rewards. In: Eigth International Conference on Hybrid Intelligent System, pp. 138–143 (2008)
5. Dung, L.T., Kokeda, T., Takagi, M.: Reinforcement Learning in Non-Markovian Environments using Sutomatic Discovery of Subgoals. In: SICE Annual Conference, pp. 2600–2605 (2007)
6. Huang, B.Q.: Reinforcement learning method and applied research. PhD thesis, Shanghai Jiaotong University, 22-23 (2007)
7. Gao, Y., Chen, S.F., Lu, X.: Research on reinforcement learning. Automatica Sinica 30(1), 86–100 (2004)

Development of Image Stabilization System Using Extended Kalman Filter for a Mobile Robot

Yun Won Choi, Tae Hun Kang, and Suk Gyu Lee

Electrical Building 208, Yeungnam Univ, Dae-dong, Kyungsan-si, Kyungsangbuk-do, Korea
empire6@lycos.co.kr, bigxihn@postech.ac.kr, sglee@ynu.ac.kr

Abstract. This paper proposes a robust image stabilization system for a mobile robot using Extended Kalman Filter (EKF). Though image information is one of the most efficient data for robot navigation, it is subject to noise which results from internal vibration as well as external factors such as uneven terrain, stairs, or marshy surface. The vibration of camera deteriorates the definition of image by destroying image sharpness, which seriously prevents mobile robots from recognizing their environment for navigation. In this paper, inclinometer was used to measure the vibration angle of the camera system mounted on the robot to obtain a reliable image by compensating for the angle of the camera shake caused by vibration. In addition angle prediction by using the EKF enhances responsibility of image analysis for real time performance. The Experimental results show effectiveness of the proposed system to compensate for the blurring of the images.

Keywords: image stabilization, mobile robot, Inclinometer, Extended Kalman Filter.

1 Introduction

Image stabilization is the process which removes the unwanted fluctuations of image sequences captured from the cameras improving, therefore, its visual quality. Image information is one of the most important qualitative features for mobile robot navigation. In order for the mobile robot to recognize its surrounding environment for navigation, acquisition of reliable image from a camera mounted on the robot is indispensable. There is much research on image stabilization for military and tele-operation utilities, as well as application in commercial palmcorders. The fundamental methods for stabilizing image include several different approaches. In [1], they proposed a new digital video stabilization approach based on a 2.5-D motion model with inertial filtering using springs and dampers. In [2] and [3], image stability by actuating the camera was proposed and implemented. [4] and [5] proposed image stabilization approaches using image compensation for the deteriorated image. The stabilization technique using springs and dampers can be used only for small range of camera vibration and needs heavy calculation burden which results in time delay of the entire processing. Image stabilization by actuating the camera is one of the basic methods for image stabilization, since it is compact and consumes low energy. [6]

We propose an image stabilization system combines two image stabilization methods. One is stabilizing method by rotating mechanically the camera from the motion of the robot. The other stabilizes the image by predicting angle using EKF. The proposed system is basically a 1-DOF camera system embodied in the robot mechanism, where the vibration is measured by using inclinometer. For image stabilization, EKF predicts the vibration using the vibration data. As shown in both simulation and experimental results, the proposed system using EKF provides more stable image comparing with conventional approaches.

2 Review of Image Stabilization Systems

2.1 Optical Image Stabilization (OIS)

OIS can be implemented in both still image and motion image. This system sometimes referred to as optoelectronic image stabilization because it uses the optical path to compensate for vibration. A moveable lens shifts the optical path in order to avoid blurring. The correction element's motion is perpendicular to the optical axis in opposite direction to the handshake. The OIS detects camera shake using two angular velocity sensors for x-axis (pitch) and one for y-axis (yaw). This system is one of most common image stabilization.

2.2 Electromechanical Image Stabilization (EMIS)

The specific feature of EMIS moves directly image sensor to compensate for handshake. Unlike optical stabilizers, it is possible to use any lens with the camera body equipped with EMIS. Angular velocity sensors detect camera movement and relay the amount of compensation necessary to the electromagnets that move the sensor to compensate for any shake. EMIS provides a crucial advantage when shooting handheld with telephoto or tele-zoom lenses, at macro distances, or any other situation that magnifies the effects of camera shake.

2.3 Digital Image Stabilization (DIS)

DIS is different from the previous techniques. It is exclusively independent of hardware. DIS is based on electrical technique since information on vibration obtained from sensors mounted on the system is used for digital image processing. The system stabilizes the image using sensor data saved. It has already shaking information, it compensates image by moving image [11].

However, general DIS consists of two parts. One estimates several local motion vectors from other locations using Block Matching Algorithm [12]. The other decides motion vector of the system using the local motion vector obtained in the previous case. The motion estimation plays an important role in digital image stabilization. The global block matching method is considered as one of the best methods because it covers the whole area. However, it needs heavy calculational burden, processing time and complex hardware system.

3 The Enhanced Image Stabilizing System

3.1 Target System

In this paper, we propose an image stabilization system which can be applied to mobile robot system. In the system, roll axis vibration which has vibration range of -90 ~ 90 degree and frequencies less than 5 Hz will be considered since roll axis vibration is dominant one in real situation.

3.2 Mechanical Mechanism

The proposed stabilization system as shown in figure1 consists of a motor, harmonic drive which is connected directly to the motor, inclinometer to measure angle and controller in the behind of the motor. The harmonic drive has a reduction ratio of 50:1 and has speed of 4,810 rpm in unload case and 4,080 in normal operation.

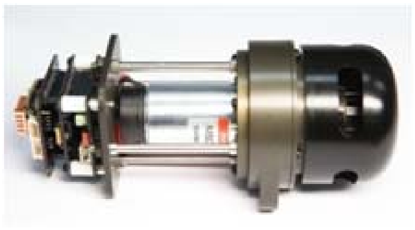


Fig. 1. Prototype of the proposed system

3.3 Control Mechanism

In order to improve the performance of the image stabilization due to vibration, it is desirable to predict the states. A novel and real-time stabilization can be obtained by using Kalman filters to remove short term image fluctuations with retained smooth gross movements. The efficiency of the Kalman filer is then due to the fact that the parameters of the resulting Gaussian can be computed in closed form [13]. Unfortunately, state transitions and measurements are rarely linear in practice. It is well known that EKF based predictor shows good performance in nonlinear system such as vibration which has nonlinear characteristic in this simulation. The image is stabilized by compensating for angular fluctuation due to vibration based on the current and estimated angle.

The vibration can be expressed as trigonometric function y as in eq. (1)

$$y = A\sin\left(\omega_{(k)} nT\right) \tag{1}$$

Where A and $\omega_{(k)}$ are amplitude and angular velocity respectively.

When the measured data are applied to the EKF, the state vector of the system has the form of

$$\vec{x}_{(k)} = [\theta, \omega]^T, \qquad \vec{z}_{(\mathrm{k})} = [\theta, \omega]^T \tag{2}$$

The state equation of the system can be expressed as

$$\vec{x}_{(\mathrm{k+1})} = f\left(\vec{x}_{(k)}\right) + w_{(k)}$$
$$\vec{z}_{(k+1)} = h\left(\vec{x}_{(k+1)}, m_{(k)}\right) + v_{(k)} \tag{3}$$

$$\theta_{(k+1)} = \theta_{(k)} + \omega_{(k)} AT \cos\left(\omega_{(k)} nT\right)$$
$$\omega_{(k+1)} = \omega_{(k)} \tag{4}$$

To linearize the above equations, the following Jacobian matrix is used:

$$F_{(k)} = \begin{bmatrix} 1 & AT\cos(\omega_{(k)} nT) - A\omega_{(k)} nT^2 \sin\left(\omega_{(k)} nT\right) \\ 0 & 1 \end{bmatrix} \tag{5}$$

Where T and n are the sampling time and the step number respectively. The function can be used to compute the predicted state from the previous estimate and similarly the predicted measurement from the predicted state. However, f and h cannot be applied to the covariance directly. To apply to the EKF using already explained system model, first, the predictions of the state and the covariance are computed by:

$$\hat{x}_{(\mathrm{k+1|k})} = f\left(\hat{x}_{(k)}, 0\right)$$
$$P_{(k+1|k)} = F_{(k)} P_{(k|k)} F_{(k)}{}^{T} + Q_{(k)} \tag{6}$$

Once we have the measures of the angle, the innovation and the covariance of the innovation can be computed by eq. (7) and (8)

$$\tilde{A}_{(k+1|k)} = z_{(k+1)} - h(x_{(k+1|k+1)}, 0) \tag{7}$$

$$S_{(k+1)} = H_{(k+1)} P_{(k+1|k)} H_{(k)}{}^{T} + R_{(k+1)} \tag{8}$$

Where H is the Jacobian matrix of observation model. Kalman gain can be computed by

$$K_{(k+1)} = P_{(k+1|k)} H_{(k+1)}{}^{T} S_{(k+1)}{}^{-1} \tag{9}$$

Finally, the estimation of the state and covariance of the system are obtained by:

$$\hat{x}_{(\mathrm{k+1|k+1})} = \hat{x}_{(\mathrm{k+1|k})} + K_{(k+1)} \tilde{A}_{(k+1)} \tag{10}$$

$$P_{(k+1|k+1)} = P_{(k+1|k)} - H_{(k)}{}^{T} + K_{(k+1)} S_{(k+1)} K_{(k+1)}{}^{T} \tag{11}$$

4 Simulation

The efficiency of angle prediction by the proposed method has been verified using the MATLAB simulation. The sample input for estimation has sine wave form with

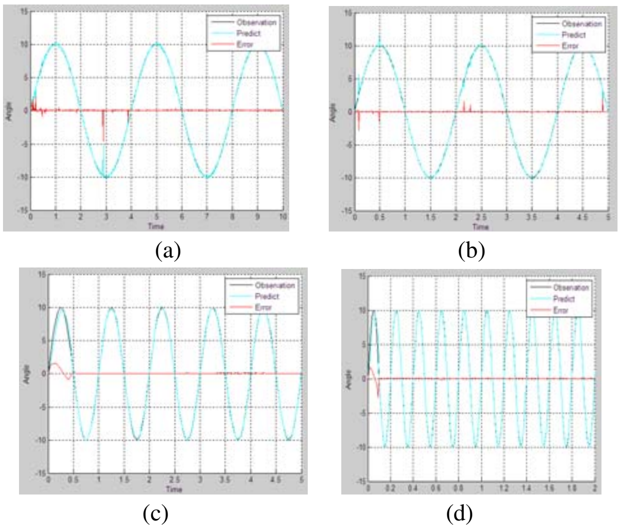


Fig. 2. Simulation result for the each frequency. (a) 0.25Hz, (b) 0.5Hz, (c) 1.0Hz, (d) 5.0Hz

amplitude of 10 degree. According to the simulation results shown in Fig. 2, the estimated signal traces the input signal, faithfully with little error. As shown in the figures, the error is reduced after 2nd period drastically.

Table 1 shows the result of the average of the error. The error increases as the frequency is bigger in the range of 5 % of error. Therefore, the angular estimation is acceptable in real applications.

Table 1. The comparison of simulation results

	0.25Hz	0.5Hz	2.5Hz	5.0Hz
Error	-0.003424	-0.017394	-0.283652	-0.557989

5 Experimental Results

5.1 Experiment Apparatus

In order to evaluate the performance of the proposed system, we built an image capturing system where frequency and amplitude of the applied vibration are adjustable. As shown in Fig. 3(a), (b). The experimental apparatus is in cylindrical shape connected with DC motor. It generates the vibrating signals with -10 ~ 10 degrees of amplitude and five steps of frequency between 0.25 ~ 5 Hz as inputs of the system for evaluation. The proposed system receives the target image shown in Fig. 3 (c) as input through the camera attached on the end of the system. The number of corners of the image is counted by using Harris Corner Detection algorithm.

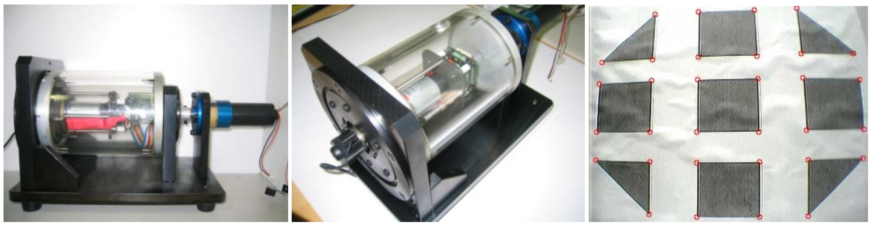
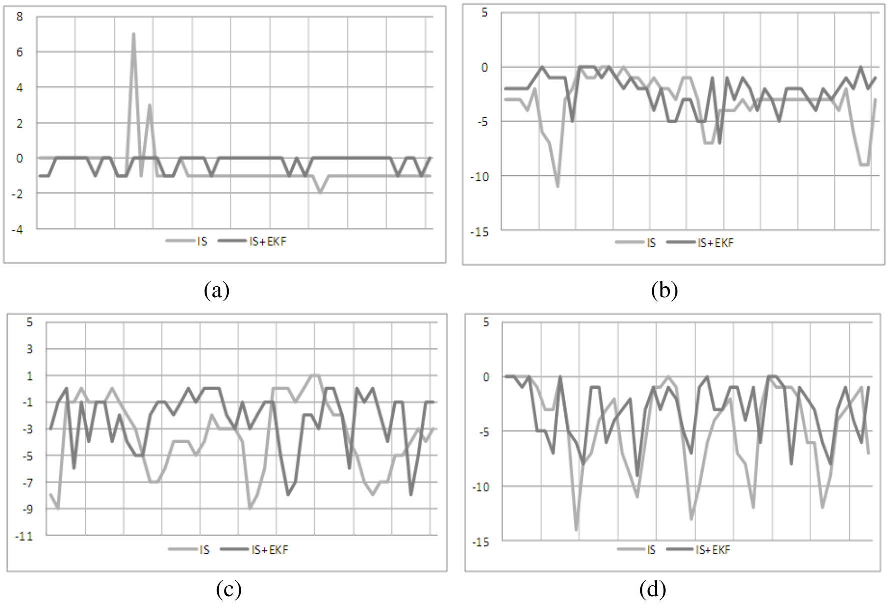
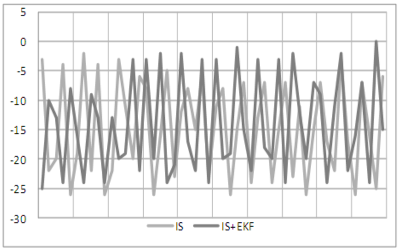

(a) (b) (c)

Fig. 3. Equipment and target image for the experiment

5.2 Experimental Results

The number of corners of the image shown in Fig. 3 (c) is set to 32. The detected number of corners of the image when the system is in vibration denotes the stability





(e)

Fig. 4. Experimental result for the each frequency (a) 0.25Hz, (b) 0.5Hz, (c) 1.0Hz, (d) 2.5Hz, (e) 5.0Hz

index of the captured image. The more closer it is to 32, the more reliable the system is. The following figures show the difference in numbers between the number of detected corners and 32.

Fig 4 (a) shows the experimental result in case of 0.25 Hz of frequency. In case of 0.5 Hz, the lowest frequency in the experiment, the EKF based method shows slightly better stabilization. As shown in the figures below, the proposed system using EKF is especially effective for image stabilization in case of 0.5, 1.0 and 2.5 Hz cases. However, for 5.0 Hz case, EKF based algorithm results in better image stabilization but slightly out of phase.

Table 2 describes the experimental results of corner detection rate in case of IS and IS+EKF for some different frequencies. According to table 2, EKF based approach shows better performance in image stabilization comparing with IS for 5 different frequencies. As a result, EKF is helpful for image stabilization.

Table 2. The comparison of experimental results

	0.1Hz	0.25Hz	0.5Hz	2.5Hz	5.0Hz
IS Only	98.19%	90.00%	89.38%	84.19%	54.38%
IS + EKF	99.31%	93.06%	92.88%	89.06%	58.88%

6 Conclusions

In this paper, we propose a novel image stabilization system based on EKF. The EKF based stabilization enables real-time utilization and provides a successful performance for fluctuation cancellation of camera movements. Differently from the conventional system based on hardware, the proposed system is designed to enhance stability and responsibility by predicting the states using previous states of velocity and angular velocity of the system. The experimental results for image stabilization show good performance of the proposed system for the input vibrations of 0.25, 0.5, 1, 2.5 and 5.0Hz. Special performance enhancement was shown in the first four cases comparing with the fourth case.

There are several important issues that need to be addressed in future research in this area for real world applications including: 1) performance verification of the proposed system in wider range of frequencies and angular, 2) 2-axis image stabilization for mobile navigation.

References

1. Jin, J.S., Member, IEEE., Zhu, Z., Member, IEEE., Xu, G., Senior Member, IEEE.: A Stable Vision System for Moving Vehicles. IEEE transactions on intelligent transportation systems 1(1), 32–39 (2000)
2. Kawano, K., et al.: Development of New Camera Stabilizer ACE-3000, Technology Report, Japan Aviation Electronics Industry, Ltd
3. Sato, K., Ishizuka, S., Nikami, A., Sato, M.: Control techniques for optical image stabilizing system. IEEE Trans. on Consumer Electronics 39(3), 461–466 (1993)

4. Kurazume, R., Hirose, S.: An Experimental Study of Teleoperation System for Walking Robots Using High-Speed Image Stabilization System. Journal of the Robotics Society of Japan
5. Chang, J.Y., Hu, W.F., Cheng, M.H., Chang, B.S.: Digital Image Translational and Rotational Motion Stabilization Using Optical Flow Technique. IEEE Transactions on Consumer Electronics 48(1) (February 2002)
6. Hayashi, K., Yokokohji, Y., Yoshikawa, T.: Tele-existence Vision System with Image Stabilization for Rescue Robots. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, Barcelona, Spain, April 2005, pp. 50–55 (2006)
7. Nikon: VR Technology, `http://www.nikon.com/about/technology/core/software/vr_e/index.htm` (12.02.2009)
8. In cameras with lens-shift stabilization, `http://www.photoreview.com.au/guides/digitalphotography/shooting-tips.aspx`
9. Minolta, Konica: body-integral CCD-Shift mechanism, `http://ca.konicaminolta.com/products/consumer/digital_camera/slr/dynax-7d/02.html`
10. PENTAX Shake Reduction Technology, `http://www.pentaximaging.com/files/scms_docs/shake_reduction_fact_sheet.pdf`
11. Electronic Image Stabilization, `http://www.ovation.co.uk/Video-Stabilization.html`
12. Ko, S.J., Lee, S.H., Lee, K.H.: Digital image stabilizing algorithms based on bit-plane matching. IEEE Transactions on Consumer Electronics 44(3) (August 1998)
13. Ibrahim, F.A.: Optimal Linear Neuron Learning and Kalman Filter Based Back propagation Neural Network for DGPS/INS Integration. In: Position, Location and Navigation Symposium, 2008 IEEE/ION (2008)
14. Golik, B.: Development of a Test Method for Image Stabilizing Systems., Diploma Thesis at the Department of Imaging Sciences and Media Technology Cologne University of Applied Sciences, October 21 (2006)
15. Uomori, K., Morimura, A., Ishii, H., Sakaguchi, T., Kitamura, Y.: Automatic Image Stabilizing System by Full-Digital Signal Processing. IEEE Transactions on Consumer Electronics 26, 510–519 (1990)
16. Peng, Y.-C., Chang, H.-A., Homer, H.: Digital image stabilization and its integration with video encoder, pp. 544–549. IEEE, Los Alamitos (2004)
17. Erturk, S.: Real-Time Digital Image Stabilization Using Kalman Filters. Real-Time Imaging 8, 317–328 (2002)

Diffusing Method for Unknown Environment Exploration in Multi Robot Systems

Dilshat Saitov[1], Ki Joon Han[2], and Suk Gyu Lee[1]

[1] Dept. of Electrical Eng., Yeungnam University,
214-1 Dae-dong, Gyongsan, Gyongbuk, Korea 712-749
[2] School of Electrical Eng., and Computer Sci. Kyungbuk Nattional University
dilshat_saitov@yahoo.com, sglee@ynu.ac.kr

Abstract. This paper proposes an algorithm for an efficient navigation and building a precise map in multi-robot systems. One of the fundamental problems in mobile robotics is an effective investigation of unknown environments. The basis of navigation algorithm in this paper is Extented Wave Algorithm, which is in our point of view, appropriate in getting accurate.Secondly, particle filter, which proved its reliability, was considered as localization algorithm. Finally, overlapping algorithm is responsible for mapping. The technique has been tested extensively in simulation runs. The results given in this paper demonstrate that our algorithm significantly reduces the exploration time compared to previous approaches.

Keywords: navigation, mapping, localization, multi-robot system, wave algorithm.

1 Introduction

A lot of researches have been done about map-building by autonomous mobile robot [2, 3, 4, 5]. An autonomous mobile robot will definitely work well, if it is equipped with different kinds of precious sensors. But it may be too expensive and less fault tolerance. So there exist some problems such as getting accurate information, enhancing the system's fault tolerance and reducing system cost. Recently, in order to cope with dynamic environment and multiple tasks, a lot of researches have been presented in multi-agent and multiple distributed autonomous robotic systems. Multi-robot system has some features: distributed control, autonomy, enhanced fault tolerance and communication. So we decided to use a team of mobile for building a map in our simulation. Each robot has its own task, such as building a map of local position. Moreover, they have to combine their data into shared maps, Using shared maps, robots coordinate their exploration strategies to maximize the efficiency of exploration. In our multiple simulations we tried to create the system more flexible, robust and efficient. So, multi-robot system will solve the map-building task better. Multiple autonomous mobile robots can complete the task through cooperation and give a more accurate map by data fusionThe robots coordinate their exploration strategies to maximize the efficiency of their exploration using these shared maps. In our multiple simulations, we tried to create a more flexible, robust, and efficient system. A multi-robot system is better able to accomplish the map-building task because the robots cooperate and assemble a more accurate map by combining all their data.

2 Navigation

2.1 Wave Algorithm and Extended Wave Algorithm

This part of paper generally describes wave algorithm [8] and more deeply extended wave algorithm.

In this section, we describe the algorithm that we presented in Saitov [8]. In general, the wave algorithm uses costs and frontier cells when choosing the best target point, just like the uncoordinated algorithm [7]. We explain the strategy in the following figure. It provides short trajectories for single robot exploration tasks. First, the robot explores the area around itself and then seeks a minimum distance to the next point. [4]

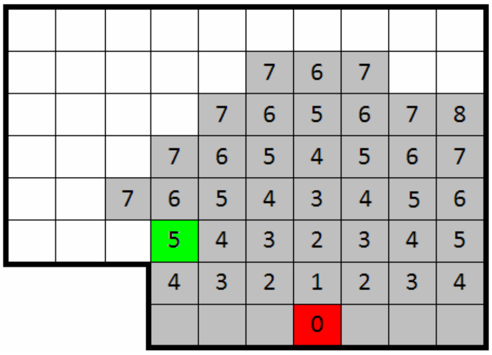


Fig. 1. The concept of the extended wave algorithm

According to improved wave algorithm, the robot will not head for cell number 5, as it was in the wave algorithm, because 5 is the smallest number among others which contiguous with "unknown" area, but find the largest number, divide it for 2 and takes its way to that cell. In case of situation depicted on figure 1, the robot will go to one of the cell numbered as 4.

Table 1. Pseudo code of the extended wave algorithm

```
wM [ cX, cY ] = 0;
i = 0;
a[n];
large=a[0];
exist = false
 do
{
   exist = false;
    for (x = 0; x < width; x ++)
      for (y = 0; y < width; y ++)
        if (dM [x, y] is open and free) and
             wM [x, y] is unassigned and
          neighbor of wM [x, y] is i)
        then (wM [x, y] = i +1; exist = true;)
   i = i + 1;
 }
while (exist);
   for(k=1;k<n;n++)
   {
    if(large<a[k])
   {
    large=a[k]
   }
   }
 k = large / 2
```

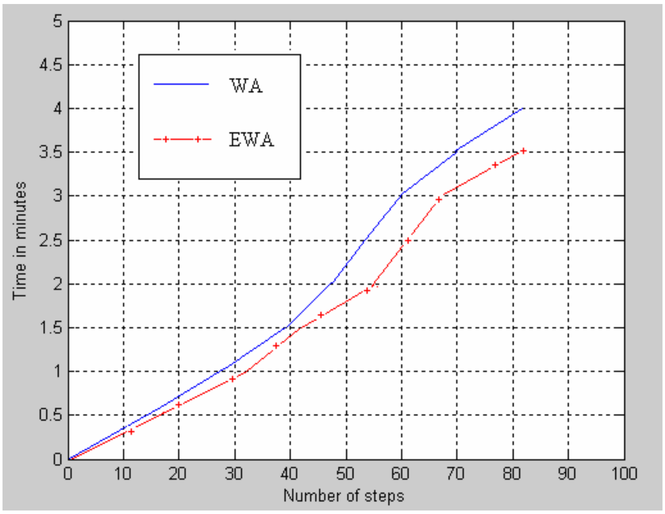


Fig. 2. Simulation results of WA and EWA

Figure 2 shows simulation of wave algorithm (WA) and extended wave algorithm (EWA).

3 Localization

3.1 Particle Filter

Localization is one of the significant segments of in mobile robotics. It is indispensable that a mobile robot estimates its own location for reaching a goal. In this section we described self-localization technique based on particle filtering. The main goal of partcile filtering is tracking a mobile robot. Multiple samples (particles) of the variable of interest are used, each one assosiated with a weght that signifies the quality of that specific particle. The weight sum of all samples guarantees an estimate of the variable of interest. There are two main parts in the particle filter algorithm: prediction and update. On the prediction step every single particle is modified according to the existing model. After that, on the update step, weight of every particle is reevaluated based on latest sensory information available and particles with small weights are expeled.This expelling process is called resampling.

Table 2. Pseudo code of the particle filter

```
Require: A set of particles for robot i at time 0:
while (exploring) do
     ;
     if   then
         {particle population depleted}
          Index = Resample  ;

      end  if
         for (j = 1 to M) do
                {prediction after action  }

         end for
         s= Sense()
     for (j = 1 to M) do (update the weights)

     end for
     for (j = 1 to M) do (normalize the weights)

     end for
end while
```

3.1.1 Prediction

In order to define robot's position at time $t+1$ we have to initiate the robot with its initial pose $[x, y, \hat{\theta}]^T$ at time t. At the beginning the robot rotates by $\delta\hat{\theta} = \hat{\theta}_k - \hat{\theta}$, where $\hat{\theta}_k = \arctan(\Delta y / \Delta x)$ to confront the destination position, after that it translates forward by distance $\rho = \sqrt{\Delta x^2 + \Delta y^2}$. As stated before the initial position is $[x, y, \hat{\theta}]^T$, the destination position will be $[x', y', \hat{\theta}]^T$

$$\begin{bmatrix} x' \\ y' \\ \hat{\theta}' \end{bmatrix} = \begin{bmatrix} x + \rho\cos(\hat{\theta}_k) \\ y + \rho\sin(\hat{\theta}_k) \\ \hat{\theta}_k \end{bmatrix} \tag{1}$$

3.1.2 Update

In the *update step*, each particle is weighted in proportion to the likelihood of the observation at time t, Yt, i.e.

$$\omega_t^{(i)} = \frac{g_t(Y_t \mid \overline{X}_t^{(i)})}{\sum_{i-1}^{N} g_t(Y_t \mid \overline{X}_t^{(i)})} \tag{2}$$

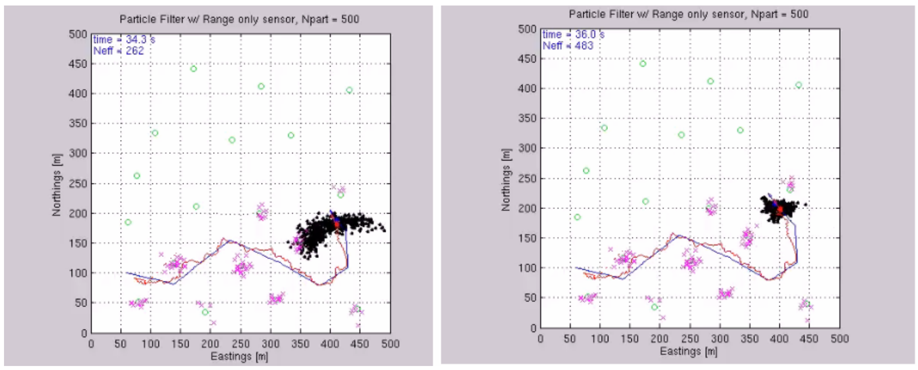


Fig. 3. Simulation results of patcile filter

4 Map Building

We used map merging, in which we consider each explored part of a map collected by a robot to be a separated matrix.[4] Let us assume that A and B are two positive real numbers. A * B is a function such that

$$m : [0, A] * [0, B] \rightarrow R \tag{3}$$

In Eq. (3), *A* and *B* represent rows and columns of the matrix, respectively. The next step is the definition of a transformation used to try different relative placements of two maps to find a good merge. We assume that the location of a point in the plane is expressed in homogeneous coordinates; that is, the point (*x*, *y*) is represented by the vector [*x y* 1]^T, where the trailing superscript *T* indicates the transpose operation.

Let t_x, t_y, and φ be three real numbers. The transformation associated with t_x, t_y and φ is the function

$$T\ t_x, t_y \varphi(x, y) : R^2 \rightarrow R^2 \tag{4}$$

defined as follows:

$$T\, t_x, t_y \varphi(\, x, y\,) : \begin{bmatrix} \cos\varphi & -\sin\varphi & tx \\ \sin\varphi & \cos\varphi & ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{5}$$

Here, the transformation corresponds to a rotation about the origin of the point (x, y) of φ, followed by a translation by (t_x, t_y). In the following sections, we see in the results that the mapping algorithms produce occupancy grids in which the rotation and translation transformations are apparently sufficient for merging real-world data.

Building a shared map using two small maps requires the best possible overlap between these two maps.[7, 8, 10] Let m_1 and m_2 be two maps in $I_{A*B.}$ The overlap between the maps can be described in the following equation:

$$\omega(m_1, m_2) = \sum_{i=0}^{A-1} \sum_{j=0}^{B-1} Eq(m_1[i, j]; m_2[i, j]) \tag{6}$$

where Eq (a;b) is 1 when a = b, and 0 otherwise. The overlapping function ω measures how much the two maps agree.

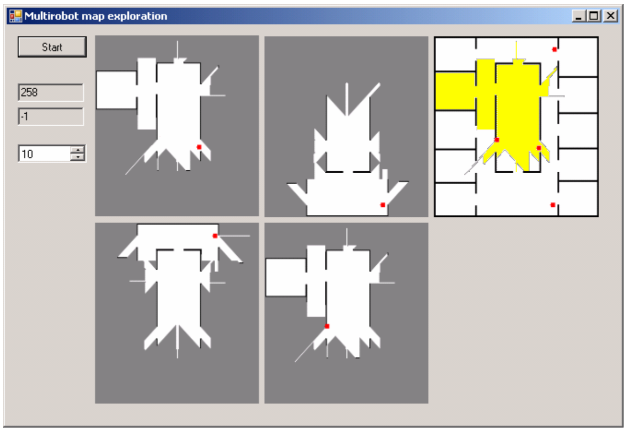


Fig. 4. Map mergiing

Figure 4 illustrates the moment when two robots meet and share their local maps into one synchronized map.

5 Simulation Results

Figure 5 represents total exploration time versus number of steps for group of 4 robots.The upper (red) line figures extended wave algorithm, the lower (green) line figures wave algorithm. It can be seen that at the same number of steps time values are significantly different.Thus we conlude that the developed navigation algorithm significantly decreases total exploration time.

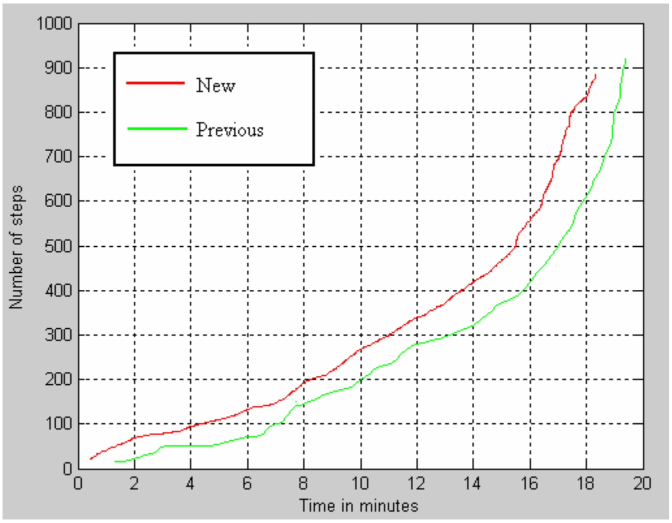


Fig. 5. Time vs Number of steps

6 Conclusions

We presented a distributed approach to mobile robot mapping and exploration. This system enables teams of robots explore environment more efficiently from either unknown or know locations. The robots explore independently from each other until they can communicate with other robots. When they meet each other, they can exchange sensor information with other robots, share explored maps and build one shared map. According to our simulation results this approach might be very useful for map building in multi-robot systems in real world and compared with [10].

Acknowledgment. This work was supported by the Korea Research Foundation(KRF) grant funded by the Korea government(MEST)(2009-0074464).

References

1. Dissanayake, G., Newman, P., Clark, S., Durrant-Whyte, H., Csorba, M.: BA solution to the simultaneous localization and map building (SLAM) problem. IEEE Trans. Robot. Autom. 17(3), 229–241 (2001)

2. Ko, J., Stewart, B., Fox, D., Konolige, K., Limketkai, B.: BA practical, decision-theoretic approach to multi-robot mapping and exploration. In: Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS), pp. 3232–3238 (2003)
3. Pulford, G.W., LA Scala, B.F.: Map estimation of target manoeuvre sequence with expectation-maximization algorithm. IEEE Trans. On aerospace and electronic systems 38(2) (April 2002)
4. Williams, S., Dissanayake, G., Durrant-Whyte, H.: Towards multi-vehicle simultaneous localization and mapping. In: Proc. 2002 IEEE Int. Conf. Robotics and Automation (ICRA), pp. 2743–2748 (2002)
5. Yamauchi, B.: Frontier-based exploration using multiple robots. In: Proc. of the Second International Conference on Autonomous Agents (1998)
6. Birk, A., Carpin, S.: Merging Occupancy Grid Maps From Multiple Robots. IEEE Trans. Robot. Autom. 94, 1384–1397 (2006)
7. Roumeliotis, S., Bekey, G.: BDistributed multirobot localization. IEEE Trans. Robot. Autom. 18(5), 781–795 (2002)
8. Burgard, W., Moors, M., Schneider, F.: BCollaborative exploration of unknown environments with teams of mobile robots. IEEE Trans. Robot. Autom. 18(5), 781–795 (2002)
9. Stachniss, C., Mozos, O.M., Burgard, W.: Speeding-up multi-robot exploration by considering semantic place information. IEEE Trans. Robot. Autom. 18(5), 581–587 (2006)
10. Stachniss, C., Mozos, O.M., Burgard, W.: Coordinated multi-robot exploration. IEEE Trans. Robot. 21(3), 376–386 (2005)
11. Saitov, D., Umirov, U., Lee, S.G., Park, J.I.: International Symposium on Mechatronics and Automatic Control Section. In: (ISMA), pp. 116–122 (October 2007)

Impulsive Consensus Seeking in Delayed Networks of Multi-agents

Quanjun Wu[1], Lan Xiang[2], and Jin Zhou[1,⋆]

[1] Shanghai Institute of Applied Mathematics and Mechanics, Shanghai University, Shanghai 200072, China
jinzhousu@yahoo.com.cn
[2] Department of Physics, School of Science, Shanghai University, Shanghai 200444, China

Abstract. This present paper addresses impulsive consensus problem in directed networks of dynamic agents having communication delays. Based on impulsive control theory on delayed dynamical systems, a simple impulsive consensus protocol for such networks is proposed, and a generic criterion for solving average consensus problem is analytically derived. It is shown that global average consensus of a directed delayed networked multi-agent systems can be achieved by a suitable design of the impulsive gain and impulsive interval. Simulations are presented that are consistent with the theoretical results.

Keywords: average consensus, impulsive consensus protocol, directed delayed networks, multi-agent systems, communication delays.

1 Introduction

The distributed coordination and cooperative control in dynamic networks of multiple agents have attracted increasing attention from various fields of science and engineering in recent years. This problem arises in several application domains, ranging from cooperative control of unmanned air vehicles (UAVs), formation control of autonomous vehicles, design of distributed sensor networks, congestion control in communication networks, etc., [1,2,3,4]. A critical problem in distributed coordinated control of multi-agents is to design appropriate protocols and algorithms such that all agents can reach an agreement regarding a certain quantity of interest that depends on the states of all agents. This problem is usually called the consensus problem. It has been paid attention for a long time by computer scientists, particularly in the field of automata theory and distributed computation [1,2,3,4,5,6,7,8,9].

To achieve cooperative consensus, a series of works have been performed recently [1,2,3,4,5,6,7,8,9]. For example, with a proposed simple model and neighbor-based rules, flocking and schooling were successfully simulated and analyzed for self-propelled particles by Vicsek et al. [1]. Jadbabaie et al. provided

⋆ Corresponding author.

a theoretical explanation of the consensus behavior of the Vicsek model using graph theory [2]. Fax et al. emphasized the role of information flow and graph Laplacians, and derived Nyquist-like criterion for stabilizing vehicle formations [3]. Olfati-Saber et al. investigated the consensus problems under a variety of assumptions on the network topology with respect to fixed or switching, presence or lack of communication delays, and directed or undirected network information flow [4]. Moreau studied the nonlinear discrete-time multi-agent systems with time-dependent communication channels and introduced a novel method based on the notion of convexity [5].

On the other hand, many biological and physical systems existing in the real world, such as biological neural networks, ecosystem management, automatic control systems, distributed computer networks and artificial intelligence robotics, usually undergo sudden and abrupt changes of states at certain moments. Such system can be well characterized by the form of impulses [10,11,12]. Consequently, there exist impulsive effects naturally in dynamic networks of multiple agents due to various instantaneous perturbations, including node and link failures/creations, packet-loss, asynchronous consensus, switching phenomenon and among many others [10,11,12]. However, so far, just few works considered consensus problems when network communication is affected by impulsive effects. Therefore, it is necessary to study consensus issue in dynamic networks of multi-agents associated with impulsive effects. This paper is an attempt toward this goal.

The primary contribution of this paper is to propose a simple distributed impulsive consensus protocol for a directed networks of dynamic agents having communication delays. The paper is organized as follows. In Section 2, some mathematical preliminaries are presented. In Section 3, a simple impulsive consensus protocol is proposed, and then a generic criterion for solving average consensus problem is analytically derived. Some simulation results are provided in Section 4. The conclusion is finally given in Section 5.

2 Preliminaries

Throughout this paper, the following notations and definitions will be used.

Let $R = (-\infty, +\infty)$ be the set of real numbers, and $N = \{1, 2, \cdots\}$ be the set of positive integer numbers. R^n be the n-dimensional real space equipped with Euclideam norm $\|\cdot\|$. For the vector $x = (x_1, x_2, \cdots, x_n)^\top \in R^n$, $x^\top$ denotes its transpose. $R^{n\times n}$ stands for the set of $n \times n$ real matrixes, for the matrix $A = (a_{ij})_{n\times n} \in R^{n\times n}$, $A^\top$ denotes its transpose, $A^s = (A + A^\top)/2$ stands for the symmetric part of A. $\lambda_{\max}(\cdot)$ denotes the maximum eigenvalue of the matrix. The spectral norm of A is defined as $\|A\| = [\lambda_{\max}(AA^\top)]^{1/2}$. E is the identity matrix of order n. Matrix dimensions, if not explicitly stated, are assumed to be compatible for algebraic operations.

Let $PC([-\tau, 0], R^n) = \{\varphi : [-\tau, 0] \to R^n, \varphi(t)$ is continuous everywhere except a finite number of points $\hat{t}$ at which $\varphi(\hat{t}^+)$ and $\varphi(\hat{t}^-)$ exist and $\varphi(\hat{t}^+) = \varphi(\hat{t})\}$. Given a constant $\tau > 0$, we equip the linear space $PC([-\tau, 0], R^n)$ with the norm $\|\cdot\|_\tau = \sup_{-\tau\le s\le 0}\|\psi(s)\|$, and for $\varphi : R \to R$, denote $\varphi(t^+) = \lim_{s\to 0^+}\varphi(t+s)$, $\varphi(t^-) = \lim_{s\to 0^-}\varphi(t+s)$, $[\varphi(t)]_\tau = \sup_{-\tau\le s\le 0}\{\varphi(t+s)\}$, $[\varphi(t)]_{\tau^-} = \sup_{-\tau\le s<0}\{\varphi(t+s)\}$.

Lemma 1. *Suppose $\beta \ge 0$, and $v(t)$ satisfies the scalar impulsive differential inequality:*

$$\begin{cases} D^+v(t) \le -\alpha v(t) + \beta[v(t)]_\tau, \quad t \ne t_m,\ \ t \ge t_0, \\ v(t_m^+) \le b_m v(t_m^-) + d_m[v(t_m)]_{\tau^-},\ \ m \in N, \\ v(t) = \varphi(t), \quad t \in [t_0-\tau, t_0], \end{cases} \tag{1}$$

where $v(t)$ is continuous at $t \ne t_m, t \ge t_0$, $v(t_m) = v(t_m^+)$ and $v(t_m^-)$ exists, $\varphi \in PC([t_0-\tau, t_0], R)$. Assume that there exist constants $\lambda > 0$ and $\sigma > 0$, such that for all $m \in N$, the following conditions are satisfied:

(i) $0 < b_{m-1} + d_{m-1}e^{\lambda\tau} \le 1$;

(ii) $\sigma - \lambda - (-\alpha + \dfrac{\beta e^{\lambda\tau}}{b_{m-1} + d_{m-1}e^{\lambda\tau}}) \ge 0$;

(iii) $(\sigma+\lambda)(t_m - t_{m-1}) < -\ln(b_{m-1} + d_{m-1}e^{\lambda\tau})$.

Then there exists a constant $M \ge 1$ such that

$$v(t) \le \|\varphi\|_\tau M e^{-\lambda(t-t_0)}, \quad t \ge t_0.$$

The proof of Lemma 1 is partly similar to that of Theorem 1 in [12], and is omitted from this paper due to the limitation of space.

3 Impulsive Consensus Protocol

Suppose that the network system under consideration consists of n agents. Each agent is regarded as a node in a directed graph G. Here we assume that the communication topology of G is balanced and has a spanning tree. Let x_i be the state of the ith agent. Suppose each agent has the dynamics as follows:

$$\dot{x}_i(t) = u_i(t), \tag{2}$$

where $u_i(t)$ is the control input (or protocol) with topology at time t if they only depend on the state of agent i and its neighors.

In this paper, we are interested in discussing average consensus problem in directed networks of dynamic agents having time-delays as well as fixed topology with impulsive effects, where the information (from v_i to v_j) passes through edge (v_i, v_j) with time-varying communication delays $0 < \tau(t) \le \tau$. To solve such a problem, we use the following impulsive consensus protocol:

$$u_i(t) = \sum_{v_j\in N_i} a_{ij}\Big(x_j(t-\tau(t)) - x_i(t-\tau(t))\Big)Dw_{ij}(t), \tag{3}$$

where D denotes the distributional derivative, $w_{ij} : J = [t_0, +\infty) \to R$ are functions of bounded variations which are right-continuous on any compact subinterval of J. Dw_{ij} represents the effects of sudden changes in the states of the system at the discontinuity points of w_{ij}.

Under the consensus protocol (3), the system (2) has the following form:

$$Dx_i(t) = \sum_{v_j \in N_i} a_{ij}\Big(x_j(t-\tau(t)) - x_i(t-\tau(t))\Big)Dw_{ij}(t). \tag{4}$$

Without loss of generality, we therefore assume that

$$Dw_{ij} = 1 + \sum_{m=1}^{\infty} \mu_{ij}^{(m)}\delta(t-t_m), \tag{5}$$

where $\delta(t)$ is the Dirac impulsive function, which means that the states of system (4) has jumps at t_m. For the measure differential equation (4), their initial condition is given by $x_i(t) = \phi_i(t) \in PC([t_0-\tau, t_0], R^n)$. We always assume that Eq. (4) has at least one solution with respect to initial conditions [10,11,12].

Remark 1. If $\mu_{ij}^{(m)} = 0$, then the model (4) becomes continuous consensus scheme with time-delays:

$$\dot{x}_i(t) = \sum_{v_j \in N_i} a_{ij}\Big(x_j(t-\tau(t)) - x_i(t-\tau(t))\Big). \tag{6}$$

System (6) has been studied by many authors [4,8,9]. Therefore, the impulsive consensus protocol (3) is a generalization of the existing general consensus protocols.

Rewrite Eq. (4) in matrix form as:

$$Dx(t) = -(L(G) \odot Dw(t))x(t-\tau(t)), \tag{7}$$

where $L(G) = [l_{ij}]_{n\times n}$ is called graph Laplacian (Laplacian matrix) induced by the information flow G and is defined by

$$l_{ij} = \begin{cases} \sum\limits_{k=1,k\neq i}^{n} a_{ik}, & j = i, \\ -a_{ij}, & j \neq i. \end{cases}$$

As L is a balanced matrix, we have $\mathbf{1}_n^\top L = \mathbf{0}$, which implies $\sum_i \dot{x}_i = 0$. Thus, $\alpha = \sum_i x_i(0)/n = Ave(x)$ is an invariant quantity. The invariance of $Ave(x)$ allows decomposition of x according to the following equation:

$$x = \alpha\mathbf{1} + \eta, \tag{8}$$

where $\alpha = Ave(x)$ and $\eta = (\eta_1, \cdots, \eta_n)^\top \in R^n$ satisfies $\mathbf{1}^\top\eta = 0$. Here, we refer to η as the (group) disagreement vector. The vector η is orthogonal to $\mathbf{1}$ and belongs to an $(n-1)$-dimensional subspace. Moreover, η evolves according to the (group) disagreement dynamics given by

$$D\eta(t) = -(L(G) \odot Dw(t))\eta(t-\tau(t)). \tag{9}$$

For convenience, denote $L^{(m)} = [l_{ij}\mu_{ij}^{(m)}]_{n\times n}$, the following sufficient conditions for average consensus of the system (7) are established based on stability theory on impulsive delayed dynamical systems.

Theorem 1. *Consider the impulsive delayed dynamical networks (7). Assume there exist constants $\lambda > 0$, $\sigma > 0$ and $\varepsilon > 0$, such that for all $m \in N$, the following conditions are satisfied:*

(A$_1$) $b_m + d_m e^{2\lambda\tau} \leq 1$;

(A$_2$) $\sigma - \lambda - (-\alpha + \dfrac{\beta e^{2\lambda\tau}}{b_m + d_m e^{2\lambda\tau}}) \geq 0$;

(A$_3$) $(\sigma + \lambda)(t_{m+1} - t_m) < -\ln(b_m + d_m e^{2\lambda\tau})$;

where

$$\begin{aligned} b_m &= \frac{1 + \varepsilon\tau p + \varepsilon\varsigma q}{1 + 2\lambda_2(L^{(m)s}) + \lambda_2(L^{(m)\top}L^{(m)})}, \\ d_m &= \frac{\varepsilon^{-1}\tau p + \varepsilon^{-1}\varsigma q + \tau^2 p^2 + (\varepsilon\tau^2 + \varepsilon^{-1}\varsigma^2)p \cdot q + \varsigma^2 q^2}{1 + 2\lambda_2(L^{(m)s}) + \lambda_2(L^{(m)\top}L^{(m)})}, \\ \alpha &= 2\lambda_2(L^s) - \varepsilon\tau\|L\|^2 - \varepsilon\varsigma r, \quad \beta = \varepsilon^{-1}\tau\|L\|^2 + \varepsilon^{-1}\varsigma r, \end{aligned}$$

and $p = \|L\| \cdot \|L^{(m)}\|$, $q = \|L^{(m)}\| \cdot \max_{1 \leq p \leq \varsigma} \|L^{(s_p)}\|$, $r = \|L\| \cdot \max_{1 \leq p \leq \varsigma} \|L^{(s_p)}\|$.

Then the dynamical networks (7) achieve average consensus globally exponentially.

Proof. Suppose that the frequency of impulses is ς at most in the time interval $[t - \tau, t]$, that is, $\varsigma = \left[\dfrac{\tau}{\min_{m \in N}\{t_m - t_{m-1}\}}\right] + 1$, and $[\xi]$ denotes the maximum integer no larger than ξ, the states of system have jump at t_{s_1}, t_{s_2}, $\cdots$, t_{s_ς}. Since the graph G has a spanning tree, then its Laplacian L has exactly one zero eigenvalue and the rest $n - 1$ eigenvalues all have positive real-parts (see [6], Lemma 3.3). Furthermore, L^s is a symmetric matrix and has zero row sums. Thus, the eigenvalues of matrix L^s can be ordered as

$$0 = \lambda_1(L^s) < \lambda_2(L^s) \leq \cdots \leq \lambda_n(L^s).$$

Since L^s is symmetric, by the basic theory of Linear Algebra we know

$$\eta^\top(t)L^s\eta(t) \geq \lambda_2(L^s)\eta^\top(t)\eta(t), \quad \text{if } \mathbf{1}^\top\eta = 0.$$

Without loss of generality, suppose that the frequency of information exchange is ς in the time interval $[t_m - \tau, t_m]$, now we rewrite the Eq. (9) as

$$\begin{aligned} D\eta(t) = &-(L \odot Dw(t))\eta(t) + (L \odot Dw(t))\Big(\int_{t-\tau(t)}^{t_{s_1}^-} \dot\eta(s)ds + \int_{t_{s_1}}^{t_{s_2}^-} \dot\eta(s)ds \\ &+ \cdots + \int_{t_{s_\varsigma}}^{t} \dot\eta(s)ds\Big) - (L \odot Dw(t))\sum_{p=1}^{\varsigma} L^{(s_p)}\eta(t_{s_p} - \tau(t_{s_p})). \end{aligned} \quad (10)$$

Let us construct the Lyapunov function as the following:

$$V(t) = \frac{1}{2}\eta^\top(t)\eta(t). \quad (11)$$

For $t \neq t_m$, by calculating the upper right-hand derivative of (11) along the solutions of (10), we have

$$\begin{aligned} D^+V(t) \leq & -\Big[2\lambda_2(L^s) - \varepsilon\tau\|L\|^2 - \varepsilon\varsigma\|L\| \cdot \max_{1\leq p\leq\varsigma} \|L^{(s_p)}\|\Big]V(t) \\ & +\Big[\varepsilon^{-1}\tau\|L\|^2 + \varepsilon^{-1}\varsigma\|L\| \cdot \max_{1\leq p\leq\varsigma} \|L^{(s_p)}\|\Big] \sup_{t-2\tau\leq s\leq t} V(s). \end{aligned} \tag{12}$$

Therefore, from the conditions (A$_1$) – (A$_3$) of Theorem 1 and (12), we get

$$D^+D(t) \leq -\alpha V(t) + \beta[V(t)]_{2\tau}. \tag{13}$$

On the other hand, by using the properties of Dirac measure, we have

$$\begin{aligned} \Big(E + L^{(m)}\Big)\eta(t_m) = \eta(t_m^-) - L^{(m)}\Big[\int_{t_m-\tau(t_m)}^{t_{s_1}^-} L\eta(s-\tau(s))ds + \int_{t_{s_1}}^{t_{s_2}^-} L \\ \times\eta(s-\tau(s))ds + \cdots + \int_{t_{s_\varsigma}}^{t_m^-} L\eta(s-\tau(s))ds\Big] - L^{(m)}\sum_{p=1}^{\varsigma} L^{(s_p)}\eta(t_{s_p} - \tau(t_{s_p})). \end{aligned}$$

We can select $\mu_{ij}^{(m)}$ such that $L^{(m)}$ is Laplacian matrix. Therefore, we obtain

$$0 = \lambda_1\Big(L^{(m)s}\Big) < \lambda_2\Big(L^{(m)s}\Big) \leq \cdots \leq \lambda_n\Big(L^{(m)s}\Big), \tag{14}$$

and

$$0 = \lambda_1\Big(L^{(m)\top}L^{(m)}\Big) < \lambda_2\Big(L^{(m)\top}L^{(m)}\Big) \leq \cdots \leq \lambda_n\Big(L^{(m)\top}L^{(m)}\Big). \tag{15}$$

It then follows from (14) and (15) that

$$\begin{aligned} & 2\Big[1 + 2\lambda_2\Big(L^{(m)s}\Big) + \lambda_2\Big(L^{(m)\top}L^{(m)}\Big)\Big]V(t_m) \\ \leq\ & \eta^\top(t_m)(E + L^{(m)\top})(E + L^{(m)})\eta(t_m) \\ \leq\ & 2\Big[1 + \varepsilon\tau\|L\| \cdot \|L^{(m)}\| + \varepsilon\varsigma\|L^{(m)}\| \cdot \max_{1\leq p\leq\varsigma}\|L^{(s_p)}\|\Big]V(t_m^-) \\ & +2\Big[\varepsilon^{-1}\tau\|L\| \cdot \|L^{(m)}\| + \varepsilon^{-1}\varsigma\|L^{(m)}\| \cdot \max_{1\leq p\leq\varsigma}\|L^{(s_p)}\| \\ & +\tau^2\|L\|^2 \cdot \|L^{(m)}\|^2 + (\varepsilon\tau^2 + \varepsilon^{-1}\varsigma^2) \cdot \|L\| \cdot \|L^{(m)}\|^2 \cdot \max_{1\leq p\leq\varsigma}\|L^{(s_p)}\| \\ & +\varsigma^2\|L^{(m)}\|^2 \max_{1\leq p\leq\varsigma}\|L^{(s_p)}\|^2\Big] \sup_{t_m-2\tau\leq s\leq t_m} V(s) \end{aligned} \tag{16}$$

Hence, from the conditions (A$_1$) – (A$_3$) of Theorem 1 and (16), we have

$$V(t_m) \leq b_m V(t_m) + d_m[V(t_m)]_{2\tau^-}. \tag{17}$$

Thus, in view of (13) and (17), all the conditions of Lemma 1 are satisfied. Theorem 1 is proven.

Remark 2. The conditions given in Theorem 1 are all sufficient conditions but not necessary, i.e., the dynamical networks achieve average consensus globally exponentially, although one of the conditions in Theorem 1 may fail. This is also illustrated through numerical examples in the next section.

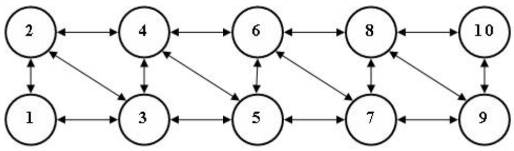


Fig. 1. An example of directed graph

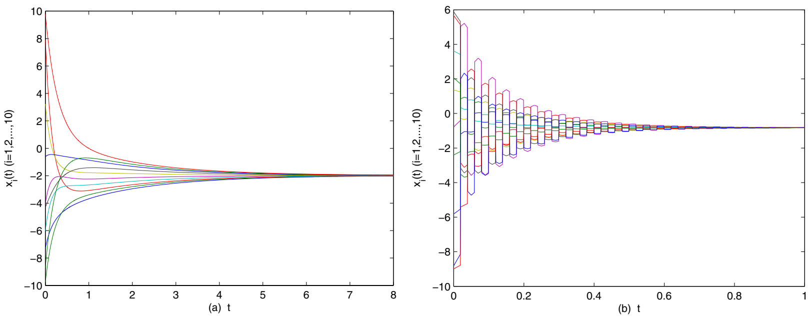


Fig. 2. Consensus process of the state variables in the delayed dynamical networks (7) with different impulsive gain μ. (a) $\mu = 0.003$; (b) $\mu = 0.3$.

4 Simulations

Here we consider a directed network with fixed topology G having 10 agents as in Fig.1. It is easy to see that G has a spanning tree. For simplicity, take the equidistant impulsive interval $t_m - t_{m-1} \equiv \Delta t = 0.02$, time-delay $\tau = 0.01$, $\lambda = 1$. And randomly choose initial value in $[-10, 10]$. Consider the impulsive gain $\mu_{ij}^{(m)} = \mu$ for $i, j = 1, 2, \cdots, 10$, $m \in N$, and $\varepsilon = 1$. Fig. 2 (a) is the simulation result corresponding to change process of the state variables of the delayed dynamical network (7) with impulsive gain $\mu = 0.003$, which satisfies the conditions of Theorem 1. And Fig. 2 (b) indicates the simulation corresponding to change process of the state variables of the delayed dynamical networks (7) with impulsive gain $\mu = 0.3$, which reveals that the conditions of Theorem 1 are sufficient conditions but not necessary, as discussed in Remark 2.

5 Conclusions

In this paper, a simple impulsive consensus protocol in directed delayed networks of dynamic agents with fixed topology has been proposed. A simple but less conservative sufficient condition under which all the nodes in the network achieve average consensus globally exponentially is analytically derived by employing impulsive control theory on delayed dynamical systems. It is shown that the consensus of the system not only depends on the topology of the entire networks and the communication time-delay, but also is heavily determined by the impulsive gain and impulsive interval.

Acknowledgment. This work was supported by the National Science Foundation of China (Grant Nos. 10672094, 10972129 and 10832006), the Specialized Research Foundation for the Doctoral Program of Higher Education (Grant No. 200802800015), the Innovation Program of Shanghai Municipal Education Commission (Grant No. 10ZZ61), the Shanghai Leading Academic Discipline Project (Project No. S30106), and Shanghai University Graduate Innovation Funds (Grant No. SHUCX091053).

References

1. Vicsek, T., Czirok, A., Jacob, E.B., Cohen, I., Schochet, O.: Novel Type of Phase Transition in a System of Self-Driven Particles. Phys. Rev. Lett. 75, 1226–1229 (1995)
2. Jadbabaie, A., Lin, J., Morse, A.S.: Coordination of Groups of Mobile Autonomous Agents Using Nearest Neighbor Rules. IEEE Trans. Autom. Contr. 48, 988–1001 (2003)
3. Fax, J.A., Murray, R.M.: Information Flow and Cooperative Control of Vehicle Formations. IEEE Trans. Autom. Contr. 49, 1465–1476 (2004)
4. Olfati-Saber, R., Murray, R.M.: Consensus Problems in Networks of Agents with Switching Topology and Time-Delays. IEEE Trans. Autom. Contr. 49, 1520–1533 (2004)
5. Moreau, L.: Stability of Multiagent Systems with Time-Dependent Communication Links. IEEE Trans. Autom. Contr. 50, 169–182 (2005)
6. Ren, W., Beard, R.W.: Consensus Seeking in Multiagent Systems Under Dynamically Changing Interaction Topologies. IEEE Trans. Autom. Contr. 50, 655–661 (2005)
7. Hong, Y.G., Hu, J.P., Gao, L.X.: Tracking Control for Multi-Agent Consensus with an Active Leader and Variable Topology. Automatica 42, 1177–1182 (2006)
8. Sun, Y.G., Wang, L., Xie, G.M.: Average Consensus in Networks of Dynamic Agents with Switching Topologies and Multiple Time-Varying Delays. Syst. Contr. Lett. 57, 175–183 (2008)
9. Lin, P., Jia, Y.M.: Average Consensus in Networks of Multi-Agents with both Switching Topology and Coupling Time-Delay. Physica A 387, 303–313 (2008)
10. Zhou, J., Xiang, L., Liu, Z.R.: Synchronization in Complex Delayed Dynamical Networks with Impulsive Effects. Phys. A 384, 684–692 (2007)
11. Cai, S.M., Zhou, J., Xiang, L., Liu, Z.R.: Robust Impulsive Synchronization of Complex Delayed Dynamical Networks. Phys. Lett. A 372, 4990–4995 (2008)
12. Zhou, J., Wu, Q.J.: Exponential Stability of Impulsive Delayed Linear Differential Equations. IEEE Trans. Circuit Syst. II 56, 744–748 (2009)

The Application of Multi-agent Technology on the Level of Repair Analysis

Xiangkai Liu[1], Yanfeng Tang[1], Lin Zheng[2], Bingfeng Zhu[1], and Jia-ning Wang[1]

[1] Department of Automobile, Academy of Military Transportation, Tianjin, China
[2] Simulation Center of Logistic, Academy of Logistic Commanding, Beijing, China

Abstract. The basic theory of level of repair analysis (LORA) has been discussed. The route to apply multi-agent technology to accomplish the computer aided analysis for LORA has been investigated. A LORA system based on multi-agent system has been presented, the structure, the non-economic analysis agent has been researched. It can overcome the problem of information sharing and the cooperation between analysts effectively, and provide a new route to accomplish the computer aided analysis for LORA.

Keywords: Maintenance Management, Level of Repair Analysis, Multi-Agent System.

1 Introduction

The determination of the repair level within the Army maintenance system is an essential element of the logistics management information (LMI). LMI will include a LORA or other analyses. A LORA shall be performed on all materiel [1].

LORA is an integral part of Logistic Support Analysis, and is a prescribed procedure for defense logistics planning. According to the factors of economic and non-economic, LORA is used to make sure whether the product needs repair and the best level to be repaired if needed, and it is the optimal balance decision plan between the design characters, support program and support resource requirements. The product in LORA may be equipment, a component, an assembly, or a part, etc.

With the launching of the Computer-Aided Logistic Support (CALS) initiative, lots of models and computer programs for logistic support analysis have been developed worldwide. LORA model is one of an important model of logistic support analysis (LSA) software. LORA program may be single software, e.g. Compass (LOGSA, U.S.), or just a part of a LSA software, e.g. RAM Commander (A.L.D., Israel), OPUS10 (Systecon, Sweden), CARMES (CEPREI, China), etc.

The LORA model (Program) is designed to assist the analyst in conducting a Level of Repair Analysis. The main functions of it include (1) Cost and allocation of manpower and equipment required to repair the item, (2) Repair and discard decision, (3) Economic and non-economic analysis, (4) Cost of initial and replenishment spares over system life, (5) Overall costs associated with transportation, cataloging, training, technical manuals, etc.

LORA is a rather complex procedure, even though a programming LORA program is available, the requirement of the analyst is rather strict, he (she) must be a specialist who owns lots of field knowledge, familiars with the structure, operation, and maintenance of the product analyzed, and has plenty of experiences. This problem has limited the application or reduced the effectiveness of LORA to a certain extent. How to decrease the requirement of analyst is an important problem for current LORA software.

Considering that a powerful built-in knowledge and model database system, which can improve the effectiveness of analysis procedure significantly, has been an essential part of a LORA program. In this paper, we will not discuss them, and the main study will focus on the route and technique for the application of multi-agent to a LORA system, and try to get an effective way to solve the difficulty, so as to provide technique support for a new kind of LORA system.

2 The Theory of LORA

LORA is one of the most important supportability analyses during acquisition of a system. LORA is performed in two steps:

a) Using non-economic decision criteria to make the initial support decisions.
b) Using an economic model to determine the most cost effective alternative to provide support for the system.

2.1 The Process of LORA

The LORA program should be integrated developed, and conjunction with other cost effective achievement of overall system objectives.

The input of LORA is very wide, such as the system specification, contract, regulations, reliability allocation report, maintainability prediction report, the level of repair and their task distributed, etc. The output is a final repair scheme for the system. The basic analysis object of LORA is a product (an item of the system). For each product, first of all, non-economic analysis should be carried out to make sure the reasonable Repair Level. When non-economic analysis can't make sure the maintenance level, then economic analysis should be carried out. After the optimization, the optimized final repair scheme will be gained. The repair scheme includes repair or discard decision, the maintenance allocation.

To accomplish LORA, a great deal of data should be collected, such as the manpower and device required, the breakdown tree and expected time etc.

2.2 Non-Economical Level of Repair Analysis (NELORA)

NELORA decision criteria are a list of rules or guidelines that are used to determine if there is an overriding reason why maintenance should be performed. Some organizations have policies that any item costing less than a predetermined price level will be discarded and replaced rather than be repaired.

When determining if an item or sub-item should be replaced and/or repaired at certain level of repair, consideration must be given to all aspects of the maintenance task.

Table 1. Non economic analysis

Non economic factors			O	I	D	reason for restriction
Safety	Do hazardous conditions exist which preclude the item from being repaired at any specified maintenance level?	High Voltage Radiation Temperature Extremes …				
Security	Do security conditions exist which preclude the item from being repaired at a specific maintenance level?					
policy/exist maintenance concepts	are there specifications, standards or regulations pertaining to the level of maintenance at which a particular item can or cannot be repaired?					
…	…	…				
other factors						

These include fault diagnostic techniques, accessibility requirements, task verification, elapsed task times, disruption to the operational environment etc.

2.3 Economical Level of Repair Analysis (ELORA)

The objective of the ELORA, when performed correctly, is to determine the optimum cost of repair of an end item/equipment, by taking into consideration all associated support cost drivers (e.g., manpower, support equipment, training, transportation, etc.) that would be required at each line of maintenance. There is an element in the LORA process that must not be overlooked, that is the Sensitivity Analysis. Varying different LCC cost drivers will determine their impact on the model's repair decisions.

Following the LORA, the ELORA provides a recommended repair strategy based upon economic rational. Consideration is also given to an item's maintainability and physical characteristics.

There are many ELORA models available that can be used for simple and complex applications[2],[3]. In these models, there are six major cost drivers as following:

a) Inventory. Includes inventory administration, attrition, repair material, scrap material, and transportation, etc.
b) Support Equipment (SE). The cost of providing the necessary SE (other than standard tools and SE) for completing a particular CM(corrective maintenance) task at a given level of maintenance.
c) Space. required for inventory storage, repair work, and support equipment.
d) Labor. The associated cost for personnel to complete each repair task.

e) Training cost. The cost for upgrading and providing the necessary training to personnel performing a maintenance task at a given maintenance site and/ or shop.
f) Documentation. The cost for developing the required documentation for each of the maintenance options.

3 The LORA System Based on Multi-Agent System

The agent technology started from the 1970s, belonging to the research area of Distributed Artificial Intelligence (DAI). It has many significant advantages, such as good flexibility and stability. It can be applied for complex and collaborative problem solving[4]. Multi-agent system is a loose coupled network architecture made up of several agents, and each agent is a software or hardware entity that has the ability of autonomy, self-maintenance, self-learning and artificial intelligence. In the process of problem solving, a number of agents cooperate to get a comprehensive solution by communication between each others[5]. In multi-agent system, the activity of each agent is autonomous, the goal and behavior is not limited by other agent members, and each agent has its own knowledge store system which can be helpful to use and acquire knowledge. The scalable and open architecture of multi-agent agent can be adapted to the situation that current equipments update more and more quickly.

Now, multi-agent systems have been applied widely in decision support systems [6]. Recently, it has been widely used in maintenance support field [7],[8],[9],[10].

3.1 The Structure of System

Fig. 1 shows the structure of the LORA based on multi-agent system. The system consists of many LORA system distributed at various positions, and they are linked by network. Each LORA system has the same structure, and it is consists of several agents. Each agent (module) inside has its individual beliefs and local state, its local knowledge and basic data. They can communicate with each other by coordinate agent, and can accomplish distributed problem-solving (DPS). The agents inside a LORA system are as the following:

a) The Interactive Agent. It is the unique human-machine interface of the system. It can communicate with current analyst (user), and other Agent inside. The analyst can input a analysis task(usually, is a product, such as a device, a system, a component, or a part), and get a desirable decision through a serious of procedures.
b) The Coordination Agent. It is the most important Agent in the system. It can anticipate problem-solving actively/passively, such as take charge of the information exchanging with each Agent and the Coordinate Agent from other systems, communication and conflict control, monitors the actions of each agent. Certainly, the basic tasks of it are accepting the analysis request from the Interactive Agent, and outputting the analysis results to the Interactive Agent.
c) The Case Agent. It has its own case database and case management system. The main function of it is to search the applicable cases in local case database, and to search the applicable cases from other LORA systems(if applicable) through the Coordinate Agent. If there are some applicable cases, the analyst can select to direct

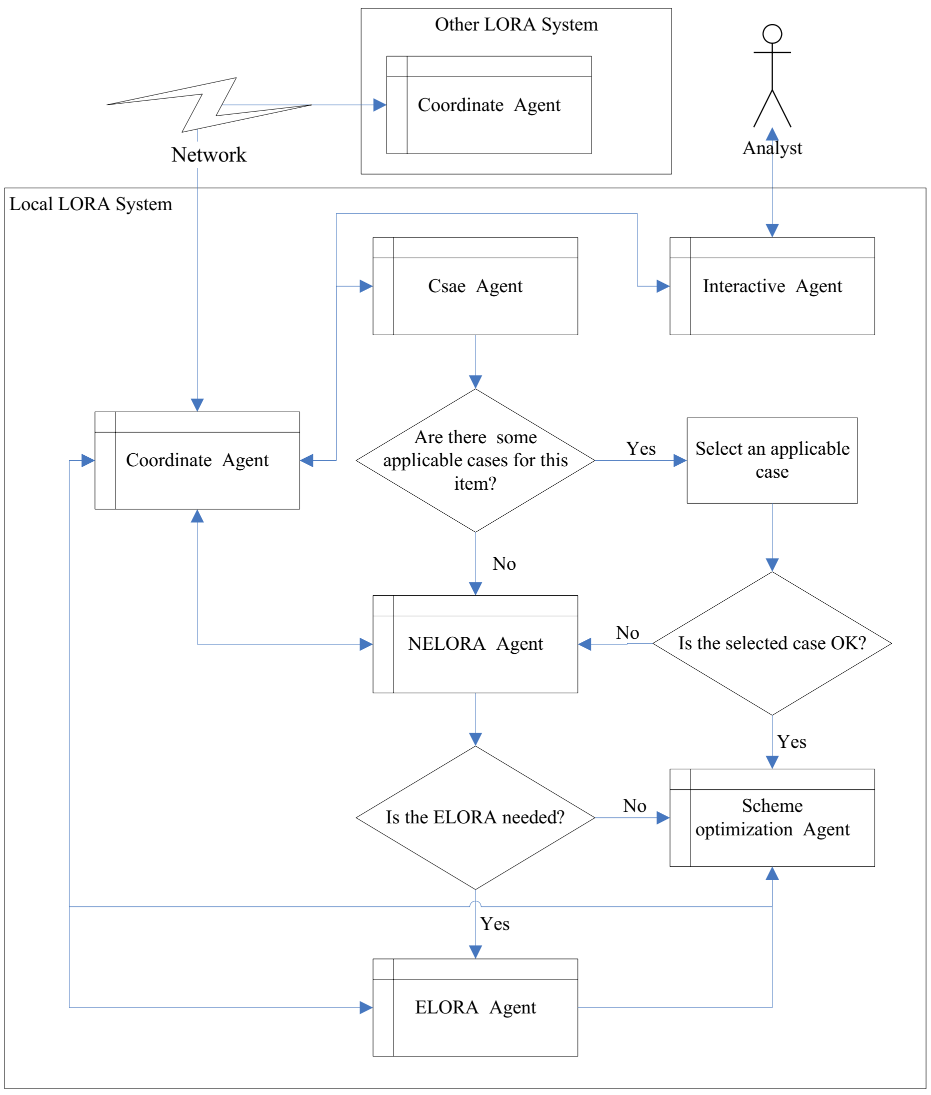


Fig. 1. The structure of the LORA based on multi-agent technology

accept the decision by the cases, modify the cases for a more desirable decision, or reject the cases. It also has the ability to active/interactive knowledge learning.

d) Non-economic Analysis Agent. It is in charge of non-economic analysis of LORA, such as determine whether the NELORA should be carried out for a certain product, if necessary, apply its local rules and reasoning methods to accomplish the task, or publish an analysis request to Coordinate Agent, or transfer the analysis results to Coordinate Agent.

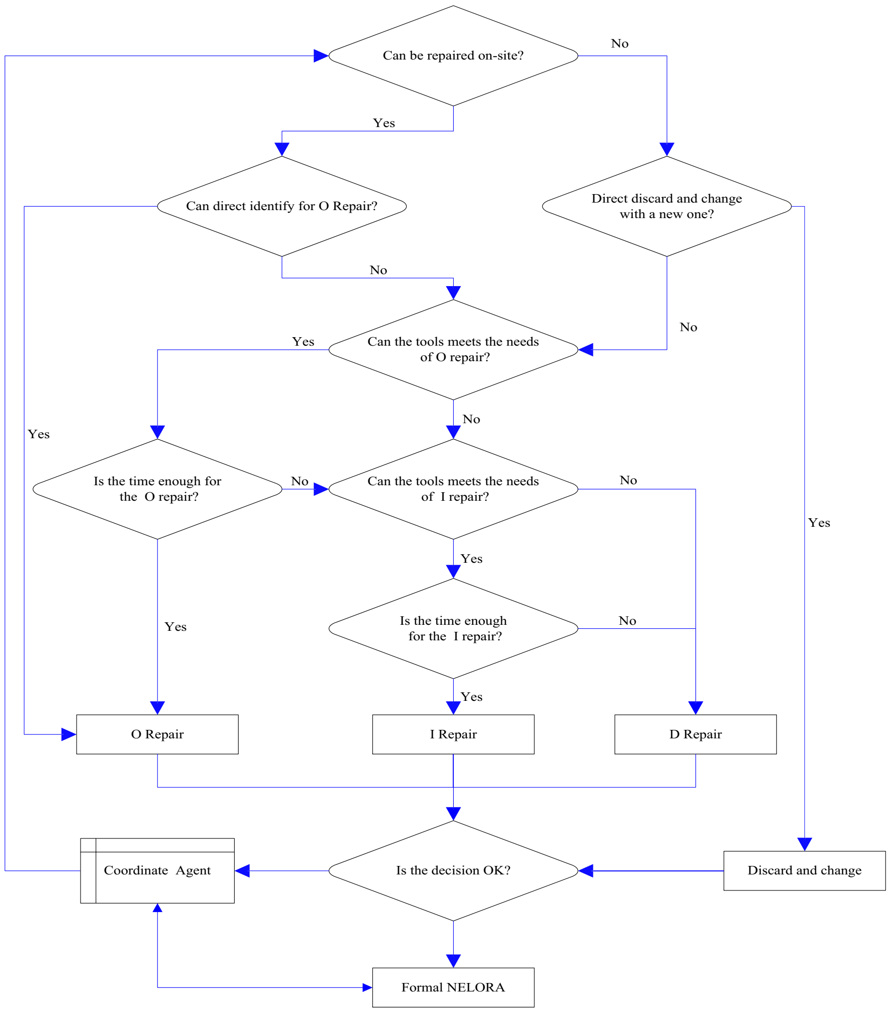

Can be repaired on-site?
No
Yes
Can direct identify for O Repair?
Direct discard and change with a new one?
No
No
Yes
Can the tools meets the needs of O repair?
No
Yes
Is the time enough for the O repair?
No
Can the tools meets the needs of I repair?
No
Yes
Yes
Yes
Is the time enough for the I repair?
No
Yes
O Repair
I Repair
D Repair
Coordinate Agent
Is the decision OK?
Discard and change
Formal NELORA

Fig. 2. The process of the NELORA agent

e) Economic Analysis Agent. The role of it is similar to NELORA, the main difference is the inter process and analysis results.
f) Scheme Optimization Agent. Optimize the repair scheme.

3.2 The Process of NELORA Agent

NELORA addresses preempting factors which override the considerations or existing LORA decisions on similar systems to determine the maintenance level where repair or discard can be performed. Preempting factors are normally a restraint, stipulation, or a special requirement which forces the repair or discard decision to a specific maintenance level or limits the support alternatives available.

In fact, in order to reduce the analysis workload, we can perform a non-formal analysis before the formal analysis. The process is briefly shows in Fig. 2.

In the system, the basic process is same as traditional analysis method, the key and significant different is that the users can share the information with each other by the Cooperate Agent.

When a user is prepared to make a decision, he can apply several methods to accomplish it:

a) Make a decision by himself.
b) Look for a successful case in local site or other systems.
c) Publish a decision making requirement or direct to ask for help to other LORA systems.

When a user has made a decision, he can apply several same methods to optimize and evaluate it.

4 Discussion

The proposed LORA system is different from traditional LORA system both in the structure and the work mode. The system can solve the information sharing and the cooperation between analysts more effectively, and provide a new route to accomplish the computer aided analysis for LORA.

Certainly, in this paper, only the basic structure and flow of the system has been discussed, there will a long way to realize it. Thorough research should be carried out in the future.

References

1. Army Regulation 700–127, Integrated Logistics Support, p. 33. Headquarters Department of the Army, Washington DC (2007)
2. MIL-STD-1390D, Level of repair analysis, pp. 56–300. Department of Defense of USA, Washington DC (1993)
3. GJB 2961, Level of repair analysis. pp. 20–50. Publishing House of Defense Industry, Beijing (1997)
4. Liu, D.Y., Yang, K., Chen, J.Z.: Research and Development of the Agent Technology. Journal of Software 11(3), 315–321 (2000)
5. Davis, D.N.: Agent-based decision-support framework for water supply Infrastructure rehabilitation and development. Computers, Environment and Urban System 24(3), 173–190 (2000)
6. Yuan, H., Li, Y., Yang, L.: Research on integrated control and supervision of engineering vehicles based on Agent. Journal of Beijing University of Aeronautics and Astronautics 30(8), 723–726 (2004)
7. Jiang, W.J., Xu, Y.S., Sun, X.M., Xu, Y.H.: Fault diagnosis model distributed intelligence based on multi_agent system. Control Theory & Applications 21(6), 945–950 (2004)

8. Ren, M.L., Yang, S.L., Zhu, W.D.: Intelligent decision support system: State of art and challenges. Journal of Systems Engineering 17(5), 30–44 (2002)
9. Guan, X., Cheng, M., Liu, B.: Agent-oriented software engineering. Journal of System Science and System Engineering 10(1), 44–49 (2001)
10. Yan, J.H., Zhang, H.C., Jing, Y.Z.: Implementation of Multi Agent on dynamic task scheduling for distributed measurement and control system. Computer Engineering and Application 45(2), 219–222 (2009)

The Framework of an Intelligent Battlefield Damage Assessment System Based on Multi-Agent System

Xiangkai Liu, Huimei Li, Jian Zhang, Jianing Wang, and Wenhua Xing

Department of Automobile, Academy of Military Transportation, Tianjin, China

Abstract. The basic content and procedure of Battlefield Damage Assessment(BDA) has been discussed and researched. The structure, the disposal strategy, the cooperation between agents, and the data flow of an Intelligent Battlefield Damage Assessment System(IBDAS) based on multi-agent system(MAS) has been studied. This system can solve the difficulty of BDA under the complicated and changing battlefield environment, and lay the theoretical foundation for the realization of a practical IBDAS based on multi-agent system.

Keywords: Battlefield damage assessment, Repair, Multi-agent system, Intelligent.

1 Introduction

The purpose of BDAR (battlefield damage assessment and repair) is to return disabled equipment rapidly to combat or to enable the equipment to self recover [1]. The repair works may be temporary and may not restore full performance capability. For example, the repair environment is complicated, the repair time may be strictly limited, the repair goal and method are not unique, the required repair personnel and materiel are uncertain.

The BDA (battlefield damage assessment) is a procedure to quickly identify damage area and extent, whether it can be field repaired, predict the repair time, and the expected capacity after repair, the repair position, method and process, and the support resources needed [2]. It is a key factor for the accomplishment of the BDAR tasks. Under the complicated and changing battlefield environment, there are many difficult problems that BDA faces, such as how to quickly identify the damage area and the influence of the damage on current task, which emergency method should be applied, whether local repair or rear fix is needed, and if local repair is needed, what emergency repair method should be applied, how many resources are needed, etc,.

The agent theory and technique originated from the Distributed Artificial Intelligence (DAI), and in the 1980s agent theory separated from DAI. It has been combined with many other technologies and obtained many applications different from DAI [3],[4],[5]. Now, multi-agent systems have been applied widely in decision support systems [6],[7],[8].

There are some disadvantages in current Battlefield Damage Assessment Systems(BDAS), for example, most systems are some types of decision support systems based on the data model and conventional numerical methods, which can not simulate complex situation under actual battlefield conditions. With the development of the Artificial Intelligence Technology, knowledge disposal method and knowledge base system have been brought into BDAS, and the Intelligent Battlefield Damage Assessment System (IBDAS) has been a new trend of BDAS[9],[10].

2 The Content and Procedure of BDA

BDA should solve the problems quickly as the following [11]:

a) The damage location and extent, the effect to current task which the equipment is carrying out.
b) Whether local repair or rear fix is needed.
c) The repair sequences.
d) The repair level.
e) Repair method and repair procedure.
f) Support resources required.
g) The status and operating limit of the equipment after repair.

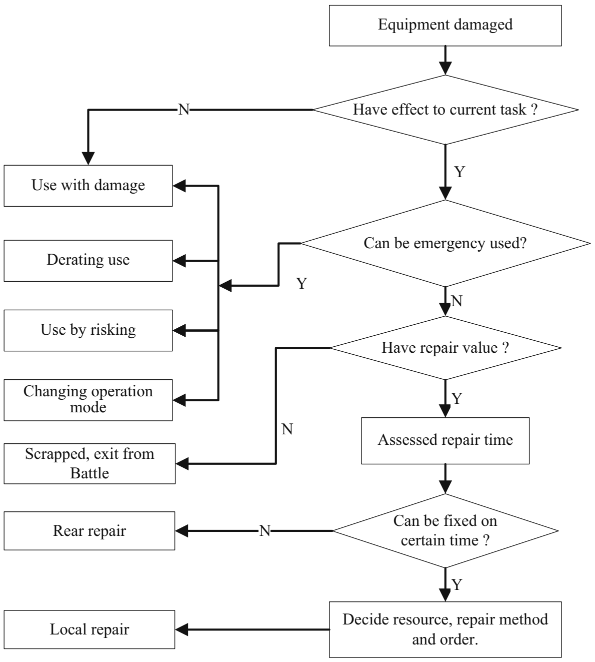


Fig. 1. The general procedure of BDA

In Fig.1, the procedure of BDA is a logic decision process. After the logic decision, the damaged equipment will be charged as the following:

a) Continue to use, including use with damage, de-rating use, use by risking, and changing operation mode.
b) Local repair, repair by maintenance team in the front.
c) Rare repair, the equipment will exit from the battlefield temperately, may be dragged to the rear areas, and repaired by the rear maintenance groups or factories.
d) Scrapped, after necessary charge of it, the equipment will be abandoned and exit from the battlefield permanently.

3 IBDAS Based on Multi-Agent System

3.1 The Framework of IBDAS Based on MAS

In an IBDAS based on MAS, there are many different agents, each agent has its own purpose and function, and many agents cooperate to solve complex practical problems. The framework of IBDAS based on MAS has been established, see Fig. 2.

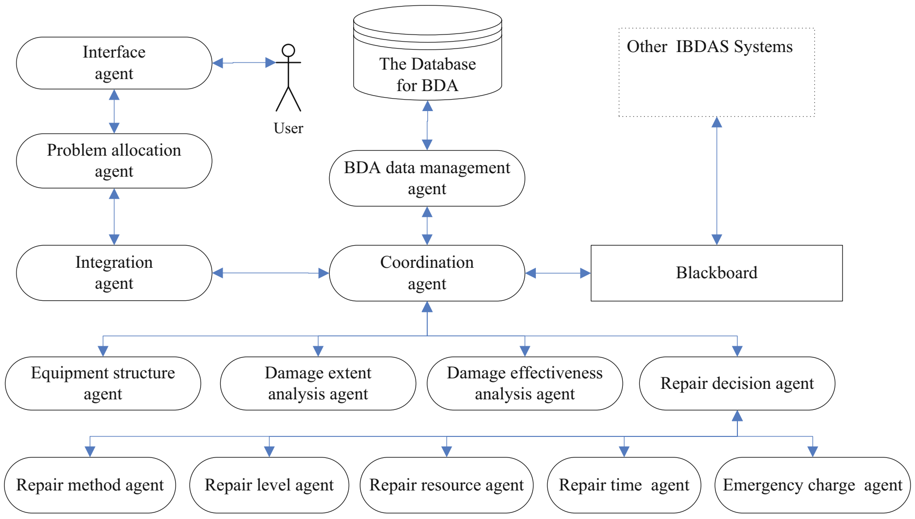


Fig. 2. The structure of IBDAS based on MAS

It is a layered and dynamic model and includes the information-sharing and the common task disposal models. Obviously, each question can be resolved by each problem resolve agent, and all agents can be managed by management agent. They can accomplish each sub-task through information sharing and cooperation.

a) The interface agent. Can communicate with the user, the user in anywhere can input the damage status of equipments, and get the assessment results.
b) The problem allocation agent. Distributes the complex problem to several sub-problems which can be comprehended and finished by each agent.

c) The coordination agent. Takes charge of the information exchanging and communication, and is also monitors the actions of each agent and distributes the sub-problem to each problem agent.
d) The blackboard. Is a data-store area of local agents in which the agents make the communication and talk with each other. The blackboard stores the information the agents shared and passes the success case to the data-management agent, so the data-management can decide whether or not to add the case to the BDA database system.
e) The BDA database system. Is a data storage system which interacts with the expert and the data management agent. The basic function of the BDA database system is to store and collect the data and give the other agent data backup.
f) The integration agent: Is in charge of the integration and the judgment of the result from the problem allocation agent and coordination agent, and gives the final result.
g) Equipment structure agent. Manages the hierarchy structure data and provide basic data of equipment to other agents.
h) Damage extent agent. Dedicates the extent and severity of damaged equipment.
i) Damage effectiveness agent. Determines the capacity and task effectiveness to the equipment.
j) Repair decision agent. Is the core agent in this system, it can determine the repair level and method, evaluate the repair time, predict the repair resources, etc.

3.2 The Disposal Strategy of IBDAS

The disposal strategy of IBDAS can be expressed as the followings:

a) The beginning of disposal. The Interface Agent accepts the task through man-machine interactive.
b) Problem breakdown. The Problem Allocation Agent breakdown the complex damage to sub problems.
c) Dispatch sub problem. The Coordination Agent dispatch sub problem to corresponding problem Agent. If a sub problem can not be solved efficiently, the Coordination Agent will transfer it to the blackboard and delivering it to Problem Allocation Agent, then return to (b).
d) Get the results of sub problem. Damage analysis agent and repair decision agent make out the results. The results may include the damage extent, the capacity remained, the damage level, the repair position, the repair method, the estimated repair time, repair personnel and resources required, etc.
e) Get the results of problem. When all of the sub problem have been solved, The Integration Agent integrate problem of each sub-problem, forming all results and output to the user through the interface Agent.

In the process of problem solving, if the data of damage and repair analysis agent can not satisfy the requirement. We can obtain data from Battlefield Damage database through BDA Data-Management Agent. If there is a new success case, we can also add it to the Battlefield Damage database through Data-Management Agent.

3.3 The Cooperation between Agents

In this system, each agent is an autonomous and independent module. During the problem solving process, each agent can work by its own objectives, knowledge and ability. Some contradictions and conflicts will occur in this system. So the cooperation between each agent is a key problem that the system must face.

Now, there are many methods to solve cooperation problem, such as alliance solving method based on cooperative game theory, BDI model, symbolic logic method, neural network method and algebraic model method [12].

In order to improve the efficiency of BDAS and solve complex social interaction between agents, algebraic model method will be applied to solve the cooperation problem between agents.

3.4 The Data Flow of the System

In the process of damage assessment, each repair unit can input equipment damage data, and make the final decision. The basic data flow of system is as fig. 3.

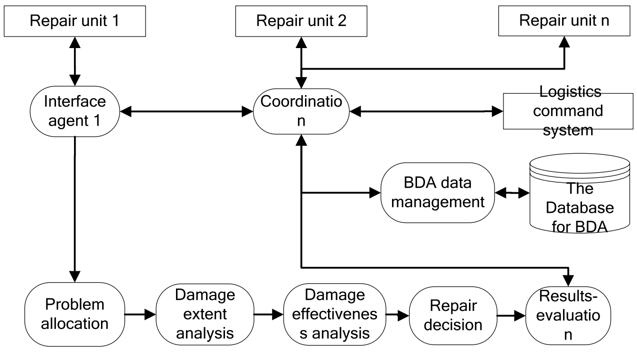


Fig. 3. The basic data flow of system

In this system, the data sources are mainly from three areas.

a) The first data source is the damage data from current repair unit. The damage data include the damage parts, the damage mode, and the damage phenomenon, etc. Typically, the damage data is complicated and consists of several damaged parts, and each damaged part may have multiple damage modes. The personnel (usually a assessor) of repair units can input the damage data to the interface agent through man-machine interface. The problem allocation agent can breakdown the damage data to simple data. Through a series of damage analysis, repair decision, and result evaluation, the preliminary assessment results and the evaluation of the results are passed to the coordinate agent. If the assessment result is feasible, the damage assessment process is over. If the damage assessment result is not feasible, the repair unit can take two ways to solve the problem: 1) Seek a compromise scheme through the interaction with the system, which often require that assessment persons have some expertise. 2) Seek help to other units.

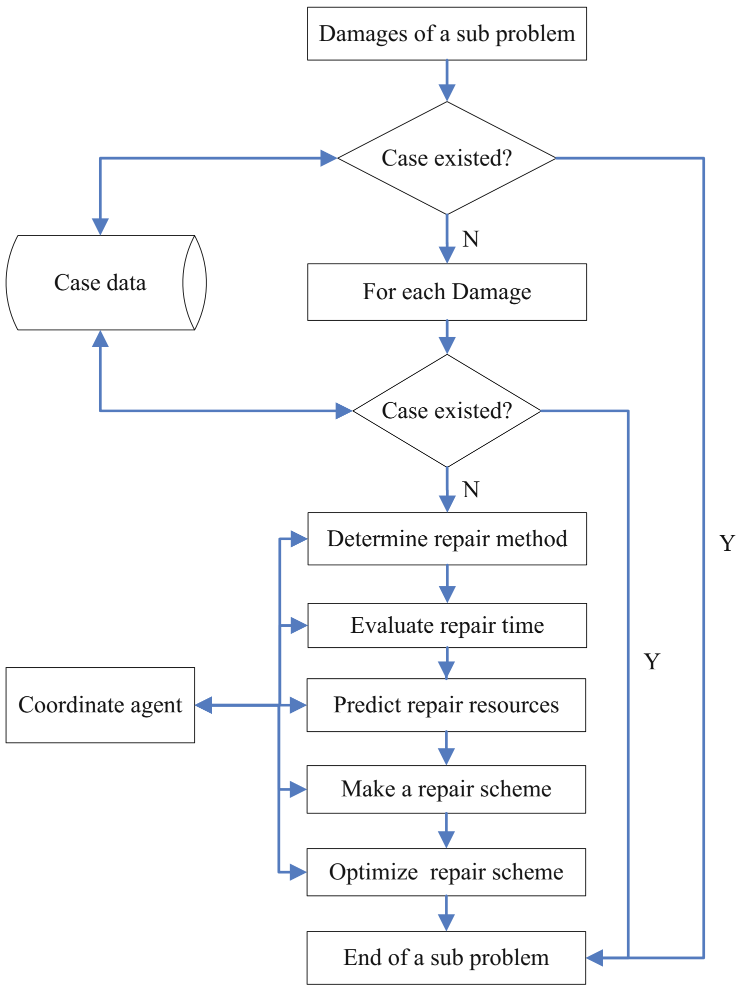


Fig. 4. The basic process flow of repair decision

b) The second data source is from other repair units. They can provide problem solving schemes for current unit.
c) Another important data source is from equipment support system, these data may be battlefield situation, reserves, personnel, support task and other mission information.

There are several main procedures in this system. The following is the simple descriptions of them:

a) Problem allocation. Transfer the intention of user to recognizable information for system, and breakdown the complex problem to sub problems. Generally, the problem is how to get an optimal repair scheme for the damaged equipment, which multi systems/components/parts may be damaged. Suppose there are n damages, the problem allocation model will breakdown them to m(m≤n) sub problems by the knowledge. A sub problem should be solved by damage and repair agent easily, and each sub problem will consist of one or more damages. If there is a successful case for a problem or sub problem, the problem allocation, damage and repair analysis will be rather simple, the final or intermediate result will be got directly from the case database.

b) Damage extent analysis. Identify the equipment's damage extent and level by the damage parts and damage modes. It will get the damage extent of every item, and determine the whole equipment's damage extent through integrated analysis. The damage extent of each item can be defined by specialist, and can be input into BDA database as some kind of knowledge.
c) Damage effectiveness analysis. Determine the equipment's capacity remained and whether the current task can be accomplished. This analysis can be carried out by a series of techniques, such as the basic capacity and capacity degraded status(DS) analysis, capacity and task analysis(C&TA), the basic item(BI) analysis, the mathematical damage tree(MDT) analysis, damage mode, effectiveness analysis (DMEA), etc.
d) Repair decision. Is the most important and complicated module in this system. The basic process flow is as fig. 4.

4 Conclusions

The suggested system can make up the weaknesses of current battlefield damage assessment system. It is more flexible, and can timely resolve problems through multi-expert cooperation and satisfy the requirement of BDA under complicated and changing battlefield environment. Also the accomplishment of it will need lots of hard works, the framework, the software architecture and other method or model will provide a good basis for the development of a practical system.

References

1. Army Regulation 700–127, Integrated Logistics Support, p. 98. Headquarters Department of the Army, Washington DC (2007)
2. GJB451A, Reliability, maintainability and supportability terms. p. 8. Publishing House of Defense Industry, Beijing (2005)
3. Liu, D.Y., Yang, K., Chen, J.Z.: Research and Development of the Agent Technology. Journal of Software 11(3), 315–321 (2000)
4. Guan, X., Cheng, M., Liu, B.: Agent-oriented software engineering. Journal of System Science and System Engineering 10(1), 44–49 (2001)
5. Qiao, B., Zhu, J.Y.: Agent-Based Intelligent Manufacturing Systems: A State-of-the-Art Survey. Journal of Nanjing University of Aeronautics & Astronautics 33(1), 1–7 (2001)
6. Davis, D.N.: Agent-based decision-support framework for water supply Infrastructure rehabilitation and development. Computers Environment and Urban System 24(3), 173–190 (2000)
7. Ren, M.L., Yang, S.L., Zhu, W.D.: Intelligent decision support system: State of art and challenges. Journal of Systems Engineering 17(5), 430–440 (2002)

8. Yuan, H.B., Li, Y.H., Yang, L.M.: Research on integrated control and supervision of engineering vehicles based on Agent. Journal of Beijing University of Aeronautics and Astronautics 30(8), 723–726 (2004)
9. Jiang, W.J., Xu, Y.S., Sun, X.M., Xu, Y.H.: Fault diagnosis model distributed intelligence based on multi-agent system. Control Theory & Applications 21(6), 945–950 (2004)
10. Liu, X.K., Dai, W.J., Tang, Y.F., Wang, J.N.: A Study of an IntelligentBattlefield Damage Assessment System Based on Multi-agent System. International Journal of Plant Engineering and management 13(1), 41–46 (2008)
11. Shi, Q., Mi, S.S., Wang, G.Y., Hu, Q.W.: The battlefield damage theory and technique for materiel, p. 266. Publishing House of Defense Industry, Beijing (2007)
12. Shuai, D.X., Jing, G.U.: A New Algebraic Modeling for Distributed Problem-Solving of Multi-Agent Systems(Part II): Colony intelligence and social dynamics. Chinese Journal of Computers 25(2), 138–147 (2002)

Adaptive System of Heterogeneous Multi-agent Investors in an Artificial Evolutionary Double Auction Market

Chi Xu[1,2], Xiaoyu Zhao[1], and Zheru Chi[1]

[1] Dept. of EIE, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
[2] Dept. of CST, The Northern China University of Technology
chi.xu@polyu.edu.hk

Abstract. In this paper, an adaptive system is proposed which attempts to combine together the approaches of studies of historical data and researches of multi-agent artificial market by evolving a double auction market model with diversity of different traders. The purpose of this research is to construct an artificial market which is more close to realistic one and more practical for future researches. The model with heterogeneous agents and the environment with which agents and market interact is complicated but controllable by data mining the optimal proportion of the different agents at the input to the market that generates an output which can fit historical data curve. The simulation results suggest that the system performance is close to the expecting values in the testing with adequate training in advance.

1 Introduction

Computational economics researchers use computational tools both for computational economic modeling and for the computational solution of analytically and statistically formulated economic problems. Agent-based computational economics (ACE) is the computational study of economic processes modeled as dynamic systems of interacting agents. ACE seeks to break down aggregate macroeconomic relationships into microeconomic decisions of individual agents, defines the set of agent that make up the economy, and specifies the types of interactions individual agents can have with each other or with the market as a whole. The models built by ACE approach specify the strategies instead of defining preferences of the agents [1]. Large numbers of individual agents who are very heterogenous can be simulated. The aggregate, macroeconomic relationships that arise from those individual actions can be studied.

The researches in financial market simulation area split into two branches: one is to forecast the market outcome in the future from historical data without considering the traders activities, and the other branch carefully investigates the behaviors of traders, constrains the market rules, and generates market dynamics of volatile of price and trading volumes. The weakness of these two branches is obvious: the first one is always vulnerable to Lucas critique which declares that it is naive to try to predict the effects of a change in economic policy entirely on the basis of relationships observed in historical data, especially highly aggregated historical data. The second one is easily affected by a butterfly effect of the simple errors in processes which accumulate to carry out an unexpected market performance [3]. The purpose of the project is attempting to combine the two branches together for forecasting the outcome of market with carefully implemented artificial market model. In our project, the agent-based computational economic model is chosen to implement the models of trader and market, for the ability to study local interactions between individual agents. For overcoming the weaknesses such as exaggeration of errors carried out in individual ACE model decision-making, some machine learning methods are applied to the model trying to optimize a choice from different strategies.

2 Methodology

2.1 Agent

Although in the real market, there exists variety of different investors who keep common belief to gain money from the market, in our project, only three kinds of agents are invoked to represent all kinds of market participants, long term investors, inside traders and momentum day traders.

Each agent should have four parameters to show whether the trading order is to buy, to sell, or to hold. Since the agents are working in a double auction market, the orders should be classified into market order, which meets the market trading price and the transaction is executed immediately, and limit order, which is placed into the order book with a limit offering price for sell or buy. The quantity of security that the agent would transact should also be informed to the system, and the offering price by the agent for a particular security should also be recorded in the order book. The demand from each agent can be defined as the vector of desire $\boldsymbol{d}$:

$$\boldsymbol{d} = (side_t, type_t, amount_t, price_t) \tag{1}$$

in which $side_t$ is the decision of an agent whether to buy or to sell or to hold at certain time point t, $type_t$ indicates whether the order is a market order or a limit one, $amound_t$ is the intended purchase quantity, and $price_t$ is the biding price.

The long term investors apply buy and hold strategy in the market. They would like to decline each day price volatility in the market and obtain a good return by gaining from the the up moving trend of the market price curve, which is supported by the efficient market hypothesis (EMH) of macroeconomics theory [2]. On the contrary, the day traders try to short on the curve peaks and buy at the valleys for making more money in greater price volatilities. In the reality, there are also inside traders who are not big in amount in the market but their purchase and sell off carry out huge effects to market price trend.

In previous analysis of the individual price predictions, the researchers revealed that the participants in the market were able to coordinate on a common prediction strategy, and the estimation of the individual predictions showed that participants had a tendency to use simple, linear forecasting rules [1]. The model of long term investor can use moving average as the trading strategy, by evaluating market price in a long period of time frame, such that the volatility of the price can be smoothed and only the moving trend remains. The autoregressive (AR) is applied, while a real stock market index is set up as the given expected time series value, and the conditional sum-of-squared errors are used to optimize the output by fitting the expected time series curve.

$$Y_t = constant + \sum_{i=1}^{p} \omega_i Y_{t-i} + e_t \tag{2}$$

in which, p is the order of the autoregressive time series, ω_i is weight of historical data, and e_t is the sum-of-squared error between actual output and the expected value.

The inside traders are those agents who really have the acknowledgement of market moving trend, so their prediction of the market price should be very accurate. Usually, such traders are difficult to model, since a perfect forecast to a market is a task impossible. In our project, the trader is implemented as similar as to the model of long term investor, while the genetic algorithm (GA) is applied to choose a better combination of weights so that the agent can generate a curve perfectly matching the expected time series. The fitness function for GA selection can be described as:

$$\hat{e} = \min\left(\frac{1}{p}\sum_{t=1}^{p} e_t^2\right). \tag{3}$$

In the system, the momentum day traders are not rational agents, and their trading strategies can be simulated by a random decision making scheme. The model for such trader is build as so called zero-intelligence trader, and the price forecast made are random numbers with a ceiled limit close to max value in the historical data.

2.2 Market

Some assumptions are made for setting up the market that individual decisions depend upon expectations or beliefs about future developments, so the financial market is an expectations feedback system, where the market history shapes individual expectations, and the individual expectations determine current aggregate market behavior. In an efficient multi-agent double auction market (MDA), the buyers and sellers submit to an order book the bids or asks about how many security they are purchasing or selling and at which prices, i.e. limit prices. According to the information, the exchange mechanism in the market compares the prices from each side in the order book, and tries to find out matches between the buy side and the sell side. The optimization should be approached by determining the amount that each agent should purchase or sell and at what price to maximize the total profit of the market [2]. In a MDA market with m buyers and n sellers, each buyer i wants to purchase Z_i unit items and each seller j has Y_j unit items to sell. Assuming both Z_i and Y_j are known to every agent. The limit prices, which are private, for buyer i and seller j are b_i and s_j. Also assuming that the limit price for each agent is static. Let q_{ij} denote the amount buyer i buys from seller j. If all information is public, the maximum total market value can be obtained by solving the following linear programming problem:

$$\max \sum_{i=1}^{m} \sum_{j=1}^{n} q_{ij}(b_i - s_j), \tag{4}$$

where

$$\sum_{i=1}^{m} q_{ij} \leq Y_j \ \forall j \tag{5}$$

and

$$\sum_{j=1}^{n} \leq Z_i \ \forall i \tag{6}$$

with

$$q_{ij} \geq 0 \text{ for every } i,j. \tag{7}$$

The market has a mechanism to observe the prices according to the desire information from agents at each time t. The prices represent the most informative prices given all information available at that particular time point. In the market only single security is traded, and the agents generate trading signals, bid or ask or hold, to the market. No crossing orders among traders are allowed in the market, and the security traded liquidates into cash immediately after the transaction which means no inventory risk exists for each trader in the market. The market executes all orders, and after execution, the market gives back knowledge of past transactions to the agents, and adjusts the proportion of different agents. The market is continuous in which the orders are executed at the moment when they arrive.

Besides the functionality of regulating the market, another important role to play is to generate the security price of the artificial market by comparing the limit order prices from different traders.

$$P_t^m = \begin{cases} \max(p_t^b) \text{ or } \min(p_t^s) & \text{if match with buy and sell in limit order book} \\ \frac{1}{2}\left(\max(p_t^b) + \min(p_t^s)\right) & \text{elsewise} \end{cases} \tag{8}$$

in which, p_t^b is the buying price in limit order book, and p_t^s is the selling price in the limit order book, at particular time point t.

The market framework scheme is shown in figure 2. Engine gives speak to agents who are allowed to speak at current time. Agents acquire information about current state of the market (best offers, current stock price, demand or

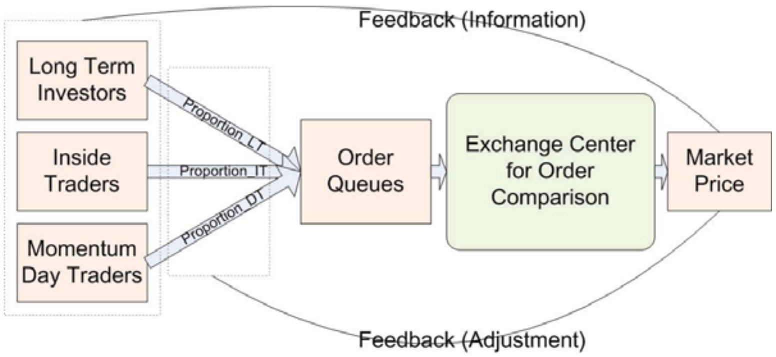


Fig. 1. Market structure

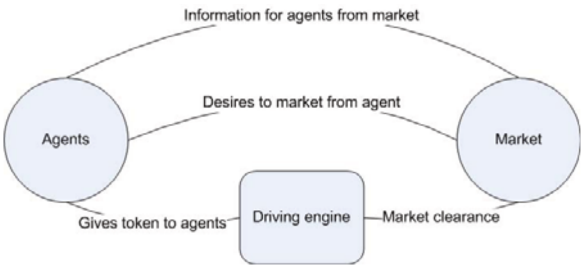


Fig. 2. Market environment

supply imbalance, etc.) before making the decision, as well as the exogenous information from the world. Agents emit to the market a desire which is stored in an accumulator who keeps tracking of agents desires. Market informs emitter about its validity when a desire is received. Engine notifies the market to take the agent desire into account when it gives speak to the particular agent. It means "enter in a clearing phase and compute a new price". Engine gives the possibility to the world to update itself.

3 Simulation and Results

In our proposed model, the inside traders is trained to forecast accurately the Hong Kong Hang Seng Index (HSI), while the long term investor is trained to be able to generate the trend of the index. The significance of the existence of the inside traders is to make the artificial market attempt to obtain the level of equilibrium adaptively, so the goal of keeping the market efficient can be achieved no matter how many momentum day traders generate noise like price volatility in the market. The performance of inside trader and long term trader is shown in Figure 3 and 4.

Initially, 10 inside traders, 30 long term investors, and 50 momentum day traders compose the initial population in the double auction market, while

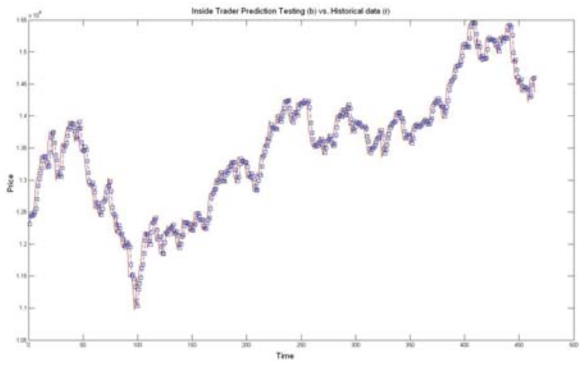


Fig. 3. Inside trader forecast performance

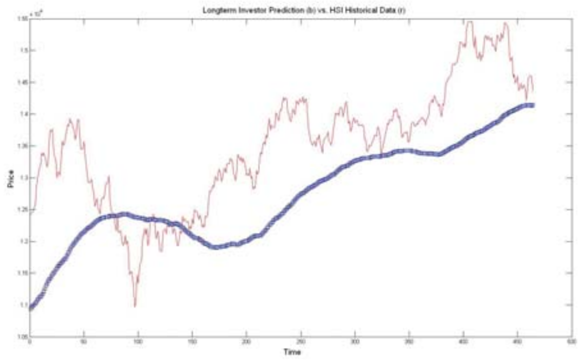


Fig. 4. Long term investor forecast performance

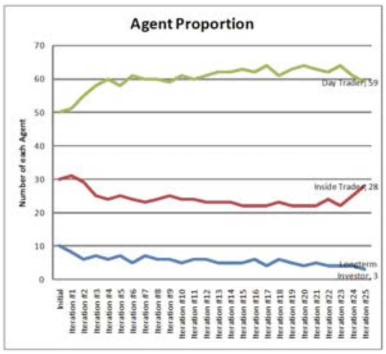


Fig. 5. Proportion of different investor in the market

output price from the market is evaluated with the historical data and back-propagates to control the input proportion of different agents to the system. Figure 5 shows the changing population of the market during 25 iterations of GA optimization.

The changing trend of existence of different traders can indicate that more although the number of day traders is larger than the other two kinds agent, it increases during the simulation, while the other two decreases. It can be explained the fact that when more rational agents, especially more similar type traders in the market, the transaction of trade should be suppressed. On the contrary, when more day traders in the market, the volatility of the market becomes more obvious.

4 Conclusion and Discussion

When people use the artificial market to research on economic phenomena of financial market, although the volatility of market price and trade volumes can be generated, it always exists a problem that how to explain the noise-like volatility from the artificial market. No matter how many features had been added into the artificial market, it always difficult for a researcher to declare what is the distance that the artificial model is from a real market. This paper presents a framework for construction of an adaptive system, which composes of heterogeneous multi-agent investors who trade in an artificial evolutionary double auction market. This framework combines the study of historical data and traders behaviors together to attempt to implement the artificial market more practical and close to a real market. The mechanism for the interactive activities between agents and market environment makes the model controllable, and the historical data curve fitting function prevents the model from butterfly effect of error accumulation. GA is applied for the rational agent models to optimize their forecasts, and for the double auction market to generate rational price output.

The individual agents that constitute an economy do not necessarily move at the same time or in the same direction, and this is the reason why the moves being executed are quite complex with a collection of relatively autonomous entities with no central control but a great deal of interaction. The goal of quantitatively replicating the important features in an actual financial market has been achieved, while the reality of the market can be maintained in high level. The simulation of market generates favorable market price dynamics, and the composition of different agents varies during the simulation. With this proposed system architecture and simulation platform, the typical events in the financial market can be simulated in the future researches.

References

1. Anufriev, M., Hommes, C.: Evolutionary Switching between forecasting Heuristics: An Explanation of an Asset-Pricing Experiment. In: Schredelseker, K., Hauser, F. (eds.) Complexity and Artificial Markets. Springer, Heidelberg (2008)

2. Chan, T.: Artificial Market and Intelligent Agents. Dissertation, MIT (2001)
3. Chen, S.-H.: Software-Agent Designs in Economics: An Interdisciplinary Framework. IEEE Computational Intelligence Magazine, 18–22 (November 2008)
4. Derveeuw, J., Beaufils, B., Mathieu, P., Brandouy, O.: Testing Double Auction as a Component Within a Generic Market Model Architecture. In: Consiglio, A. (ed.) Artificial Markets Modeling, Methods and Applications. Lecture Notes in Economics and Mathematical Systems. Springer, Heidelberg (2007)
5. Hayward, S.: Financial Modeling and Forecasting with an Evolutionary Artificial Neural Network, ch. 1. Financial Modeling and Forcasting (2006)
6. Malik, S.: Artificial Stock Market for Testing Price Prediction Models. In: 2nd IEEE Internaltional Conference on Intelligent Systems (2004)
7. Markellos, R.N., Siriopoulos, C.: Times-series behavior of intradaily data from the Athens Stock Exchange. International Transactions in Operational Research, Res. 9, 619–628 (2002)
8. Wanas, N., Auda, G., Kamel, M.S., Karray, F.: On the optimal number of hidden nodes in a neural network. In: IEEE Canadian Conference on Electrical and Computer Engineering, vol. 2, pp. 918–921 (1998)
9. Xu, C., Chi, Z.: Pattern-Oriented Agent-Based Modeling for Financial Market Simulation. In: Liu, D., Fei, S., Hou, Z.-G., Zhang, H., Sun, C. (eds.) ISNN 2007, Part I. LNCS, vol. 4491, pp. 630–635. Springer, Heidelberg (2007)
10. Xu, C., Cai, Y., Chi, Z.: Stock Investor Behavior Simulation with Hybrid Neural Network and Technical Analysis on Long-term Increases of Hong Kong Hang Seng Index. In: Dynamics of Continuous, Discrete & Impulsive Systems, Series A: Mathematical Analysis, Supplementary Advances in Neural Networks, August 2007, vol. 14(S1), pp. 624–630. Copyright@Watam Press (2007)
11. Zimmermann, H.G., Neuneier, R., Grothmann, R.: Multiagent Modeling of Multiple FX-Markets by Neural Networks. IEEE Transactions on Neural Networks 12(4), 735–743 (2001)

Average Consensus for Directed Networks of Multi-agent with Time-Varying Delay

Tiecheng Zhang and Hui Yu

Institute of Nonlinear and Complex Systems, College of Science,
China Three Gorges University, Yichang 443002, Hubei, China
ztchongctgu@yahoo.cn, yuhui@ctgu.edu.cn

Abstract. The average consensus in directed network of multi-agent with both switching topology and time-varying delay is studied. An orthogonal matrix is introduced to change the initial system into a reduced dimensional system. Based on linear matrix inequalities (LMIs) technique, a sufficient condition about average consensus problem is proposed. A novel form in terms of LMIs is obtained via taking the relationship between the terms in the Newton-Leibniz formula into account. Because some free weighted matrices are employed in the analysis processing and are selected through solving LMIs appropriately, our method is less conservative and more general. Finally, simulation examples are given to demonstrate the effectiveness of the theoretical results.

Keywords: Average consensus, Directed networks, Time-varying delay, LMIs.

1 Introduction

In recent decades, distributed coordinated control of networks of dynamic agents has received a great deal of attention within the control community. That can be attributed to its broad applications in many areas.

A critical problem for coordinated control is to design appropriate protocols and algorithms such that all agents can reach agreement on a certain quantity of interest. Such problem is called consensus problem. In [1], a simple but interesting model of autonomous agents all moving in the plane with the same speed but with different headings was proposed. Simulation results provided in [1] showed that all agents can eventually move in the same direction without centralized coordination. The first paper on a theoretical explanation for this model was provided in [2]. In [3], theoretical results were extended to the case of directed graph. A set-valued Lyapunov approach was used to study consensus problem with undirected time-dependent communication links in [4]. In [5], average consensus problem in network of multi-agent with switching topology was discussed. When network communication was affected by time-delay, the consensus problem was investigated in [5] and [6]. Weighted average consensus in directed networks and undirected networks with time-delay was proposed in

[7]. In [8], multi-vehicle consensus problem with a time-varying reference state was discussed.

Recently, a multitude of works on consensus problem in network of agents were studied using LMIs method in [9], [10], [11] and [12]. In the delay system, the network topology is a key factor in the analysis of stability of multi-agent system. Some issues about network topology in solving average consensus problem had been considered in [11] and [12].

In this paper, we study the average consensus problem for directed networks of multi-agent with both switching topology and time-varying delay. Because the system matrix is singular, the traditional LMI-based control theory is invalid. Compared to [9] and [10], a new method is proposed in our work. It deals with the system models indirectly by introducing an orthogonal matrix. In addition, it does not use the improved inequality to estimate the upper bound of $2a^Tb$, so this reduces the conservatism in the derivation of the stability conditions. Especially, some free weighted matrices are employed to express the influence of the terms in the Newton-Leibniz formula. This is the main advantage of our method, which is less conservative and more general than the existing ones.

2 Problem Statement and Preliminaries

2.1 Algebraic Graph Theory

Let $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{A})$ be a weighted directed graph of order n $(n \geq 2)$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, $\mathcal{A} = [a_{ij}] \in \mathbb{R}^{n \times n}$ is a weighted adjacency matrix and a_{ij} are selected from a finite set of nonnegative elements. The node indexes belong to a finite index set $\mathcal{I} = \{1, 2, \cdots, n\}$. A directed edge from agent i to agent j is denoted by $\varepsilon_{ij} = (v_i, v_j)$, which represents a directed information exchange link from agent i to agent j, that is, agent i can receive or obtain information from agent j. $\varepsilon_{ij} \in \mathcal{E}$ if and only if $a_{ij} > 0$. Moreover, we assume $a_{ii} = 0$ for all $i \in \mathcal{I}$. The set of neighbors of nodes v_i is denoted by $\mathcal{N}_i = \{v_j \in \mathcal{V} : (v_i, v_j) \in \mathcal{E}\}$. A directed path in graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{A})$ is a sequence of ordered edges $(v_{i_1}, v_{i_2}), (v_{i_2}, v_{i_3}), (v_{i_3}, v_{i_4}), \cdots$, where $v_{i_j} \in \mathcal{V}$. Graph $\mathcal{G}$ is called strongly connected if there is a directed path between any tow distinct nodes.

The in-degree and out-degree of node v_i are respectively defined as follows: $d_{\mathtt{in}}(v_i) = \sum\limits_{j=1}^{n} a_{ji}, d_{\mathtt{out}}(v_i) = \sum\limits_{j=1}^{n} a_{ij}$. The degree matrix of graph $\mathcal{G}$ is a diagonal matrix $\mathcal{D} = [d_{ij}]$, where $d_{ij} = 0$ for all $i \neq j$ and $d_{ii} = d_{\mathtt{out}}(v_i)$. The Laplacian matrix associated with the graph is defined as $L = \mathcal{D} - \mathcal{A}$. An important fact of L is that all the row sums of L are zero and thus $\mathbf{1}_n = [1, 1, \cdots, 1]^T \in \mathbb{R}^n$ is an eigenvector of L associated with the eigenvalue $\lambda = 0$.

Definition 1. (Balanced graph [5]). The ith node of a directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{A})$ is balanced if and only if $d_{\mathtt{in}}(v_i) = d_{\mathtt{out}}(v_i)$. A graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{A})$ is called balanced if and only if all of its nodes are balanced.

Definition 2. (Balanced matrix [10]). A square matrix $\mathcal{M} \in \mathbb{R}^{n\times n}$ is said to be a balanced matrix if and only if $\mathbf{1}_n^T \mathcal{M} = 0$ and $\mathcal{M}\mathbf{1}_n = 0$.

Lemma 1. [5]. If the graph $\mathcal{G}$ is strongly connected, then its Laplacian matrix L satisfies: (1) rank(L)= $n-1$; (2) zero is one eigenvalue of L, and $\mathbf{1}_n$ is the corresponding eigenvector, then, $L\mathbf{1}_n = \mathbf{0}$; (3) the rest $n-1$ eigenvalue all have positive real-parts. In particular, if the graph $\mathcal{G}$ is undirected, they are all positive and real.

2.2 Consensus Problems in Networks

Consider a group of n agents with dynamics given by

$$\dot{x_i}(t) = u_i(t), \qquad i \in \mathcal{I} \tag{1}$$

where $x_i \in \mathbb{R}^n$ is the state of the ith agent at time t, and $u_i(t) \in \mathbb{R}$ is the control input (or protocol) at time t.

We say that protocol u_i asymptotically solves the consensus problems, if and only if the states of agents satisfy $\lim\limits_{t\to\infty} \|x_i(t) - x_j(t)\| = 0$, for all $i, j \in \mathcal{I}$. Furthermore, if $\lim\limits_{t\to\infty} x_i(t) = \frac{1}{n}\sum\limits_{i=1}^{n} x_i(0) = \mathtt{Ave}(x(0))$, We say that protocol u_i asymptotically solves the average consensus problem.

In this paper, we are interested in discussing the average consensus problem in network of dynamic agents with both switching topology and time-varying delay, where the information (from v_i to v_j) passes through edge (v_i, v_j) with time-varying delay. To solve such a problem, we use the following protocol:

$$u_i = \sum_{v_j \in \mathcal{N}_i} a_{ij}[x_j(t-\tau(t)) - x_i(t-\tau(t))], \quad i \in \mathcal{I} \tag{2}$$

With (2), (1) can be written in matrix form as:

$$\dot{x_i} = -L_k x(t-\tau(t)), \qquad k = s(t) \tag{3}$$

where $x(t) = (x_1(t), x_2(t), \cdots, x_n(t))^T$, and the map $s(t) : [0, \infty] \longrightarrow \mathcal{I}_\Gamma = \{1, 2, \cdots, N\}$ ($N \in \mathbb{Z}^+$ denotes the total number of all possible directed graphs) is a switching signal that determines the network topology, and $L_k = L(\mathcal{G}_k)$ is the Laplacian matrix of the graph $\mathcal{G}_k$ that belongs to a set $\Gamma = \{\mathcal{G}_k : k \in \mathcal{I}_\Gamma\}$, which is obviously finite due to the weights a_{ij} are selected from a finite set of nonnegative elements.

Lemma 2. (Schur complement [13]). Let M, P, Q be given symmetric matrixes such that $Q > 0$, then $\begin{bmatrix} P & M \\ M^T & -Q \end{bmatrix} < 0 \quad \Leftrightarrow \; P + MQ^{-1}M^T < 0$.

Lemma 3. For any balanced matrix $A \in \mathbb{R}^{n\times n}$, there exists an orthogonal matrix W such that $W^TAW = \begin{bmatrix} \bar{A} & \mathbf{0}_{(n-1)\times 1} \\ \mathbf{0}_{1\times(n-1)} & 0 \end{bmatrix}$, where the last column of matrix W is $\frac{\mathbf{1}_n}{\sqrt{n}}$, $\bar{A} \in \mathbb{R}^{(n-1)\times(n-1)}$.

3 Main Results

In this section, we provide the convergence analysis of the average consensus problem in directed network with switching topology and time-varying delay.

If the communication topology $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{A})$ keeps strongly connected and balanced, we have $\mathbf{1}_n^T L_k = \mathbf{0}$, which imply $\sum_{i=1}^{n} \dot{x}_i = \sum_{i=1}^{n} u_i = 0$. Then, $\alpha = \mathtt{Ave}(x(0))$ is an invariant quantity. Thus we have the decomposed equation $x(t) = \alpha \mathbf{1}_n + \delta(t)$, where $\delta = (\delta_1, \delta_2, \cdots, \delta_n)^T \in \mathbb{R}^n$, satisfies $\sum_i \delta_i(t) = 0$, i.e., $\mathbf{1}_n^T \delta = \mathbf{0}$. Then (3) is equivalent to the following equation:

$$\dot{\delta}(t) = -L_k \delta(t - \tau(t)). \tag{4}$$

Suppose that the directed graph $\mathcal{G}$ with Laplacian matrix L is balanced. By Lemma 3, we have $W^T L_k W = \begin{bmatrix} \bar{L}_k & \mathbf{0}_{(n-1)\times 1} \\ \mathbf{0}_{1\times(n-1)} & 0 \end{bmatrix}$, where $\bar{L}_k \in \mathbb{R}^{(n-1)\times(n-1)}$, and W is defined in Lemma 3.

The matrix W is an orthogonal one, so $\dot{\delta}(t) = -WW^T L_k WW^T \delta(t - \tau(t))$, then $W^T \dot{\delta}(t) = -W^T L_k WW^T \delta(t - \tau(t))$. Let $W^T \delta(t) = [\Delta^T(t), 0]^T$, then (4) can be transformed into the following equation:

$$\dot{\Delta}(t) = -\bar{L}_k \Delta(t - \tau(t)), \tag{5}$$

where $\Delta(t) \in \mathbb{R}^{(n-1)}$, $\bar{L}_k \in \mathbb{R}^{(n-1)\times(n-1)}$.

Lemma 4. If $\lim_{t\to\infty} \|\Delta(t)\| = 0$, then $\lim_{t\to\infty} \|\delta(t)\| = 0$.

Proof. From $W^T \delta(t) = [\Delta^T(t), 0]^T$, we have $\delta(t) = W[\Delta^T(t), 0]^T$. Therefore, when $\lim_{t\to\infty} \|\Delta(t)\| = 0$, we have $\lim_{t\to\infty} \|\delta(t)\| = 0$. This completes the proof.

In the following section, we will discuss the convergence of dynamical system (5), that is $\lim_{t\to\infty} \Delta(t) = \mathbf{0}$.

Theorem 1. Consider a directed network of multi-agent with both switching topology and time-varying delay $\tau(t)$ satisfying $\tau(t) \le d$ and $\dot{\tau}(t) \le h < 1$. Suppose that the communication topology $\mathcal{G}_k (k \in \mathcal{I}_\Gamma)$ keeps strongly connected and balanced. Protocol (1) asymptotically solves the average consensus problem, if there exists positive definite matrices $P, Q, R \in \mathbb{R}^{(n-1)\times(n-1)}$, positive semi-definite matrix $X = \begin{bmatrix} X_{11} & X_{12} \\ X_{12}^T & X_{22} \end{bmatrix} \ge 0$, and appropriate dimensions free weighted matrices H, U, such that the following LMIs are true:

$$\Xi = \begin{bmatrix} \Phi_{11} & \Phi_{12} & 0 \\ \Phi_{12}^T & \Phi_{22} & -d\bar{L}_k^T R \\ 0 & -dR^T \bar{L}_k & -dR \end{bmatrix} < 0, \quad \Theta = \begin{bmatrix} X_{11} & X_{12} & H \\ X_{12}^T & X_{22} & U \\ H^T & U^T & R \end{bmatrix} \ge 0, \tag{6}$$

where $\Phi_{11} = H + H^T + Q + dX_{11}$, $\Phi_{12} = -P\bar{L}_k - H + U^T + dX_{12}$, $\Phi_{22} = -U - U^T - (1-h)Q + dX_{22}$.

Proof. Define a common Lyapunov-Krasovskii function for system (5) as follows:

$$V(t) = \varDelta^T(t)P\varDelta(t) + \int_{t-\tau(t)}^{t} \varDelta^T(s)Q\varDelta(s)ds + \int_{-d}^{0}\int_{t+\theta}^{t} \dot{\varDelta}^T(s)R\dot{\varDelta}(s)dsd\theta,$$

where P, Q and $R \in \mathbb{R}^{(n-1)\times(n-1)}$ are positive definite matrices. Along the trajectory of system (5), we have

$$\begin{aligned}\dot{V}(t) = &-2\varDelta^T(t)P\bar{L}_k\varDelta(t-\tau(t)) - (1-\dot{\tau}(t))\varDelta^T(t-\tau(t))Q\varDelta(t-\tau(t))\\ &+\varDelta^T(t)Q\varDelta(t) + d\varDelta^T(t-\tau(t))\bar{L}_k^T R\bar{L}_k\varDelta(t-\tau(t)) - \textstyle\int_{t-d}^{t} \dot{\varDelta}^T(s)R\dot{\varDelta}(s)ds\\ \leq &-2\varDelta^T(t)P\bar{L}_k\varDelta(t-\tau(t)) - (1-h)\varDelta^T(t-\tau(t))Q\varDelta(t-\tau(t))\\ &+ \varDelta^T(t)Q\varDelta(t) + d\varDelta^T(t-\tau(t))\bar{L}_k^T R\bar{L}_k\varDelta(t-\tau(t)) - \textstyle\int_{t-\tau(t)}^{t} \dot{\varDelta}^T(s)R\dot{\varDelta}(s)ds.\end{aligned}$$

By Newton-Leibniz formula $\varDelta(t-\tau(t)) = \varDelta(t) - \int_{t-\tau(t)}^{t}\dot{\varDelta}(s)ds$, then $\varUpsilon = \varDelta(t) - \int_{t-\tau(t)}^{t}\dot{\varDelta}(s)ds - \varDelta(t-\tau(t)) = 0$. So we have

$$2[x^T(t)H + x^T(t-\tau(t))U]\varUpsilon = 0, \tag{7}$$

where the free weighted matrices H and U are appropriately dimensioned matrices, which indicate the relationship betwteen the terms in Newton-Leibniz and can easily determined by solving LMIs. Furthermore, for any positive semi-definite matrix $X \geq 0$, we have

$$d\eta^T(t)X\eta(t) - \int_{t-\tau(t)}^{t} \eta^T(t)X\eta(t)ds \geq 0, \tag{8}$$

where $\eta(t) = [x^T(t), x^T(t-\tau(t)]$. Therefore, using (7) and (8), we have

$$\begin{aligned}&\dot{V} = \dot{V}_1 + \dot{V}_2 + \dot{V}_3 \leq \dot{V}_1 + \dot{V}_2 + \dot{V}_3 + 2[x^T(t)H + x^T(t-\tau(t))U]\varUpsilon\\ &+d\eta^T(t)X\eta(t) - \textstyle\int_{t-\tau(t)}^{t}\eta^T(t)X\eta(t)ds := \eta^T(t)\varLambda\eta(t) - \int_{t-\tau(t)}^{t}\eta^T(t,s)\varTheta\eta(t,s)ds,\end{aligned}$$

where $\eta(t,s) = [x^T(t), x^T(t-\tau(t)), \dot{x}^T(s)]^T$, $\varLambda = \begin{bmatrix}\varPhi_{11} & \varPhi_{12}\\ \varPhi_{12}^T & \varPhi_{22} + d\bar{L}_k^T R\bar{L}_k\end{bmatrix}$, $\eta(t)$ and $\varTheta$ are defined in (8) and (6), respectively. Then, a sufficient condition for $\dot{V}(t) < 0$ is $\varLambda < 0$ and $\varTheta \geq 0$ for any $\eta(t) \neq 0$.

Therefore, from Lemma 2 and Lemma 4, average consensus can be achieved if the matrix inequalities (6) hold. This completes the proof.

When the information of $\dot{\tau}(t)$ is unknown, i.e., h is unknown, we have the following theorem.

Theorem 2. Consider a directed network of multi-agent with both switching topology and time-varying delay $\tau(t)$ satisfying $\tau(t) \leq d$, and $\dot{\tau}(t)$ is unknown. Suppose that the communication topology $\mathcal{G}_k(k \in \mathcal{I}_\varGamma)$ keeps strongly connected and balanced. Protocol (1) asymptotically solves the average consensus problem, if there exists positive definite matrices $P, R \in \mathbb{R}^{(n-1)\times(n-1)}$, positive semi-definite matrix $X = \begin{bmatrix}X_{11} & X_{12}\\ X_{12}^T & X_{22}\end{bmatrix} \geq 0$, and appropriate dimensions free weighted matrices H, U, such that the following LMIs satisfy:

$$\Omega = \begin{bmatrix} \Psi_{11} & \Psi_{12} & 0 \\ \Psi_{12}^T & \Psi_{22} & -d\bar{L}_k^T R \\ 0 & -dR^T\bar{L}_k & -dR \end{bmatrix} < 0, \quad \Theta = \begin{bmatrix} X_{11} & X_{12} & H \\ X_{12}^T & X_{22} & U \\ H^T & U^T & R \end{bmatrix} \geq 0, \tag{9}$$

where $\Psi_{11} = H + H^T + dX_{11}$, $\Psi_{12} = -P\bar{L}_k - H + U^T + dX_{12}$, $\Psi_{22} = -U - U^T + dX_{22}$.

Proof. The proof of the theorem 2 is similar to that of theorem 1, therefore, it is omitted here.

4 Simulation

Consider a group of 10 agents labeled 1 through 10. Fig. 1 shows four examples of directed graph, which are strongly connected and balanced, and the corresponding adjacency matrices are limited to $0-1$ matrices. A finite automation with set of states $\{G_1, G_2, G_3, G_4\}$ is shown in Fig. 2. According to the state machine in Fig. 2, it starts at the discrete state G_1 and switches to the next state every simulation time step.

Let the orthogonal matrix

$$W = \begin{bmatrix} 0.0275 & -0.0303 & 0.3594 & -0.1500 & -0.4082 & 0.7071 & -0.0809 & 0.1749 & -0.2062 & 0.3162 \\ 0.0275 & -0.0303 & 0.3594 & -0.1500 & -0.4082 & 0.7071 & -0.0809 & 0.1749 & -0.2062 & 0.3162 \\ 0.0275 & -0.0303 & 0.3594 & -0.1500 & -0.8165 & 0.0000 & -0.0809 & 0.1749 & -0.2062 & 0.3162 \\ 0.0594 & -0.0294 & 0.3782 & -0.1390 & -0.0000 & 0.0000 & -0.1310 & 0.6023 & -0.5946 & 0.3162 \\ 0.0594 & -0.0294 & -0.2404 & -0.1390 & -0.0000 & 0.0000 & -0.1310 & 0.6694 & -0.5946 & 0.3162 \\ -0.1323 & -0.5742 & 0.2482 & -0.5742 & -0.0000 & 0.0000 & -0.2061 & -0.1207 & -0.3231 & 0.3162 \\ 0.4914 & 0.6506 & -0.2653 & -0.2557 & -0.0000 & 0.0000 & -0.1511 & -0.1290 & -0.2450 & 0.3162 \\ -0.8162 & 0.3793 & -0.2234 & -0.1447 & -0.0000 & 0.0000 & -0.0820 & -0.1087 & -0.0230 & 0.3162 \\ 0.2559 & -0.3149 & -0.4793 & -0.6714 & -0.0000 & 0.0000 & -0.0090 & -0.2331 & -0.0206 & 0.3162 \\ 0.0000 & 0.0000 & -0.0000 & -0.1615 & -0.0000 & 0.0000 & -0.9348 & -0.0000 & -0.0000 & 0.3162 \end{bmatrix},$$

where the last column of matrix W is $\frac{\mathbf{1}_{10}}{\sqrt{10}}$. According to Theorem 1 and Theorem 2, simulations can be conducted in the following cases:

(1) when $h = 0$, i.e., $\tau(t) = 0$, and the time delay is constant, the maximal allowable time-delay $d \leq 0.32$;

(2) when $h = 0.3$, the maximal allowable time-delay $d \leq 0.31$;

(3) when h is unknown, we can obtain the maximal allowable time-delay $d \leq 0.29$.

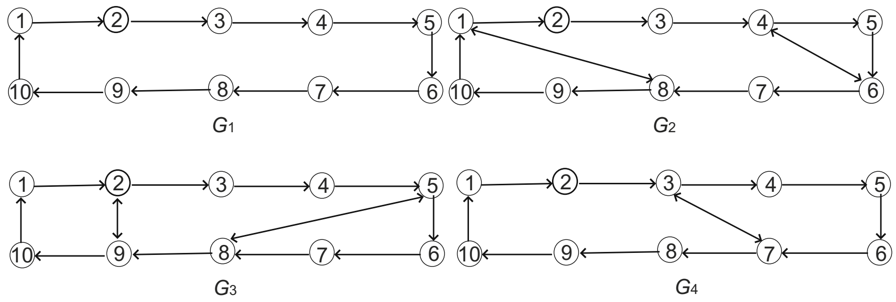


Fig. 1. Examples of strongly connected and balanced directed graph

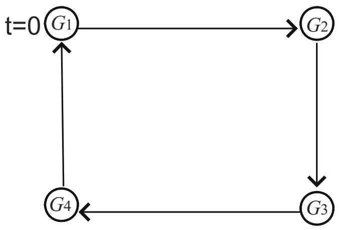


Fig. 2. A finite automation with four states

Fig. 3 shows the corresponding error system with switching topology converges zero asymptotically when h is know. when h is unknown, we can gain the corresponding simulation figure, too. The corresponding feasible solution P, Q, R, X, H and U can be obtained by employing the LMI Toolbox in Matlab.

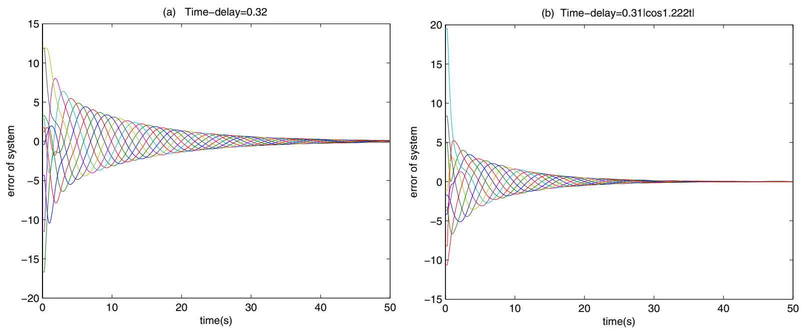


Fig. 3. Error system with switching topology converges zero asymptotically. (a) constant time-delay $\tau = 0.32s$; (b) time-varying delay $\tau(t) = 0.31|\cos 1.222t|s$.

Remark 1. For the examples of Lin et al. [10], the maximal allowable upper bounds on the time-delay can be easily gained using our method. For comparison, the simulation results are listed in the following Table 1.

Table 1. A Comparison with Lin et al. [10] with regard to the maximal allowable upper bounds of time-delay

methods	Lin et al. [10]	our results
$d(h = 0)$	0.21	0.31
$d(h = 0.3)$	0.15	0.29
$d(h\ is\ unknown)$	-	0.27

5 Conclusion

This paper addresses an average consensus problem of multi-agent systems. Directed switching network topology and time-varying communication delay are considered in this paper. By utilizing LMIs techniques and taking the relationship between the terms in the Newton-Leibniz formula into account, a novel method is proposed. Some free weighted matrices are employed in the analysis processing and are selected through solving LMIs appropriately, so this approach is less conservative and more general than the existing results.

Acknowledgments. This work is supported by National Natural Science Foundation of China under the grant 60604001 and 60674081, Hubei Provincial Natural Science Foundation under the grant 2008CDB316 and 2008CDZ046, and Scientific Innovation Team Project of Hubei Provincial College under the grant T200809.

References

1. Vicsek, T., et al.: Novel type of phase-transition in a system of self-driven particles. Physical Review Letters 75, 1226–1229 (1995)
2. Jadbabaie, A., Lin, J., Morse, A.S.: Coordination of groups of mobile autonomous agents using nearest neighbor rules. IEEE Transactions on Automatic Control 48, 988–1001 (2003)
3. Ren, W., Beard, R.W.: Consensus seeking in multiagent systems under dynamically changing interaction topologies. IEEE Transactions on Automatic Control 50, 655–661 (2005)
4. Moreau, L.: Stability of multiagent systems with time-dependent communication links. IEEE Transactions on Automatic Control. 169-182 (2006)
5. Olfati-Saber, R., Murray, R.M.: Consensus problems in networks of agents with switching topology and time-delays. IEEE Transactions on Automatic Control 49, 1520–1533 (2004)
6. Tian, Y., Liu, C.: Consensus of multi-agent systems with diverse input and communication delays. IEEE Transactions on Automatic Control 53, 2122–2128 (2008)
7. Yu, H., Jian, J.G., Wang, Y.J.: Weighted average consensus for directed networks of multi-agent. Microcomputer Information 23, 239–241 (2007)
8. Ren, W.: Multi-vehicle consensus with a time-varying reference state. Systems & Control Letters 56, 474–483 (2007)
9. Sun, Y.G., Wang, L., Xie, G.M.: Average consensus in directed networks of dynamic agents with time-varying delays. In: Proceedings of IEEE Conference on Decision & Control, vol. 57, pp. 3393–3398 (2006)
10. Lin, P., Jia, Y.M.: Average consensus in networks of multi-agents with both switching topology and coupling time-delay. Physica A 387, 303–313 (2008)
11. Sun, Y.G., Wang, L., Xie, G.M.: Average consensus in networks of dynamic agents with switching topologies and multiple time-varying delays. Systems & Control Letters 57, 175–183 (2008)
12. Lin, P., Jia, Y., Lin, L.: Distributed robust H∞ consensus control in directed networks of agents with time-delay. Systems & Control Letters 57, 643–653 (2008)
13. Horn, R.A., Johnson, C.R.: Matrix analysis. Cambridge University Press, Cambridge (1985)

Multi-Agent Cooperative Reinforcement Learning in 3D Virtual World

Ping Zhang, Xiujun Ma*, Zijian Pan, Xiong Li, and Kunqing Xie

Key Laboratory of Machine Perception (Minister of Education), Peking University, 100871 Beijing, China
{zhangping,maxj,lixiong,kunqing}@cis.pku.edu.cn, pzj1@163.com

Abstract. A virtual world is an online community in the form of a computer-based simulated environment, through which users can interact with one another and use and create objects. The non-player characters (NPC) in virtual world are following a fixed set of pre-programmed behaviors and lack the ability to adapt with the changing surrounding. Reinforcement learning agent is a way to deal with this problem. However, in a cooperative social environment, NPC should learn not only by trial and error, but also through cooperation by sharing information. The key investigation of this paper is: modeling the NPCs as multi-agent, and enable them to conduct cooperative learning, then speeding up the learning process. By using a fire fighting scenario in Robocup Rescue, our research shows that sharing information between cooperative agents will outperform independent agents who do not communicate during learning. The further work and some important issues of multi-agent reinforcement learning in virtual world will also be discussed in this paper.

Keywords: Multi-Agent Reinforcement Learning, Cooperative, 3D Virtual World, Robocup Rescue.

1 Introduction

A virtual world is an online community in the form of a computer-based simulated environment, through which multi-users can interact with one another and use and create objects [1]. Lots of virtual worlds depict a world very similar to the real world, with real world rules and real-time actions, and communication. To support the running of a virtual world, there need some non-player characters (NPC) play some roles. To make the world more real, NPCs need be smart. And reinforcement learning agent is a way to improve NPCs' intelligence.

Reinforcement learning agents identify the new events of environment change, then they can incrementally learn an efficient decision policy over a sate space by trial-and error [2]. However, when a task is too big for a single agent to handle, the agents may cooperate to accomplish the task, which is common in human society, such as a disaster rescue, there need the firefighters, ambulance and polices cooperate with each other. In such a cooperative social environment, agents should learn not

* Corresponding author.

only by trial and error, but also through cooperation such as sharing information. They together will outperform any single agent as matter of the fact that the more information, the more better chance of receiving rewards [3].

There are quite a few studies on Reinforcement Learning in Theory. M. Lauer and M. Riedmiller adopts The Distributed Q-learning algorithm [4], it solves the cooperative task without assuming coordination, but it is only valid in the deterministic setting [5]. Ming Tan address the multi-agent reinforcement learning by cooperative is a better way in theory, and the study is mainly focus on homogeneous agents [6]. Kathryn Merrick is a pioneer of using virtual world to research agent learning. Merrick introduced a motivated reinforcement learning [7], which is able to help agents continually identify new events on which to focus their attention and learn about. The work is mainly about single agent learning.

However, most works on reinforcement learning has focused on the single agents, and many research of multi-agent learning are still at the theoretical level. Virtual world is very similar to the real world and it is the best test bed to take multi agent learning theory into practice, and it provides a visual environment that we can observe the whole agent learning process directly as well.

The key investigation of this paper is: if cooperation is done intelligently among agents, each of them will benefit from others. So we model the NPCs as agents, and design cooperation mechanisms to enable them to conduct cooperative learning, then speeding up the learning process. By using a fire fighting scenario in Robocup Rescue implemented in Second Life Virtual World. We set up three experiments, the first one we want to show the advantage of reinforcement learning and study the effect of agents' sense field of depth, and second, we study the effect of two cooperation mechanisms we designed, last, we will show homogeneous and heterogeneous agents can both benefit from cooperation for a multi targets task. Through a series of experiments, we make an evaluation at the performance of multi-agent cooperative reinforcement learning and draw a conclusion.

This paper is organized as follows. Section 2 gives a brief introduction of the reinforcement learning and cooperation mechanisms we designed. Section 3 describes the experiment scene Section 4 describes the content of experiment and shows the result. Section 5 concludes.

2 Multi-Agent Cooperative Reinforcement Learning

Reinforcement learning is an on-line technique used by an agent for learning behavior through trial and error interactions with a dynamic environment. The aim of reinforcement learning is to maximize the long-term discounted reward per action [2].

In this study, each agent uses the basic Q-learning algorithm. Q-learning is one of the reinforcement learning technique that works by learning an action-value function that gives the expected utility of taking a given action in a given state[8].This algorithm is guaranteed to converge to an optimal decision policy for a finite Markov decision process[8]. Q-learning algorithm need not be given a model of its environment and can be used on-line and it is easy to implement. The algorithm's process as follows:

(a)First, from the current state *s*, select an action *a*. This will cause a receipt of an immediate payoff *R(s,a)*, and arrival at the next state *s'*. *R* is the payoff function.
(b)Second, update *Q(s,a)* based upon this formula:

$$Q(s,a) = \beta * [R(s,a) + \gamma * \max\{Q(s',b)\} - Q(s,a)] \qquad (1)$$

where β is the learning rate and $0 < \gamma < 1$ is the discount factor
(c)Final Go to (a) until enough iteration

Our study wants to show cooperative agents will outperform independent agents who do not communicate during learning. So we design two ways to let these agents to cooperate with each other by sharing information. First, agents can communicate the instant information, such as sensation, action, and reward. The method is like this:

(a)Instant information is mainly impact the payoff function *R*
(b)So an agent will use its sensation first, and *CR* is the cooperation function that give the additional information from its partners

$$R(s,a) = R(s,a) + \sum CR(s) \qquad (2)$$

(c)Use (1) to update *Q(s,a)*

Second, agents can tell each other their decision policies which are the action-value function of Q learning. The method is like this:

(a)After update its *Q(s,a),*an agent will ask its partners' *Q(s,a)*, and then choose the best one

$$Q(s,a) = \max\{Q(s,a), Q_p(s,a)\} \qquad (3)$$

These cooperation mechanisms can be applied into the homogeneous agents. For heterogeneous agents, as they have different goal, we just apply the first one, and through experiments, we also find that the cooperative agents perform better than independent ones. As a result, if agents can cooperate with each other, they will benefit a lot. The detail of these cooperation mechanisms will show in the section 4.

3 Experiment Scenario

In order to make our experiments more convincing, all the experiments in this study involve a Robocup Rescue scene implemented in the Second Life virtual world. At this scene, as shown by Figure 1, there are two kinds of agents: the ball represents the firefighter while the cube represents the ambulance, a grid world represents a city, each block is a building, among them, a red one is a building on fire, a blue one is a rescue point. On each time step, each agent has five possible actions to choose from moving up/down/left/right and extinguish fire/rescue people within the boundary. At the very beginning, agents make random moves as a result of equal Q-values. Two agents can stay at the same block. And a fire is extinguished or the people are rescued when the agent has arrived at the fire block/rescue point and do the extinguish/rescue action. Agents will receive reward from environment for each move.

Every agent in our experiments has a limited sense field of depth, which is adjusted to different number for different experiment. The sense field of depth is shown at Figure 2. See Figure 2, the depth of the agent is 2, that is to say, agent can sense all the information from a field whose center is the agent's location and radius is 2. With the sensation's help, agent can speed up the whole learning process: firstly, agent will choose the action which has the highest Q-value, and then if there are two maximum Q-values and the fire building is in the sensation field, the agent will choose the action at the fire building's direction. Each agent's sensation is represented by a vector (x, y) which is relative distance of the closest disaster place to the agent's location.

Each run of each experiment will consist of 100 trails. For every trial, all agents will begin with in random locations. Agents will choose the actions by the highest Q-value. Each trial ended when all the fire is extinguished and all the people are rescue. All experiments measure the total number of step per run. The Q-value will be convergence.

The Q-learning parameters were set at β=0.8□γ=0.9. These values are reasonable for these experiments. Experiment's parameters also include the number of fire place, the number of firefighter/ambulance agent and agent's the sense field of depth.

For each experiment, we set up a comparison group. The real question we want to discuss is at the virtual world environment, whether or not cooperation among homogeneous/heterogeneous agents can further improve the learning performance, whether or not multi agent learning can be applied to the virtual world.

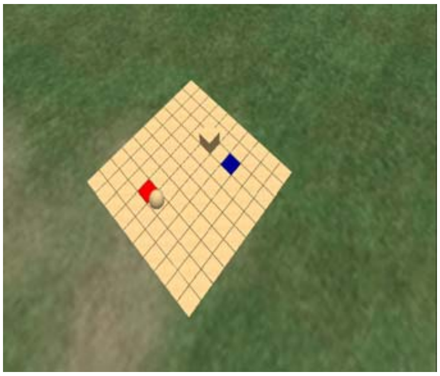


Fig. 1. The Experiment World

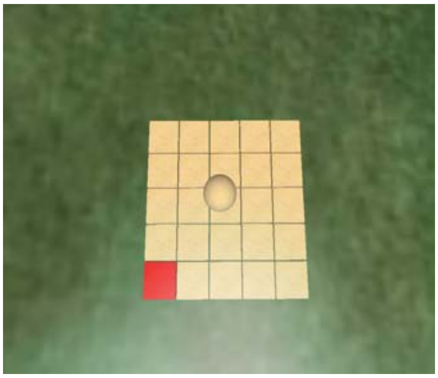


Fig. 2. The sense field of depth is 2

4 Experiment Evaluation

4.1 CASE 1: Single Agent Reinforcement Learning

In order to show the advantage of reinforcement learning and study the effect of agents' sense field of depth, we study the single agent reinforcement learning at first. So we choose the one firefighter/one fire building task. At the first experiment of this case, we set the agent's sense field of depth to 1, we want to test the performances of basic independently learning agent which has no help from sensation in the learning process. Later we extend the sense field of depth to 2 and 3, and then we demonstrate that the deeper sense field of depth is, the better the performance of agent learning shows.

Table 1 shows the average numbers of steps and average time of different kind of agent to extinguish a fire in training after 100 trails. The random one is the agent without learning, its average step is 92.482, when the learning was introduced to the agent, the result has improved a lot, even without sense, the number of average steps is 18.364. From this table we can conclude, (a) Reinforcement learning surely can improve the agent's ability. (b) The more information the agent collect, the better result will receive. As the agent's sense field of depth increases, the fewer steps it takes. And time spent is reduced too.

Table 1. Average Steps/Time to Extinguish a Fire: Random versus Learning Agent

Agent Type	Sense Field of Depth	Average Steps To Extinguish a Fire	Average Time To Extinguish a Fire(second)
Random	1	92.482	22.534
Learning	1	18.364	4.747
Learning	2	14.494	3.916
Learning	3	10.352	2.773

4.2 CASE 2: Multi Cooperative Agent Reinforcement Learning

After the single learning experiment, now we proof the sharing information between cooperative agents will outperform independent agents' learning. The cooperative of agent in our study will be sharing and exchanging information. And the ways of sharing information are sharing sensation information and sharing policies.

First, we study the effect of sharing sensation information among agents, we choose a two firefighters/one fire building task. To serve as a partner, a firefighter agent takes searching and fire extinguishing jobs. At this experiment, agent's sense field of depth is 2. When each agent takes a step in a learning trail, as a searcher, one agent will send its all sensation information to each other. Therefore, an agent can incrementally compute the location of the fire building sensed by its partner. We want to demonstrate that more sense information will help improve agent's learning.

Second, we study the effect of sharing policies between agents in a two firefighters/one fire building task. As in a learning trail, agents use Q value to choose action at every step. So we consider letting agents exchange their individual decision policies. We will show that such cooperative agents can speed up learning greatly, measured by the average number of steps in learning trails. If agents execute the same task, their decision policies can be different during the learning period as they may have explored the different place of this virtual world. So, the two agents can complement each other by exchanging their individual decision policies and they can benefit from the other's experience. We can assume that each agent can send its current policy to each other simultaneously, then for each Q value, agent can choose the better one. This method helps an agent adopt another agent's decision policy and improve the learning. Also, at this experiment, agents' sense field of depth is 2.

In order to demonstrate the benefits from cooperate we set a comparison experiment which is also a two firefighter/one fire building task, but there is no cooperate between agents. Two agents will find the fire building independently. Once one of them extinguishes the fire, one learning trail is over.

From Table 2 we can see:(a) Both cooperative and independent agents take fewer steps than one agent to extinguish a fire; (b) Cooperative agents outperform independent agents: the average number of independent agents' cumulative steps is 22.136 while the this number of cooperative ones are 18.16 and 15.319 respectively (c) Sharing policies agents have a faster learning speed than sharing sensation agents: the average steps of Sharing Policy is 15% less than Sharing Sensation Information. Also we can see Figure 3, cooperative agents converged much quicker than independent agents did and Sharing Policy is quicker than Sharing Sensation Information. In our case, it seems that exchanging knowledge (decision policies) is better than exchanging information (sensation). However, although the exchanging decision policy is better, it takes more communicating time than exchanging sensation. If the policy exchanging is too frequency, the more communication time will be taken. Generally speaking, during the experiment, cooperative learning outperforms independent learning. Their differences in performance are statistically significant according to the average steps in a learning trail.

Table 2. Average Steps to Extinguish a Fire: Independent versus Cooperative Agent

Agent Type	Agent A: Average Steps to Extinguish a Fire	Agent B: Average Steps to Extinguish a Fire
Independent	10.958	11.178
Sharing Information	9.068	9.092
Sharing Policy	7.631	7.688

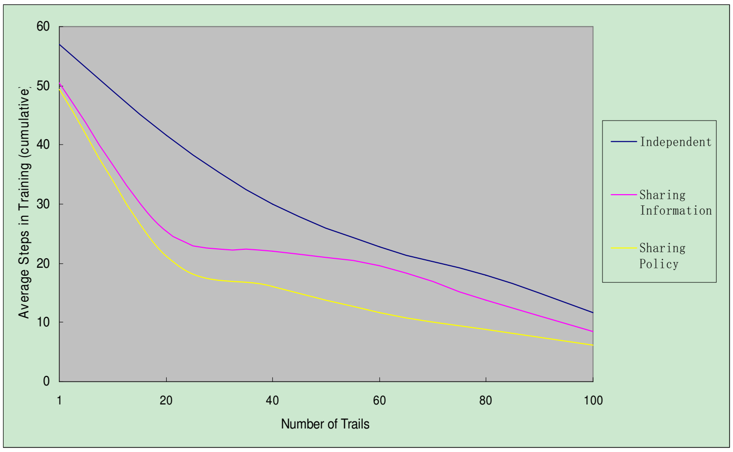


Fig. 3. Independent versus Cooperative Agent

4.3 Case 3: Multi Cooperative Agent Reinforcement Learning: Homogeneous and Heterogeneous

In the previous two case studies, there only one fire building and the agents are homogeneous. In this case we will demonstrate that cooperative agents can learn to perform the multi-target task better than independent agents, both for homogeneous and heterogeneous agents.

Assume that the agents' sense field of depth is 2. Let's first consider the homogeneous agents, that is a two firefighters/two fire building task. Only the two fires are extinguished, one learning is end. When two independent firefighters are given the job, each of them tends to find a fire building directly. It may cause that two firefighter come to a same fire building, which wastes lots of time than one firefighter extinguishing one fire at the same time.

The problem with independent agents is that they ignore each other. They cannot distinguish the situation whether one is nearby a fire or not. If each agent can cooperate with each other, the problem will be solved. To address it, we can extend the method of sharing information that exchanging not only the relative location between a fire building and the firefighter, but also the relative location between the firefighter and its partner. So if one agent knows its partner is closer to a fire building, than it will choose to find another fire building. That will improve the learning process than the independent one. Although this kind of cooperative brings more communication cost, it will eventually overtake the independent one.

The result from Table 3 and Figure 4 shows that the independent agents performance fluctuated and on the average they need 36.412 steps to finish the job. This is proof the problem between the two independent agents. While after 20 trails, cooperative agents soon outperform the independent agents. The average cumulative steps of cooperative agents are 36.05% less than independent one.

Table 3. Average Steps to Extinguish a Fire: Independent versus Cooperative Agent (cumulative)

Agent Type	Average Step to Extinguish a Fire (cumulative)
Independent	36.412
Sharing Information	23.284

Next we will consider the heterogeneous situation, that is a one firefighter, one ambulance/one fire building, one rescue point task. At this scene, firefighter should find the fire building and ambulance should get to the rescue point to save people. Each of them can do it own task independently. A learning trail is over, if and only if two tasks are all finished, that is to say, although one has finished its job, it should wait to start a new job until the other has finished. It may also cause a waste of time. However, if they can cooperative, things will be different. For example, when a firefighter is searching for the fire building, it may pass the rescue point while the ambulance has no idea about where the rescue point is. So if firefighter can send a message to tell the ambulance that the relative location between it and the rescue point as well as the relative location between it and its partner. (So does the ambulance) The learning

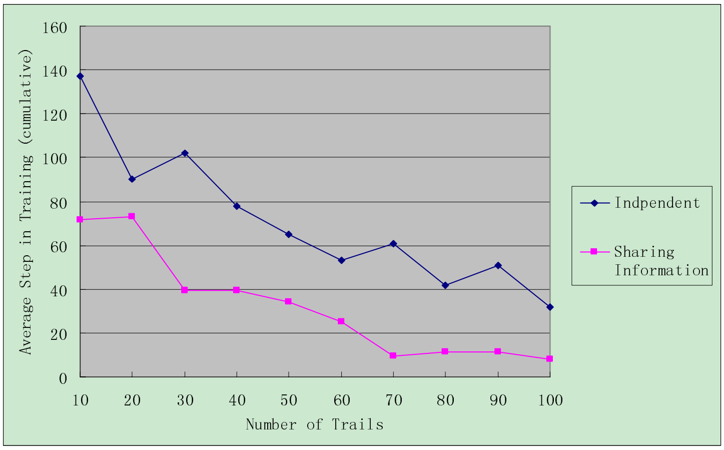
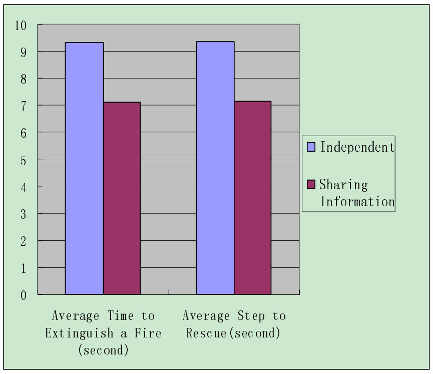


Fig. 4. Independent versus Cooperative Agent for homogeneous agents

Table 4. Average Steps/Time to Extinguish a Fire and Rescue: Independent versus Cooperative

Agent Type	Average Step to Extinguish a Fire	Average Step to Rescue	Average Time to Extinguish a Fire (second)	Average Step to Rescue(second)
Independent	20.948	20.404	9.332	9.343
Sharing Information	15.391	15.882	7.122	7.137



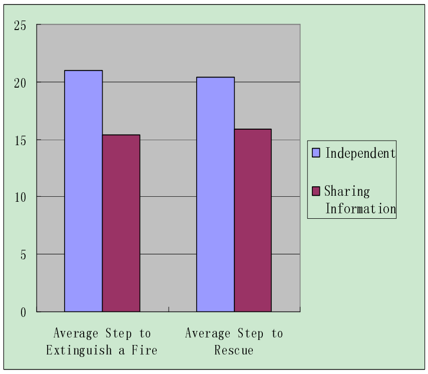


Fig. 5 and Fig. 6. Independent versus Cooperative Agent for heterogeneous agents

will speed up. The result measured by average step of learning trail proves this point. From Table 4 we can see, the performance of cooperative agents is much better than independent ones. Figure 5&6 show a much clear difference between the two experiment group, not only in average step, but also in average time.

As a result, in our study, homogeneous and heterogeneous agents can both benefit from cooperation for a multi targets task.

5 Conclusion

This paper shows that sharing information and knowledge between cooperative agents will outperform independent agents who do not communicate during learning. This paper also identifies that homogeneous and heterogeneous agents can both benefit from cooperation. And our results are base on a scene implemented in the Second Life virtual world, we believe the conclusions can be applied to the other cooperative scene in a virtual world and it is one way to solve the NPCs learning problem. On the other hand, sharing information and knowledge comes with a communication cost and larger state space. These tradeoffs must be taken into consideration for learning agents to cooperate with each other. These problems will be studied in our future work.

Acknowledgement

This work was supported by The National High Technology Research and Development Program of China (863 Key Program) No.2007AA120502 and the National Natural Science Foundation of China under Grant No.40801046 and No. 60874082.

References

1. Bishop, J.: Enhancing the understanding of genres of web-based communities: The role of the ecological cognition framework. International Journal of Web-Based Communities 5(1), 4–17 (2009) (Available online)
2. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research, 237–285 (1996)
3. Shoham, Y., Powers, R., Grenager, T.: Multi Agent Reinforcement Learning: a critical survey (2003)
4. Lauer, M., Riedmiller, M.: An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In: Proceedings Seventeenth International Conference on Machine Learning (ICML-2000), Stanford University, US, 29 June – 2 July, pp. 535–542 (2000)
5. Busoniu, L., Babuska, R., De Schutter, B.: Multi Agent Reinforcement Learning: A survey. In: Proceedings of the 9th International Conference on Control, Automation, Robotics and Vision (ICARCV 2006), Singapore, pp. 527–532 (2006)
6. 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, San Francisco (1993)
7. Merrick, K., Maher, M.L.: Motivated Reinforcement Learning for Non-Player Characters in Persistent Computer Game Worlds. In: Proceedings of the 2006 ACM SIGCHI international conference on Advances in computer entertainment technology (2006)
8. Watkins, C.J.C.H.: Learning from Delayed Rewards. PhD thesis, Cambridge University, Cambridge, England (1989)

Author Index