Soft Comput (2016) 20:555–566
DOI 10.1007/s00500-014-1522-3

METHODOLOGIES AND APPLICATION

Hybrid immune algorithm based on greedy algorithm and delete-cross operator for solving TSP

Guo Pan · Kenli Li · Aijia Ouyang · Keqin Li

Published online: 25 November 2014

Abstract This paper first introduces the fundamental principles of immune algorithm (IA), greedy algorithm (GA) and delete-cross operator (DO). Based on these basic algorithms, a hybrid immune algorithm (HIA) is constructed to solve the traveling salesman problem (TSP). HIA employs GA to initialize the routes of TSP and utilizes DO to delete routes of crossover. With dynamic mutation operator (DMO) adopted to improve searching precision, this proposed algorithm can increase the likelihood of global optimum after the hybridization. Experimental results demonstrate that the HIA algorithm is able to yield a better solution than that of other algorithms, which also takes less computation time.

Keywords Delete-cross operator · Dynamic mutation · Greedy algorithm · Immune algorithm · TSP

1 Introduction

Traveling salesman problem (TSP) is a classical combinational optimization problem with strong performances in engineering and wide applicability. TSP problem can be formally described as: already known N cities C={$C_1, C_2, \ldots, C_N$}, and the distance between two randomly selected cities is denoted as $d(c_i, c_j)$, then the solution of the closed path passing all the cities within C just once will be C_π={$C_{\pi(1)}, C_{\pi(2)}, C_{\pi(N)}$} to minimize the total travel distance $\sum_{i=1}^{N-1} d(C_{\pi(i)}, C_{\pi(i+1)}) + d(C_{\pi(N)}, C_{\pi(1)})$.

There are many types of TSPs, such as multiobjective TSP (Shim et al. 2012), dynamic TSP (Cheong and White 2012), Dubins TSP (Le Ny et al. 2012), sequence-dependent TSP (Alkaya and Duman 2013), double TSP (Carrabs et al. 2013). For large-scale TSP problems, people tend to figure out an acceptable approximate solution within the time limit. The approximate algorithms for TSP problems can be categorized into tour construction algorithms and tour improvement ones. Tour construction algorithm starts from an illegal solution and changing the path gradually until the legal route is acquired. These kinds of algorithms cover the Clarke–Wright algorithm (Clarke and Wright 1964), nearest neighbor algorithm (Hurkens and Woeginger 2004), greedy algorithm (Hassin and Keinan 2008) and Christofides algorithm (An et al. 2012), etc. Tour improvement algorithm searches for a solution with better quality by adopting a certain strategy after the acquisition of initial legal solution. These kinds of algorithms include local search strategy (Opt, LK, LKH, LK-circulation (Karapetyan and Gutin 2011), etc), tabu search (Pedro et al. 2013), simulated annealing (Kalender et al. 2013), cuckoo search algorithm (Ouaarab et al. 2013), particle swarm algorithm (Beheshti et al. 2013), ant colony optimization algorithm (Gan et al. 2010; Cecilia et al. 2013; Mavrovouniotis and Yang 2013; Mora et al. 2013), neural network (Yang and Yi 2013), memetic algorithm (Badillo et al. 2013), GA–PSO–ACO (Deng et al. 2012), artificial bee colony algorithm (Marinakis et al. 2011; Kıran et al. 2013), genetic algorithm (Yuan et al. 2013; Nagata and Soler 2012), multiagent optimization system (Xie and Liu 2009),

Communicated by V. Loia.

G. Pan · Kenli Li (✉) · A. Ouyang · Keqin Li
College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, Hunan, China
e-mail: lkl@hnu.edu.cn

G. Pan
e-mail: panguo@hnu.edu.cn

A. Ouyang
e-mail: oyaj@hnu.edu.cn

Keqin Li
Department of Computer Science, State University of New York, New Paltz, NY 12561, USA
e-mail: lik@newpaltz.edu

and artificial vaccines (Montiel and Diaz Delgadillo 2013), etc.

By imitating the mechanisms of the biological immune system which goes through immune response, immune memory and immune regulation, the artificial immune system (AIS) constructs a self-organizing artificial intelligence system with strong robustness. With more research related with artificial immune system going on, the issue of artificial immunity has become another hot research area following the step of neural network, fuzzy logic and evolutionary computation (Dasgupta and Forrest 1999). Based on immunity theories, several heuristic algorithms have been proposed up to now. In 1995, Hunt and his colleagues presented the first artificial immune system model, i.e., B Cellular Network Model (Hunt and Cooke 1995). Chun (1997) proposed another immune algorithm based on immune network theories, which conducts selective operation according to individual fitness value and similarity, thereby restraining similar individuals and keeping the diversity of the population. From this perspective, the algorithm has global optimum searching ability. De Castro and Von Zuben (2002) put forward a clonal selection algorithm which imitates the clonal selection in immune system. Specifically speaking, by conducting cloning and mutating operations to selected individuals for local optimum searching, the algorithm replaces the individuals with low fitness value in the population with randomly generated individuals to ensure the diversity of the population.

The selection, crossover, mutation and immune operator in immune algorithm are considered important influential factors on the algorithm's global search performance. However, the searching direction of the immune algorithm in immune operation is always overlooked. To acquire high-performance candidate solution, the searching direction should be able to indicate the potential heading direction of immune algorithm in certain areas. The adoption of random numbers in immune operation usually leaves candidate solution in a random position in the searching space. If the position of the optimum is far from the current searching space, which cannot be preliminarily recognized, the speed of locating the optimum will get lowered, especially in the case of multi-optimization problems.

Aiming to solve the problems of slow convergence and low accuracy of the immune algorithm, this paper has come up with the HIA algorithm for TSP. The main contributions of the paper are summarized as follows. By adopting a hybrid method, the immune algorithm is used for global search in individuals while the greedy algorithm is used to initialize population and conduct local search in chromosome jointly with delete-cross operator. When upgrading the individuals, we adopt the high-frequency mutation operator based on dynamic mutation probability to improve the mutation operator. These strategies stress the potential searching direction of immune algorithm, which gives progenies chance to advance towards the direction and search for other space of high quality. Strengthened superior antibodies therefore achieve mature affinity, and and strike a balance between deep exploitation and broad exploration. The simulation on TSP cases demonstrates that HIA possesses reliable global convergence and rapid rate of convergence.

The rest of our paper is organized as follows: The definitions of Directed Graph and TSP are outlined in Sect. 2. We then describe the basic principles of IA, GA and DO in Sect. 3. Section 4 proposes the HIA algorithm for TSP. Section 5 presents the experimental results and analysis. Finally, we conclude our work in Sect. 6.

2 Description of TSP

Definition 1 *Directed Graph* It is assumed that the triple of directed graph D consists of V, E and F. V is a nonempty set here, and its elements are called the nodes of directed points; E is a set, and its elements are called segmental arc (edge); F is a map (function) from E to $V \times V$.

Definition 2 *TSP* It is assumed that $C = \{c_1, c_2, \ldots, c_n\}$ is the set of n cities. $L = \{l_{ij} \mid c_j, c_j \subset C\}$ is a set consisting of connections of two random factors (cities) within Set C. $d_{ij}(i, j = 1, 2, , n)$ is the Euclidean distance of l_{ij}, which is

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \qquad (1)$$

$G = (C, L)$ is a directed graph. The purpose of TSP is to search for the shortest Hamilton cycle in the directed graph D, which essentially is the shortest closed tour visiting all the factors (cities) in $C = \{c_1, c_2, \ldots, c_n\}$ which happens only once.

Metaphorically speaking TSP can be simply described as: assume that there are n cities, one traveling salesman starts from one city, travels to all the others one by one and heads back, and to the departure city. A shortest route is needed.

TSP can be categorized into symmetric traveling salesman problem and asymmetric traveling salesman problem. If the distance from A to B is the same as the distance from B to A, then it is called symmetric TSP, otherwise, it is an asymmetric TSP. The issue of symmetric TSP will be the focus of this paper.

Given data of TSP include the weight of each edge in a finite complete graph whose purpose is to search for a Hamilton cycle with the minimum total weight. There are different closed paths for a TSP with n cities. The best way to solve the problem is global search, but when n is comparatively big, it becomes impossible to find the exact optimum with global search. As for the high representativeness and wide applicability of TSP, many questions can be modeled and solved as a kind of TSP problem.

3 Fundamental algorithms

3.1 Immune algorithm

3.1.1 Basic principles

The immune system is a natural defense one protecting living organisms from intruding pathogens or bacteria. The permanent cycle of immune response against such pathogens displays many dynamic characteristics (Zhang and Qian 2011; Zhang et al. 2014). Biological immune system is a highly evolved biological system which aims to distinguish external harmful antigen from internal organizations, thus maintaining the stable organism. From the perspective of calculation, biological immune system is a system with high concurrency, distribution, self-adaptation and self-organization. It is superior in terms of learning, recognition and memory ability.

Immune system has the following features:

- The ability to generate diversified antibodies. The immune system can generate a mass of antibodies to resist all kinds of antigen.
- Self-regulating mechanism. The immune system possesses a mechanism to maintain balance, which can regulate itself by the production of appropriate quantity of necessary antibodies after either inhibiting or enhancing the antibodies.
- Immune memory function. Part of the cells generating antibodies will be preserved as memory. During the future invasion of congeneric antigens, corresponding memory cells will be triggered immediately and produce a mass of antibodies.

An immune algorithm is developed from basic theories on artificial immunity (IEEE 2013); it is the expansion and development of the application and studies on artificial immunity theories. As artificial immunity is based on the fundamental concepts and theories of biological immune system, the theories of biological immune system are considered as the direct source of immune algorithm. Referring mainly to features of the immune system such as antigen recognition immune memory and immune regulation, immune algorithm applies concepts and theories of immune system into the calculation. Among the features, antigen recognition is the process in which the recognition is finished by expressing the mutual matching and selection between epitope on the surface of antigen and the counterpoint on the surface of antibody. Immune memory means that the immune system can maintain and memorize the antibody which reacts to antigen as memory cells; when congeneric antigens invade again, the system will instantly recognize the process of reacting. Immune regulation happens during immune reaction when the stimulus of antigen to immune cells is reduced due to the generation of abundant antibody. Against such backdrop, the differentiation and multiplication of antibody are restrained, and the balance between antigens and antibodies as well as between two antibodies can be maintained to a certain degree. Besides, the core of immune algorithm lies in the immune vaccine, which is rooted in the concepts and theories of biological vaccine. Biological vaccine was invented by medical scientists based on the immune memory feature of the immune system. Quick recognition of antigen can be realized through vaccine injection.

The immune algorithm is a calculation model for solving all kinds of combinatorial search and optimal computation problems in the fields of science, technology and engineering by integrating the genetic algorithm on the basis of abstracting and reflecting artificial immune theories. When adopting this model, we can abstract out a problem and solution as a antigen and antibody respectively, and the vaccine in the model corresponds to certain characteristic information of the solution of the problem to be solved.

The core concept of immune algorithm is to upgrade the population fitness and speed up the iterative process to prevent degeneration of the population through vaccination and immune selection on the basis of rational extraction of vaccine. The immune algorithm inherits the genetic operator of genetic algorithm, which results in a stronger global search ability. Moreover, the added immune operators enable the immune algorithm to effectively prevent degradation in the late stage of generic algorithm and accelerate convergence. In addition, based on related theories, the probability of the immune algorithm to converge is 1, which implies that it has strong convergence properties.

Both immune algorithms and genetic algorithms adopt global search strategies, and give priority to the information exchange among individuals in the population. Hence, they share a lot in common. For example, they possess almost identical algorithm structure with an iterative process consisting of generation of an initial population, evaluation criterion calculation, information exchange among individuals in the population, and generation of a new population. The optimum of a problem will be acquired through its higher probability. Besides, the two algorithms are both parallel and inherently advantageous in that it can be combined with other intelligent calculation methods.

Some differences do exist, though, between immune algorithms and genetic algorithms, which mainly lie in ways of evaluation, selection and production of individuals. In genetic algorithm (Xu et al. 2014), evaluation of individuals is done through calculating individual fitness, which is the only criterion for selecting parent individuals in the algorithm. Whereas in immune algorithms, evaluation of individuals is done through calculating affinity, on which individual selection is based. Individual affinity contains antibody–antigen affinity (matching degree) and antibody–antibody affinity (level of similarity). It reflects the diversification of the real immune system, so its evaluation of individuals is more comprehensive and therefore, the individual selection is more reasonable in immune algorithms. Besides, new individuals can be generated through immune operations such as crossover and mutation in immune algorithm. Although inherent immune operations such as crossover and mutation are broadly adopted in immune algorithm, some mechanisms which are missing in the algorithm can be used to generate new antibodies. These mechanisms include clonal selection, immune memory and vaccination. Meanwhile, immune algorithms can promote or restrain the generation of antibodies. This reflects the self-adjusting function of immune action and ensures the diversity of individuals. Moreover, the immune algorithm converges fast , making it less likely to get trapped in local optimums (Ding et al. 2012; Chen et al. 2013).

3.1.2 Fundamental definitions

Definition 3 Antigen in bioscience: an antigen is a kind of material which can stimulate the immune system and induce it to generate immune response and then generate intracorporal or extracorporeal idiosyncratic reaction with corresponding immune response products. In the algorithm of this paper, an antigen means all possible wrong genes, which are non-optimal genes of individuals.

Definition 4 Antibody in bioscience: antibody means the immune globulin which can realize specific binding with antigen and is produced in the process of immune cells being transformed into plasmocyte after the immune system gets stimulated by the antigen. An antibody in the paper refers to the new individual based on certain corrected individual genes by vaccine. The process of correcting certain individual genes using the vaccine is called vaccination, and its purpose is to eliminate negative effects brought by the new individual when it comes into being.

Definition 5 Immune vaccine: immune vaccine here refers to the estimation on the acquired optimum individual gene based on the environment evolution or the problem which needs to be worked out.

Definition 6 Immune operator: similar to the theory of immunity in bioscience, an immune operator is divided into two categories: pan-immunity and objective immunity. These two categories correspond to nonspecific immunity and specific immunity in bioscience, respectively. Pan-immunity is the immunity type in which the immune operation is conducted in each stage after the mutation operation of each individual in the population. Objective immunity is the immunity type in which individuals show immune reactions only on application points after certain judgment of the mutation operation of individuals. The former mainly happens in the initial stage of individual evolution, which is basically not working during the evolution, otherwise, the assimilation in the usual sense is highly likely to happen. The latter normally accompanies the whole process of the population evolution, and is a frequently used operator in immune operation.

Definition 7 Immune regulation: during the process of immunization, the generation of many antibodies will decrease the antigen stimulation to immune cells and restrain the differentiation and multiplication of antibodies. And the mutual stimulation and restraint exist among simultaneously generated antibodies. The mutual restrictions between antigens and antibodies as well as between different antibodies enable the antibodies to maintain certain intensity and further keep the immune balance of the organism.

Definition 8 Immune memory: immune memory means that the immune system can memorize and keep the antibodies which can react to antigens. When congeneric antigens invade again, corresponding memory cells will be activated, thereby producing a great many antibodies. The immune reaction time will be substantially shortened. Antigen recognition is the process in which the recognition is finished by expressing the mutual matching and selection between epitope on the surface of antigen and the chemical base on the counterpoint on the surface of antibody. The matching process is also a process of unceasing learning on antigen, and finally the most suitable antibody will be selected to combine with the antigen and the latter will be eliminated.

3.1.3 The flow of immune algorithm

Analyze the problem and the characteristics of its solution, then design the appropriate form of expression for the solution. The flow of immune algorithm is demonstrated as follows:

3.1.4 Components of immune algorithm

1. *Extraction of immune vaccine*
   In immune algorithm, vaccine refers to a kind of feature information extracted from the prior knowledge of detailed to-be-solved problems. It can be seen as an estimation of the best individual matching model for to-be-solved problems. As a unique mechanism for individual update and optimization, proper selection of the vaccine is the prerequisite for effective realization of immune operations, which is important to the operating

Algorithm 1 Immune algorithm

Input: $L, D, P, N, n, P_c, P_m, G$
Output: b
1: **Step 1**: Generate initial antibody population and extract vaccine.
2: **Step 2**: Update individuals: run crossover operator, mutation operator and vaccination operator.
3: **Step 3**: Calculate the fitness of each individual within the population.
4: **Step 4**: Execute immune selection operation: implement immune detection operator, immune balance operator and selection operator.
5: **Step 5**: Record the optimal individuals.
6: **Step 6**: Judge whether the maximum iteration has been achieved. If the answer is yes, then output the optimum; or goes back to **Step 2**.

efficiency and performance of the algorithm. Regarding the process of choosing vaccine for a certain problem, the vaccine can be made based on the feature information of the problem. Extraction methods vary according to different actual problems. For instance, while solving the TSP problem, the distance between different cities can be regarded as the vaccine; while applied to the categorization and cluster of model identification, the feature value distance between sample and template or samples can be deemed as the vaccine; the optimum solution for each generation can be seen as the vaccine as well to dynamically establish a vaccine database. When the current optimum has higher affinity than the worst vaccine in the library, the worst vaccine will get replaced.

2. *Vaccination operator*
As explained above, the vaccine comes from the prior knowledge of the problem. The amount and accuracy of the information has great influence on the performance of the algorithm.
Suppose the population size is N and $\alpha \times N$ antibodies are selected from the antibody population according to the probability α. Vaccination operation is conducted on them. According to known information, that is, previously extracted vaccine, vaccination is implemented for each gene position of the selected antibodies according to the vaccination probability P_i. If vaccination is decided to be done, the gene value relevant to each gene of the vaccine will be used to change the corresponding genes of current antibodies. This will make individuals more likely to adapt well.
3. *Immune detection operator*
The immune detection operator is able to judge whether the antibodies are optimized or not after receiving vaccination, that is, whether the affinity value of the antibodies with vaccination is higher than the previous antibodies or not. If yes, the antibodies with vaccination will be put among the new population; otherwise, the parent antibodies prior to the vaccination are used to replace the new antibodies.
In general evolutionary algorithms, the process of choosing operators does not detect new individuals, making the crossed and varied individuals worse than the parent individuals, that is, degeneration. This greatly affects the convergence of the algorithm. For immune algorithm, its convergence is fundamentally guaranteed by the immune selection operator. The immune selection can guide the evolution direction. In this way, individuals evolve towards optimized direction and the degeneration phenomenon which might happen in general evolutionary algorithms is avoided, thus improving the algorithm efficiency and convergence speed.
4. *Immune balance operator*
In the immune system, the concentration of antibodies with great adaptability constantly increases. When the concentration reaches a certain value, the production of such antibodies will be restrained. On the contrary, it will improve the production and selection probability of antibodies with low concentration. Such mechanism ensures the diversity of the updated antibody populations and avoids immature convergence to some extent. Thanks to the immune balance operators, the higher the antibody concentration, the more the inhibition; the lower the concentration, the greater the promotion.

1. *Concentration calculation* Concentration C_i is defined as the proportion of the antibodies with an adaptability close to that of the i individual in the population. See formula (1):

$$C_i = \frac{\sum_j (|\text{Fitness}(j) - \text{Fitness}(i)| \le \varepsilon)}{N} \tag{2}$$

In the formula, ε is adjustable parameters between 0 and 1 such as 0.5; N is the total number of current antibodies (population size).
2. *Concentration probability calculation*
A concentration threshold value is set to calculate antibodies whose concentration is higher than the set value. Suppose the amount is $k(1 \le k \le N$, N is population size). The concentration probability of the k antibodies with higher concentration is:

$$P_{d(k)} = \frac{1}{N}\left(1 - \frac{k}{N}\right) \tag{3}$$

Then the rest of the $N-k$ antibodies with lower concentration have a concentration probability of:

$$P_{d(N-k)} = \frac{1}{N}\left(1 + \frac{k}{N} g \frac{k}{N-k}\right) \tag{4}$$

It can be noted that the concentration probability of all antibodies is 1. The more the antibodies with higher concentration than the set threshold value in the population, the smaller the concentration probability $P_{d(k)}$ of antibodies with high concentration, and the greater the concentration probability $P_{d(N-k)}$ of antibodies with low concentration; the fewer the antibodies with higher concentration than the set threshold value, the greater the concentration probability $P_{d(k)}$ of antibodies with high concentration, and the smaller the concentration probability $P_{d(N-k)}$ of antibodies with low concentration.

3. *Calculation of the probability of selection*
   The probability of selection is composed of two parts: adaptability and concentration probability. The selection probability of antibodies with higher concentration is:

$$p = \alpha g p_f + (1-\alpha) g p_{d(k)} \tag{5}$$

and the selection probability of antibodies with lower concentration is:

$$p = \alpha g p_f + (1-\alpha) g p_{d(n-k)} \tag{6}$$

In the formula, p_f is the adaptability probability of antibodies and the defined as the ratio between the adaptability of antibodies and the total adaptability; pd is the depth probability of antibodies; 0< α < 1, 0< p_f < 1, 0< p_d < 1.

Obviously, from the selection probability it can be concluded:

(a) The greater the adaptability of antibodies, the greater the corresponding selection probability.
(b) The more the antibodies with higher concentration than the set threshold value in the population, the smaller the concentration probability $P_{d(k)}$ of antibodies with high concentration and the smaller the selection probability p. The probability for the antibody being selected is small, so it is restrained; on the contrary, the fewer the antibodies with higher concentration than the set threshold value, the greater the concentration probability $P_{d(k)}$ of antibodies with high concentration and the greater the p. The probability for the antibody being selected is high, which means its effect is enhanced. Hence, the greater the concentration of antibodies, the immune balance operator will make the antibodies more restrained, whilst the lower the concentration, it will promote the performance of antibodies.

3.2 Greedy algorithm

Greedy algorithm refers to choosing the best or optimized (the most favorable) in each step so as to bring about the best or optimized overall performance of the algorithm. For instance, in the problem of TSP, if the salesman chooses the nearest city every time, it can be regarded as a kind of greedy algorithm.

Greedy algorithm is particularly effective in solving the problem of optimal substructure. Optimal substructure means that the local optimum can determine the global optimum. Put simply, the problem can be divided into sub-problems for solution. The optimum for the sub-problems can be recurred to the optimum for the final problem.

The difference between greedy algorithm and dynamic planning lies in that the former selects the solution for each sub-problem without backspacing. Dynamic planning will save previous algorithm results which serve as a basis for the subsequent selection. It has the function of backspacing. Greedy algorithm can solve some optimization problems such as the minimum spanning tree in the chart and Huffman encoding. For other problems, greedy algorithm, however, usually cannot offer a desirable answer. Provided that a problem can be solved by greedy algorithm, greedy algorithm tends to be the best solution for the problem. Greedy algorithm is highly efficient and the answer it offers is close to the optimized results. It can also be used as auxiliary algorithm or directly to solve some problems which do not require precise results.

The basic principle of greedy algorithm is: It starts with the initial solution of a problem to approach the set goal step by step so as to obtain better solutions using the least time. When it cannot proceed anymore in a certain algorithm, it will stop. This algorithm has the following disadvantages: (1) it cannot guarantee that the final solution is the best; (2) it cannot be used for maximal or minimal solution problems; (3) it can only acquire a rough scope of the feasible solutions which satisfy certain constraints. The greedy algorithm can be applied extensively. For example, Prim algorithm seeking the minimal spanning tree and Kruskal algorithm are both good examples of greedy algorithms. The application-oriented algorithms of greedy algorithm include Dijksstras single-source shortest path and Chvatals greedy set cover heuristic method, etc.

3.3 Delete-cross operator

In order to accelerate the algorithm's convergence, delete-cross strategy can be adopted to delete the crossover paths of the travel route (Shi et al. 2007). For the position X of an antibody, $line[i]$ indicates the segment between the node i and $i+1$ the node in X. A Boolean function is defined to judge whether the segments are crossed or not. If they are crossed, the Boolean function is true; otherwise the function is false. The pseudo code of the delete-cross strategy can be demonstrated as follows:

$for(i = 1; i < m, i++)$
$if\ (crossover(line[i], line[j])\ is\ true)$

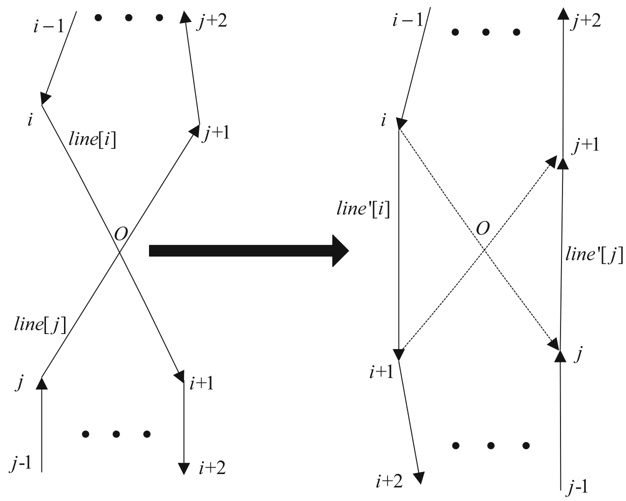

Fig. 1 Schematic diagram of delete-crossover

$$for(k = 0; k < (j - i)/2; k + +)$$
$$swap(x_{j-k}, x_{i+k+1})$$

It can be seen from Fig. 1 that: before and after the delete-cross strategy is adopted, nothing has changed in the two travel circuits except that segments $line[i]$ and $line[j]$ are respectively transformed to $line'[i]$ and $line'[j]$. In line with the principle that the sum of the two lines is greater than the third line in a triangle, a conclusion can be easily obtained from Fig. 1: $line'[i]+line'[j] < line[i]+line[j]$. It means that delete-cross strategy can improve the algorithm performance. If such a strategy is adopted for all antibodies with immune selection operator in the immune algorithm, the search quality will be greatly enhanced.

4 Hybrid immune algorithm for TSP

4.1 Basic principles

With added functions such as vaccine operator, immune detection operator and immune balance operator, the immune algorithm has been greatly improved in terms of updating and selecting individuals, as well as maintaining the diversity when compared with genetic algorithms. However, the algorithm is not perfect as it randomly initializes the population, and the process of updating individuals gives rise to some infeasible solutions for not using the greedy obsolete mechanism. Because of this, when it comes to solving NP-hard problems such as TSP, the convergence speed is slow and the search efficiency is yet to be strengthened. Hence, two local strategies , namely the greedy algorithm and delete-cross operator, are integrated into the immune algorithm, which serves as the the global algorithm. This, using the hybrid method, shapes the hybrid immune algorithm. The immune algorithm is used for the global search of individuals while the greedy algorithm for the initialization of population. Thereafter, the greedy algorithm is used together with the delete-cross operator used together for local seeking in the chromosomes. When individuals are updated, the mutation operator in the basic immune algorithm is modified and high-frequency mutation operator based on dynamic mutation probability is adopted so as to enhance the search efficiency and quality. The experiment shows that the hybrid immune algorithm is more effective and efficient than the basic one.

4.1.1 Greedy initialization of population

Most of the genes covered by the immune algorithm come from individuals themselves. Therefore, the quality of individuals determines the algorithm efficiency. If the adaptability value of all individuals in the population is poor, it will definitely affect the global performance of the algorithm, which is prominent in TSP problems. In order to overcome such shortcomings, all points in TSP problems are sequenced to build gene fragments based on the gene library, thereby building the chromosome string.

Build gene library
For N cities with TSP problem, C cities which are nearest to i city are encoded based on distances ranging from small to large, thus forming $N \times C$ $(c < n)$ matrix and further gene library $A_{N\times C}$. A_{ij} element is the code of the city whose distance is j to i city in terms of closeness. The line i element is the code of C cities which are close to i city. $A[i]$ and $A[j]$ elements are respectively the city which is the nearest to i city and the second nearest one to i city. It goes on like this. The cities ranking before C form gene library. Usually the value of C is roughly equal to 3.

Generate initial population
The first city i code is generated randomly. The code j of the city which is near to the city is selected first in the line i of the gene library. As the next traveling city, the city code h with closer distance in the line j is selected from the gene library. Based on this rule, the next cities are selected to form gene fragments. If all cities in the gene library have appeared in previous codes, the city which has never been used will be randomly selected.
The initial population generated by building gene library is currently the best solution for building population individuals. This is a kind of greedy selection strategy. The gene fragments of the gene library are characterized by short definition length, low order and high adaptability value and therefore, it is also called building

blocks. Such sub-string set (also called building blocks) with short definition length, low order and high adaptability plays an important role in the immune algorithm because the combination of these building blocks will become the chromosomes which have better performance. In the model principle, the building blocks are supposed to organized small blocks into larger ones by IA so as to prompt IA to locate the optimum quickly.

4.1.2 Greedy local strategy and delete-cross operator local strategy

The evolutionary mechanism of the immune algorithm aims to enhance the global performance of the population by keeping the good individuals, and further searching for optimization step by step. In this paper, greedy selection strategy is adopted in the process of immune evolution. In other words, filial generation of individuals will replace parent individuals if they are better enough after going through crossover or mutation and other gene operations; otherwise, the replacement will not be conducted. Besides, delete-cross operator is implemented using the solutions obtained from conducting immune operator in each generation. If they are better than parent individuals, the replacement will happen. The purpose is to ensure that the evolution direction of individuals and the search quality are satisfactory, search precision can be improved and the errors reduced in random operations.

4.1.3 Inver-over mutation operator based on dynamic mutation probability

Dynamic mutation operator
In order to accelerate the convergence of the algorithm, the following formula is used to dynamically change the mutation probability p.

$$p < - - - p \times \left(1 - \frac{It}{It_m} \times 0.01\right) \tag{7}$$

High-frequency mutation operator
For TSP problem, Inver-over mutation operator is adopted for the mutation in this paper. The experiment has proved that the mutation operator (Michalewicz 2000) is superior to traditional genetic operators such as partial mapped crossover, order crossover and cycle crossover (Larranaga et al. 1999), etc.

4.2 Procedures of hybrid immune algorithm

The detailed steps for realizing immune algorithm are as follows:

Algorithm 2 Hybrid immune algorithm

Input: $L, D, P, N, n, P_c, P_m, G$
Output: b

1: **Step 1**: Extract antigens according to TSP, that is, vaccine is extracted according to the objective function form and constraint conditions of the TSP problem. Algorithm parameters, such as population size, maximum iterations, crossover probability and mutation probability, should be set in the first place.
2: **Step 2**: Initialize the population by the greedy algorithm.
3: Step 2.1: Greedy crossover operator: a certain number of antibodies are selected based on the adaptability value and antibody selection probability determined by antibody concentration. Then randomly select two individuals from these individuals. The crossover positions are controlled by the crossover probability Pc. The genes of crossover positions are operated with crossover. Compare the new individuals with old ones, and use Greedy strategy for solution replacement.
4: Step 2.2: Greedy mutation operator: $rand$ is produced for antibodies with the crossover operation. When the dynamic mutation probability $P_m > rand$, Inver-over mutation operation is conducted to generate new individuals. Compare the new individuals with old ones, and use Greedy strategy for solution replacement.
5: Step 2.3: Vaccination operator: previously extracted vaccines are used for the vaccination of the selected antibodies. In other words, the values of the corresponding gene positions of the antibodies are changed based on the relevant gene positions in the vaccines.
6: **Step 3**: Calculate the adaptability of each antibody in the population.
7: **Step 4**: Immune selection.
8: Step 4.1: Immune detection operator: compare the adaptability values of two antibodies before and after vaccination. If the antibodies after vaccination are not as excellent as parent antibodies, then choose the latter to replace the antibodies after vaccination to participate the population selection. Antibody concentration shall be calculated for individuals after immune detection.
9: Step 4.2: Immune balance algorithm: selection probability is determined based on the adaptability and concentration of antibodies. See the selection probability in the following formula.

$$p = \alpha g p_f + (1 - \alpha) g p_d \tag{8}$$

In the formula, p_f is the adaptability probability of antibodies and defined as the ratio between the adaptability of antibodies and the total adaptability ; p_d is the concentration probability of antibodies. The greater the antibody concentration , the immune balance operator will make the antibodies more restrained, whilst the lower the concentration, it will promote the performance of antibodies. α is the proportion index and determines the function degree of adaptability and concentration.
10: Step 4.3: Selection operator: the selection is conducted using some common selection methods such as roulette wheel and simulated annealing for new populations. In this paper, the roulette wheel mechanism is adopted.
11: **Step 5**: Generate new individuals after delete-cross operator is carried on each of the individuals obtained after immune selection. Compare the new individuals with old ones, and use Greedy strategy for solution replacement
12: **Step 6**: Search optimal individuals in the new population and record them.
13: **Step 7**: Judge whether the conditions permit stop or not, that is, the maximum iteration. If yes, then stop the circulation and export the optimum; otherwise, go back to **step 3** for iteration.

The flow chart of hybrid immune algorithm is demonstrated in Fig. 2.

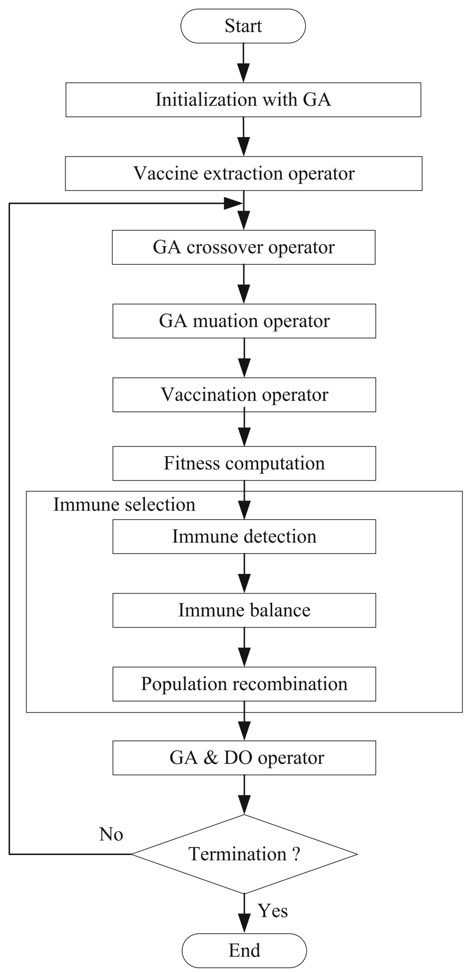

Fig. 2 Flowchart of HIA

5 Experiment and discussion

In order to better illustrate the performance of HIA, many sample instances are selected from the well-known TSPLIB test library for testing. Each test is run for 10 independent replications. The simulation environment is: Windows XP operating system, Intel Dual core 2.2 GHz CPU, 2 G memory and using the C language by software VC++ 6.0. The experimental condition is the same as that in literature Wang et al. (2011). The parameter settings of IA and HIA in the experiment are as follows: the population size of antibodies is 50; the memory library capacity is 10; the crossover probability is 0.6; the initial mutation probability $P_m = 0.78$; the conditions for algorithm stop vary due to different experimental purposes.

In Tables 1, 2 and 3, BIO stands for the basic Inver-Over operator memetic algorithm (Guo and Michalewicz 1998), IIO stands for the improved Inver-Over operator memetic algorithm (Wang et al. 2011), GSTM stands for the greedy sub-tour mutation (GSTM) algorithm (Albayrak and Allahverdi 2011), ILKMA stands for the InverCover & LKMA (removelocalsearch) memetic algorithm (Wang et al. 2011). In order to discuss the performance of the proposed HIA, we compared the HIA against the BIO, the IIO, the GSTM, the ILKMA and the IA. In this section, the benchmark set is composed of 14 examples from TSPLIB with the size ranging from 52 to 442. Table 1 provides the data of IIO and BIO, Table 2 for the data of GSTM and ILKMA, and Table 3 for the experimental results generated by HIA and IA.

Table 1 Error comparisons between improved invercover operator and basic invercover operator

Instance	IIO			BIO		
	Best (%)	Ave. (%)	Time (s)	Best (%)	Ave. (%)	Time (s)
berlin52	0.0000	0.0000	0.0664	0.0000	0.2307	0.0654
kroA100	0.0658	0.9496	0.0893	0.1927	1.7287	0.0874
pr144	0.7488	1.4835	0.0978	0.8695	2.5208	0.0969
ch150	1.1949	2.3468	0.1163	2.3591	3.6581	0.1138
kroB150	2.3995	4.5790	0.1152	3.7849	5.3142	0.1109
pr152	1.4101	3.1746	0.1026	3.1785	3.9318	0.0997
rat195	3.4438	5.0065	0.1199	3.9173	5.2734	0.1187
d198	3.3714	5.2700	0.1392	4.4867	6.5868	0.1341
kroA200	2.5129	4.7034	0.1386	3.3983	6.2303	0.1343
ts225	1.3487	2.4773	0.1424	1.4261	3.3603	0.1390
pr226	1.2219	2.4013	0.1429	2.0481	3.0035	0.1391
pr229	8.3895	10.7337	0.1802	9.7197	10.9705	0.1780
lin318	6.9476	9.7992	0.1976	8.9248	10.9648	0.1935
pcb442	8.5431	11.4006	0.2543	12.2868	13.5251	0.2526
Average	2.9710	4.5947	0.1359	4.0423	5.5214	0.1331

Table 2 Error comparisons between ILKMA and GSTM on 14 TSPLIB instances

Instance	ILKMA			GSTM		
	Best (%)	Ave. (%)	Time (s)	Best (%)	Ave. (%)	Time (s)
berlin52	0.0000	0.0000	0.4867	0.0000	0.0000	0.8360
kroA100	0.0000	0.0000	0.6193	0.0000	1.1836	6.9870
pr144	0.0564	0.1350	0.6878	0.0000	1.0809	13.5980
ch150	0.0000	0.3585	0.8566	0.4596	0.6357	11.2400
kroB150	0.0421	0.6456	0.7829	0.9644	1.7616	11.6840
pr152	0.0000	0.1259	0.7144	0.7695	1.6202	7.9370
rat195	0.4305	0.6586	0.8503	0.6027	1.8425	15.0500
d198	0.3359	0.6800	0.9391	0.3866	1.2193	12.0960
kroA200	0.4052	0.5816	0.9047	0.8683	1.5432	13.2920
ts225	0.0000	0.4850	0.9815	0.2527	0.4994	11.5590
pr226	0.1344	0.4270	0.9206	0.7242	1.5287	13.8430
pr229	0.6661	2.3401	1.1949	1.2326	2.9169	17.4240
lin318	1.3610	2.3058	1.2731	0.9827	3.3099	14.6430
pcb442	1.4494	2.1127	1.6723	2.0501	2.7758	19.1320
Average	0.3486	0.7754	0.9203	0.6638	1.5655	12.0940

Table 3 Error comparisons between HIA and BIA on 14 TSPLIB instances

Instance	HIA			BIA		
	Best (%)	Ave. (%)	Time (s)	Best (%)	Ave. (%)	Time (s)
berlin52	0.0000	0.0000	0.0399	0.0000	0.0047	0.7245
kroA100	0.0000	0.0000	0.0724	0.0012	1.2485	5.8994
pr144	0.0714	0.1284	0.0856	0.0041	1.3456	13.7889
ch150	0.0254	0.3218	0.1034	0.3142	0.9872	12.4040
kroB150	0.0345	0.5412	0.1041	0.9864	1.8324	11.5234
pr152	0.0000	0.0923	0.0922	0.7818	1.7325	7.2481
rat195	0.3246	0.5989	0.1094	0.6235	1.9024	12.3478
d198	0.2984	0.6543	0.1301	0.4034	1.1198	13.8404
kroA200	0.3822	0.5782	0.1314	0.8818	1.4524	14.4024
ts225	0.0000	0.4699	0.1385	0.3495	0.5013	12.8948
pr226	0.1026	0.4079	0.1389	0.7880	1.4781	14.7818
pr229	0.5981	2.1985	0.1687	1.3402	2.8045	18.2436
lin318	1.1973	2.2844	0.1881	1.0081	3.1182	15.7049
pcb442	1.3488	2.0286	0.2638	2.5244	2.7205	20.1232
Average	0.3131	0.7360	0.1262	0.7148	1.5892	12.4234

We can see that the computation precision of HIA is much higher than that of BIO, IIO, ILKMA, GSTM and IA. In Tables 1, 2 and 3, where, Best denotes the best value of the error (%) of the solution to the optimum solution, Ave.(%) denotes the average value of the errors of 10 independent runs to the optimum solution and time (s) denotes the CPU time which is the average time of 10 independent runs. We can see from the three tables that: for the average value of the Average Error, HIA is more efficient and effective than BIO, IIO, ILKMA, GSTM and IA; in terms of the average value of the Best Error, the performance of HIA is better than that of BIO, IIO, GSTM and IA, but the average value of the Best Error of HIA (0.07148 %) is higher than that of ILKMA (0.0564 %) for solving pr144; in terms of the average value of the Average Time, HIA is smaller than that of BIO, ILKMA, GSTM and IA, but the average value of the Average Time

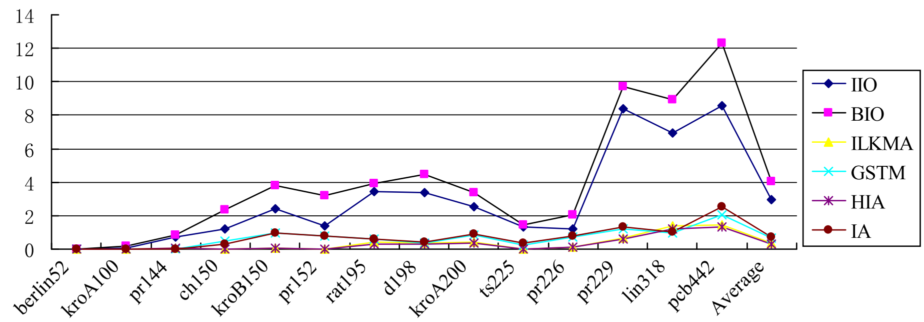

Fig. 3 Best error for the 14 TSP instances

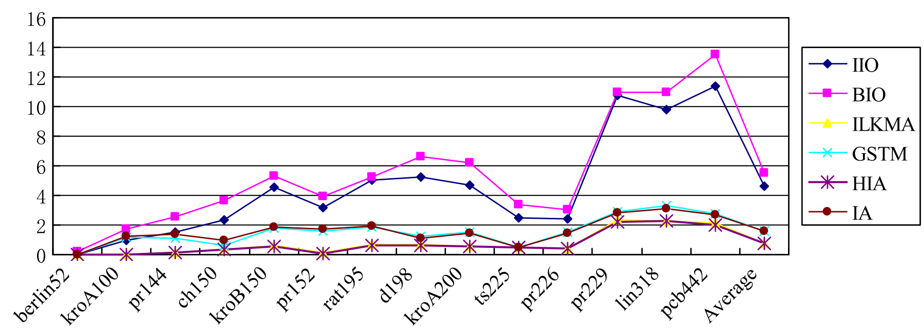

Fig. 4 Average error for the 14 TSP instances

of HIA (0.2638 s) is bigger than that of IIO (0.2543 s) for solving pcb442. Figure 3 shows that HIA has competitive advantages over BIO, IIO, ILKMA, GSTM and IA in terms of the errors which occur in the 10 independent runs. Figure 4 shows that HIA has competitive advantages over BIO, IIO, ILKMA, GSTM and IA in terms of average errors of the average values of 10 independent runs.

6 Conclusion

When a basic immune algorithm is used to solve TSP problems, the antibody population is randomly initialized, which might lead to many infeasible solutions. As a consequence, iterations of the algorithm have to be increased to find an acceptable solution. The whole process is time-consuming and the search precision is yet to be improved. Hence, in this paper, we adopt the greedy algorithm to initialize antibodies to obtain local optimums, 80 % of which are the edges of the global optimum. In addition, the intersection of multiple local optimal paths contains more edges of the global optimum. In the immune algorithm, the global search is conducted on antibodies first obtained through greedy algorithm. Greedy crossover operator and greedy dynamic mutation operator are applied so that "the survival of the fittest" can be realized. The delete-cross strategy will be applied to the antibodies in each new generation. This can delete the crossover paths in the travel route and effectively accelerate the convergence of the algorithm. The TSP experiment results prove that the hybrid immune algorithm has reliable and fast global convergence, which can effectively enhance the search ability of the immune algorithm. It has to be noted, however, whether the algorithm can be universally applied to other NP optimization problems needs to be examined in future studies.

Acknowledgments We are very grateful to the anonymous reviewers and the editors for their comments. The research was partially funded by the Key Program of National Natural Science Foundation of China (Grant Nos. 61133005, 61432005), and the National Natural Science Foundation of China (Grant Nos. 61370095, 61472124), the Ph.D. Programs Foundation of Ministry of Education of China (20100161110019), the Research Foundation of Education Bureau of Hunan Province (No. 13C333), the Project Supported by the Science and Technology Research Foundation of Hunan Province (Grant No. 2014GK3043).

References

Albayrak M, Allahverdi N (2011) Development a new mutation operator to solve the traveling salesman problem by aid of genetic algorithms. Expert Syst Appl 38(3):1313–1320

Alkaya AF, Duman E (2013) Application of sequence-dependent traveling salesman problem in printed circuit board assembly. IEEE Trans Compon Packag Manuf Technol 3(6):1063–1076

An HC, Kleinberg R, Shmoys DB (2012) Improving christofides' algorithm for the st path TSP. In: Proceedings of the 44th symposium on theory of computing, pp 875–886

Badillo AR, Ruiz JJ, Cotta C, Fernández-Leiva AJ (2013) On user-centric memetic algorithms. Soft Comput 17(2):285–300

Beheshti Z, Shamsuddin SM, Yuhaniz SS (2013) Binary Accelerated Particle Swarm Algorithm (BAPSA) for discrete optimization problems. J Global Optim 57(2):549–573

Carrabs F, Cerulli R, Speranza MG (2013) A branch-and-bound algorithm for the double travelling salesman problem with two stacks. Networks 61(1):58–75

Cecilia JM, García JM, Nisbet A, Amos M, Ujaldón M (2013) Enhancing data parallelism for ant colony optimization on gpus. J Parallel Distrib Comput 73(1):42–51

Chen J, Ding Y, Jin Y, Hao K (2013) A synergetic immune clonal selection algorithm based multi-objective optimization method for carbon fiber drawing process. Fibers Polym 14(10):1722–1730

Cheong T, White CC (2012) Dynamic traveling salesman problem: value of real-time traffic information. IEEE Trans Intell Transp Syst 13(2):619–630

Chun JS (1997) Shape optimization of electro-magnetic devices using immune algorithm. IEEE Trans Magn 33:1876–1879

Clarke G, Wright JW (1964) Scheduling of vehicles from a central depot to a number of delivery points. Oper Res 12(4):568–581

Dasgupta D, Forrest S (1999) Artificial immune systems in industrial applications. In: Proceedings of the 2nd international conference on intelligent processing and manufacturing of materials. IEEE Press, Hawaii, pp 257–267

De Castro LN, Von Zuben FJ (2002) Learning and optimization using the clonal selection principle. IEEE Trans Evol Comput 6(3):239–251

Deng W, Chen R, He B, Liu Y, Yin L, Guo J (2012) A novel two-stage hybrid swarm intelligence optimization algorithm and application. Soft Comput 16(10):1707–1722

Ding Y, Wang Z, Ye H (2012) Optimal control of a fractional-order HIV-immune system with memory. IEEE Trans Control Syst Technol 20(3):763–769

Gan R, Guo Q, Chang H, Yi Y (2010) Improved ant colony optimization algorithm for the traveling salesman problems. J Syst Eng Electron 21(2):329–333

Guo T, Michalewicz Z (1998) Inver-over operator for the TSP. In: Proceedings of the 5th parallel problem solving from nature. Lecture notes in computer science. Springer, Amsterdam, pp 803–812

Hassin R, Keinan A (2008) Greedy heuristics with regret, with application to the cheapest insertion algorithm for the TSP. Oper Res Lett 36(2):243–246

Hunt JE, Cooke DE (1995) An adaptive, distributed learning system based on immune system. In: IEEE international conference on system, man and cybernetics. IEEE Press, Vancouver, pp 2494–2499

Hurkens CA, Woeginger GJ (2004) On the nearest neighbor rule for the traveling salesman problem. Oper Res Lett 32(1):1–4

IEEE801.11 Working Group (2013). http://grouper.ieee.org/groups/802/1/index.html

Kalender M, Kheiri A, Özcan E, Burke EK (2013) A greedy gradient-simulated annealing selection hyper-heuristic. Soft Comput 17(12):2279–2292

Karapetyan D, Gutin G (2011) Lin–Kernighan heuristic adaptations for the generalized traveling salesman problem. Eur J Oper Res 208(3):221–232

Kıran MS, ¡şcan H, Gündüz M (2013) The analysis of discrete artificial bee colony algorithm with neighborhood operator on traveling salesman problem. Neural Comput Appl 23(1):9–21

Larranaga P, Kuijpers CMH, Murga RH, Inza I, Dizdarevic S (1999) Genetic algorithms for the traveling salesman problem: a review of representations and operators. Artif Intell Rev 13:129–170

Le Ny J, Feron E, Frazzoli E (2012) On the Dubins traveling salesman problem. IEEE Trans Autom Control 57(1):265–270

Marinakis Y, Marinaki M, Dounias G (2011) Honey bees mating optimization algorithm for the Euclidean traveling salesman problem. Inf Sci 181(20):4684–4698

Mavrovouniotis M, Yang S (2013) Ant colony optimization with immigrants schemes for the dynamic travelling salesman problem with traffic factors. Appl Soft Comput 13(10):4023–4037

Michalewicz Z (2000) How to Solve It: Modern Heuristick. Springer, Berlin

Montiel O, Diaz-Delgadillo FJ, Seplveda R (2013) Combinatorial complexity problem reduction by the use of artificial vaccines. Expert Syst Appl 40(5):1871–1879

Mora AM, García-Sánchez P, Merelo JJ, Castillo PA (2013) Pareto-based multi-colony multi-objective ant colony optimization algorithms: an island model proposal. Soft Comput 17(7):1175–1207

Nagata Y, Soler D (2012) A new genetic algorithm for the asymmetric traveling salesman problem. Expert Syst Appl 39(10):8947–8953

Ouaarab A, Ahiod B, Yang XS (2013) Discrete cuckoo search algorithm for the travelling salesman problem. Neural Comput Appl. doi:10.1007/s00521-013-1402-2

Pedro O, Saldanha R, Camargo R (2013) A tabu search approach for the prize collecting traveling salesman problem. Electron Notes Discret Math 41:261–268

Shi XH, Liang YC, Lee HP, Lu C, Wang QX (2007) Particle swarm optimization-based algorithms for TSP and generalized TSP. Inf Process Lett 103(5):169–176

Shim VA, Tan KC, Cheong CY (2012) A hybrid estimation of distribution algorithm with decomposition for solving the multiobjective multiple traveling salesman problem. IEEE Trans Syst Man Cybern Part C Appl Rev 42(5):682–691

Wang Y, Li J, Gao K, Pan Q (2011) Memetic algorithm based on improved inver-over operator and Lin–Kernighan local search for the Euclidean traveling salesman problem. Comput Math Appl 62(7):2743–2754

Xie X-F, Liu J (2009) Multiagent optimization system for solving the traveling salesman problem (TSP). IEEE Trans Syst Man Cybern Part B Cybern 39(2):489–502

Xu Y, Li K, Hu J, Li K (2014) A genetic algorithm for task scheduling on heterogeneous computing systems using multiple priority queues. Inf Sci 270:255–287

Yang G, Yi J (2013) Dynamic characteristic of a multiple chaotic neural network and its application. Soft Comput 17(5):783–792

Yuan S, Skinner B, Huang S, Liu D (2013) A new crossover approach for solving the multiple travelling salesmen problem using genetic algorithms. Eur J Oper Res 228(1):72–82

Zhang Z, Qian S (2011) Artificial immune system in dynamic environments solving time-varying non-linear constrained multi-objective problems. Soft Comput 15(7):1333–1349

Zhang Z, Yue S, Liao M, Long F (2014) Danger theory based artificial immune system solving dynamic constrained single-objective optimization. Soft Comput 18(1):1–22