Applied Soft Computing 13 (2013) 1365–1375

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

A model induced max-min ant colony optimization for asymmetric traveling salesman problem

Jie Bai[a], Gen-Ke Yang[a], Yu-Wang Chen[b], Li-Sheng Hu[a], Chang-Chun Pan[a,*]

[a] *Department of Automation, and Key Laboratory of System Control and Information Processing, Ministry of Education of China, Shanghai Jiao Tong University, Shanghai 200240, China*
[b] *Decision and Cognitive Sciences Research Centre, MBS, The University of Manchester, Manchester M15 6PB, UK*

ARTICLE INFO

Article history:
Received 31 December 2011
Received in revised form 28 March 2012
Accepted 2 April 2012
Available online 3 May 2012

Keywords:
Hybrid evolutionary algorithm
Ant colony optimization
ATSP
Model induced

ABSTRACT

A large number of hybrid metaheuristics for asymmetric traveling salesman problem (ATSP) have been proposed in the past decades which produced better solutions by exploiting the complementary characteristics of different optimization strategies. However, most of the hybridizations are criticized due to lacking of sufficient analytical basis. In this paper, a model induced max-min ant colony optimization (MIMM-ACO) is proposed to bridge the gap between hybridizations and theoretical analysis. The proposed method exploits analytical knowledge from both the ATSP model and the dynamics of ACO guiding the behavior of ants which forms the theoretical basis for the hybridization. The contribution of this paper mainly includes three supporting propositions that lead to two improvements in comparison with classical max-min ACO optimization (MM-ACO): (1) Adjusted transition probabilities are developed by replacing the static biased weighting factors with the dynamic ones which are determined by the partial solution that ant has constructed. As a byproduct, nonoptimal arcs will be indentified and excluded from further consideration based on the dual information derived from solving the associated assignment problem (AP). (2) A terminal condition is determined analytically based on the state of pheromone matrix structure rather than intuitively as in most traditional hybrid metaheuristics. Apart from the theoretical analysis, we experimentally show that the proposed algorithm exhibits more powerful searching ability than classical MM-ACO and outperforms state of art hybrid metaheuristics.

1. Introduction

Asymmetric traveling salesman problem (ATSP) is one of a class of difficult problems in combinatorial optimization that is representative of a large number of scientific and engineering problems. ATSP and its variants are commonly used models for formulating many practical applications in manufacturing scheduling problem. For example, the scheduling problem in a discrete manufacturing is mainly concerned with how to determine the sequence of jobs so as to minimize the total set-up cost. This problem can be easily formulated into an ATSP problem. Considerable industrial applications on the basis of ATSP can be found in [1–4]. Hence, ATSP has always been one of the most attractive problems in academic community. Before the early 1990s, exact algorithms, form the main stream of solvers. However, solving ATSP optimally is NP-hard and the exact algorithm may be difficult to produce a provably optimal solution in a reasonable time. Metaheuristics have found wide acceptance in the arena where suboptimal or satisfied solutions are expected to be generated in a given time period. Among those metaheuristics ant colony optimization (ACO) which was proposed by Dorigo and Gambardella [5], is considered to be one of the most representative ones [6]. It is an iterative approach in which a number of artificial ants construct solutions randomly but are guided by pheromone information that stems from former ants building good solutions. Blum and Dorigo [7] presented a hyper-cube ACO by introducing a normalized way for the pheromone value by which the pheromone values were limited in the interval [0,1]. A survey was given of recent applications and variants of ACO methods by Dorigo and Blum [8].

Recently, many hybrid algorithms that combine ACO and other optimization algorithms have received more and more attention. These hybrid algorithms can produce better solutions by exploiting the complementary characteristics of different optimization strategies. Roughly speaking, ACO based hybrid algorithms fall into two categories. This first category is the hybrid optimization that combines local heuristic search, such as 2-OPT [9] with ACO algorithm. Cheng and Mao [10] developed ACS-TSPTW to solve TSP problem with time window where two local heuristics were embedded to manage the time window constraints. A hybrid ACO algorithm combined with a mutation and 2-OPT heuristic for generalized TSP was

* Corresponding author.
E-mail addresses: pan_cc@sjtu.edu.cn, panchangchun@gmail.com (C.-C. Pan).

1568-4946/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.asoc.2012.04.008

proposed by Yang et al. [11]. A web-based simulation and analysis software based on ACO for TSP was developed by Aybars et al. [12]. Puris et al. [13] presented a two-stage ACO in order to obtain good exploration of the search space. ACO optimization applied to multiple TSP problem can be found in Ghafurian and Javadian [14]. Chen and Chien [15] proposed a hybrid algorithm with a combination of ACO algorithm, Simulated Annealing (SA), Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) for solving TSP. Dong and Guo [16] developed a cooperative ACO & GA algorithm in which mutual information exchange between ACO and GA helps conduct the selection of the best solutions for next iteration. ACO hybridized with heuristic rules was also investigated by Keskinturk et al. [4] to solve sequence-dependent setup parallel machine scheduling problem.

The second category is concerned with the combination of exact methods and ACO algorithm. A hybrid algorithm called Approximate Nondeterministic Tree Search (ANTS) is the first to integrate branch-and-bound techniques into ACO for quadratic assignment problem (QAP) [17]. According to our investigation, algorithms for ATSP problem that combine ACO and exact methods are very scarce. However, there are a number of schemes that hybridize metaheuristics other than ACO with exact methods. For example, Choi et al [18] developed a hybrid algorithm for ATSP problem that embedded an integer programming solver into GA. Cowling and Keuthen [19] employed a decomposition-recombination scheme for TSP problem in which the original problem is first decomposed into small subproblems. These subproblems are then solved using exact algorithms and the solutions are re-embedded into the original problem. Note that most of the hybrids between exact methods and metaheuristics are operated in a heuristic way. Although the hybrid algorithms can produce high performance as mentioned by the references above, these methods are somewhat unreliable because advantages coming with the hybridization are mostly demonstrated by means of experimental study. As pointed out by the “no free lunch theorems for optimization” [20], any elevated performance over one class of problems is offset by lower performance over another class. In other words, the hybridization in a heuristic manner may not always produce better solutions. Blum et al. [21] and Jourdan et al. [22] surveyed and categorized current hybrid algorithms. Their statistical results indicate that more efforts are needed to design hybrids of exact methods and metaheuristics in a systematic and cooperative way in terms of analytical results.

This paper aims to develop a well-suited hybrid algorithm for ATSP problem in which analytical results can be utilized and embedded into ACO algorithm. The information obtained by analyzing both the ATSP model and the dynamics of ACO algorithm itself is used to guide the search of ants in one of the most powerful variant of ACO for TSP problem, i.e., max-min ACO algorithm [23]. Hence, we name the method model induced max-min ant colony optimization (MIMM-ACO). Specifically, our contributions lie in two aspects:

- Adjusted transition probabilities are developed by replacing the static biased weighting factors with the dynamic ones. The dynamic weighting factor is closely dependent on the partial solution that ant has constructed. The ideal behind it is that we favor the choice of edges with small residual cost instead of with the small actual cost. As a byproduct nonoptimal arcs will be indentified at each step of tour construction using the dual information derived from solving the associated assignment problem (AP) and these arcs will be discarded from future consideration.
- A terminal condition is determined analytically based on the state of pheromone matrix structure. The result comes with a necessary condition for obtaining one optimal solution.

The rest of the paper is organized as follows. In the next section, the relevant background contents about the ATSP formulation and the MM-ACO optimization are briefly reviewed. Section 3 is dedicated to the design of MIMM-ACO algorithm in which several supporting analytical results are presented. Section 4 presents some computational results with which we show that the proposed algorithm has a remarkable performance, in particular on the running-time efficiency compared with several state of the art algorithms. Section 5 offers a summary and outlines future work.

2. Preliminaries

Given a directed graph $\boldsymbol{G}=(\boldsymbol{V}, \boldsymbol{A})$ with vertex/city set $\boldsymbol{V}\overset{\text{def}}{=}\{1, 2, \ldots, n\}$, arc set $\boldsymbol{A}\overset{\text{def}}{=}\{(i, j)|i, j = 1, 2, \ldots, n\}$ and cost $c_{i,j}$ associated with each arc (i,j). If $c_{i,j} = c_{j,i}$ for all $(i,j)\in\boldsymbol{A}$ then the TSP is symmetric, otherwise it is asymmetric (ATSP). Formally, ATSP may be stated as an integer programming (IP) of the following form.

We first explain the notations before present in the IP.

Indices: $i,j\in V$ indicate the vertex;
Parameters: $c_{i,j}$, $(i,j)\in\boldsymbol{A}$ indicates travel cost of arc (i,j)
Decision variables: $x_{i,j}\in\{\mathbf{0},\mathbf{1}\}$, $(i,j)\in\boldsymbol{A}$

$$\text{Objective}: \quad Z^* = \min\sum_{i\in\boldsymbol{V}}\sum_{j\in\boldsymbol{V}} c_{i,j}x_{ij} \tag{1}$$

s.t.

$$\begin{aligned}&\sum_{i\in\boldsymbol{V}} x_{i,j} = 1, \quad j\in\boldsymbol{V}\\ &\sum_{j\in\boldsymbol{V}} x_{i,j} = 1, \quad i\in\boldsymbol{V}\end{aligned} \tag{2}$$

$$\sum_{i,j\in\boldsymbol{S}} x_{i,j} \le |\boldsymbol{S}| - 1, \qquad \forall\boldsymbol{S}\subset V, \quad \boldsymbol{S}\ne\varnothing \tag{3}$$

$$x_{i,j}\in\{0, 1\}, \quad i, j\in\boldsymbol{V} \tag{4}$$

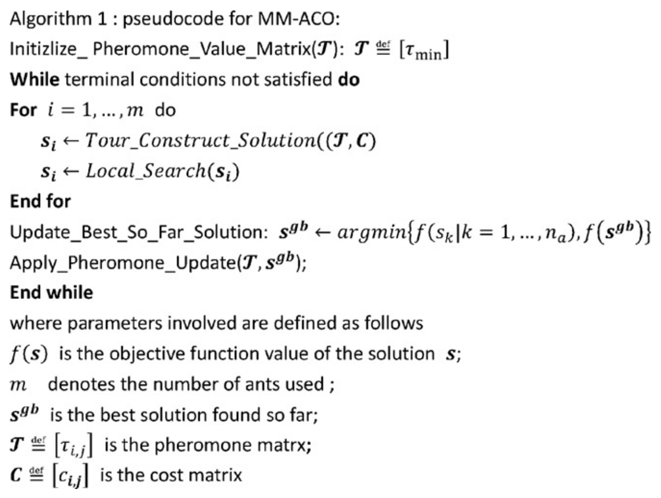
where if arc (i,j) is present in a solution, travel occurs from vertex i to j, $x_{i,j} = 1$; otherwise $x_{i,j} = 0$. The ATSP involves specifying a minimum-cost tour that visits each vertex once and returns the starting one, that is, a Hamiltonian cycle. The objective in (1) is to minimize the total travel cost. Constraints (2) ensure that each vertex is visited only once. Constraints (3) are used to eliminate subtours. Constraints (4) are binary restriction for decision variables.

For attacking ATSP problem, a number of algorithms have been developed just as that mentioned in Section 1. The classical MM-ACO by Stutzle and Hoos [23] is introduced briefly in order to derive our method smoothly. The pseudocode is given in Fig. 1.

Tour_Construct_solution $(\mathcal{T}, \boldsymbol{C})$. In MM-ACO algorithm artificial ants build solutions in terms of the current pheromone matrix $\mathcal{T}$ and the cost matrix $\boldsymbol{C}$. In the construction phrase an ant incrementally constructs a partial solution by adding an unvisited city to the partial solution constructed so far until a feasible solution is obtained. Let $\boldsymbol{s^p}(r)$ denote the partial tour with city r being the last visited city. The choice of the next city to be added is given by the following rule

$$k = \begin{cases} arg\ max_{u\in\boldsymbol{J}(\boldsymbol{s^p}(r))}\{(\tau_{r,u})(\eta_{r,u})^{\beta}\} & if\ q\le q_0\\ using transition probabilities given by & (6)\end{cases} \tag{5}$$

where $\boldsymbol{J}(\boldsymbol{s^p}(r))$ represents the set of cities that the ant positioned at city r is allowed to add to the current partial tour; $\tau_{r,u}$ the pheromone level on arc (r,u); $\eta_{r,u}$ the static biased weight for the choice of the arc (r,u), usually set $\eta_{r,u} = 1/c_{r,u}$; q a random number



```
Algorithm 1 : pseudocode for MM-ACO:
Initizlize_ Pheromone_Value_Matrix($\mathcal{T}$): $\mathcal{T} \overset{\text{def}}{\equiv} [\tau_{\min}]$
While terminal conditions not satisfied do
For $i = 1, \ldots, m$ do
    $s_i \leftarrow Tour\_Construct\_Solution((\mathcal{T}, C)$
    $s_i \leftarrow Local\_Search(s_i)$
End for
Update_Best_So_Far_Solution: $s^{gb} \leftarrow argmin\{f(s_k|k = 1, \ldots, n_a), f(s^{gb})\}$
Apply_Pheromone_Update($\mathcal{T}, s^{gb}$);
End while
where parameters involved are defined as follows
$f(s)$ is the objective function value of the solution $s$;
$m$ denotes the number of ants used ;
$s^{gb}$ is the best solution found so far;
$\mathcal{T} \overset{\text{def}}{\equiv} [\tau_{i,j}]$ is the pheromone matrx;
$C \overset{\text{def}}{\equiv} [c_{i,j}]$ is the cost matrix
```

Fig. 1. The pseudocode of traditional MM-ACO.

uniformly distributed in [0,1]; and q_0 are β are two parameters with $(0 \le q_0 \le 1)$,

The transition probabilities is defined by

$$p(k/\boldsymbol{s^p}(r)) = \begin{cases} \dfrac{(\tau_{r,k})(\eta_{r,k})^{\beta}}{\sum_{u \in \boldsymbol{J}(\boldsymbol{s^p}(r))} (\tau_{r,u})(\eta_{r,u})^{\beta}} & \text{if } k \in \boldsymbol{J}(\boldsymbol{s^p}(r)) \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Apply_Pheromone_Update ($\mathcal{T}$, $\boldsymbol{s^{gb}}$). The global best offline pheromone update is used as that

$$\begin{gathered} \forall(i,j) : \tau_{i,j} \leftarrow (1-\rho)\tau_{i,j} \\ \forall(i,j) \in \boldsymbol{s^{gb}} : \tau_{i,j} \leftarrow \tau_{i,j} + g(\boldsymbol{s^{gb}}) \\ \forall(i,j) : \tau_{i,j} = \max\{\tau_{\min}, \tau_{i,j}\} \end{gathered} \tag{7}$$

where ρ,$0 < \rho < 1$ is the evaporation rate and $g(\boldsymbol{s})$,$0 < g(\boldsymbol{s}) < +\propto$ is the prize function with

$$f(\boldsymbol{s}) < f(\boldsymbol{s}') \Rightarrow g(\boldsymbol{s}) \ge g(\boldsymbol{s}') \tag{8}$$

In ACO algorithm the prize function will determine the maximum pheromone level τ_{max}. More specifically, Stutzle and Dorigo [24] has proven that the maximum pheromone value is bounded asymptotically by

$$\tau_{\max} = \frac{1}{\rho} g(\boldsymbol{s}^*) \tag{9}$$

3. Methodology

In this section we will present the proposed strategies and related theoretical results and then develop the overall implementation of MIMM-ACO. Our work primarily consists of the following aspects.

3.1. Transition probabilities of MIMM-ACO

In every ant based search procedure the transition probabilities are of vital importance. Our intention is to promote the searching activity of ants by embedding the domain or model knowledge into construction of more reasonable transition probabilities. The revision of (6) is based on the following proposition.

Proposition 1. Given a partial path $\boldsymbol{s^p} \overset{\text{def}}{=} \{i_1, i_2, \ldots, i_p\}$ and a feasible solution $\boldsymbol{s}$, if $f(\boldsymbol{s}) - \tilde{Z}^*_{AP} \le \sum_{(i,j) \in \boldsymbol{s^p}} \bar{c}_{i,j}$ is satisfied then the following inequality holds

$$f(\boldsymbol{s}) \le Z^*(\boldsymbol{s^p}) \tag{10}$$

where $\tilde{Z}^*_{AP}$ is the optimal solution value of the assignment problem (AP) defined by (1), (2) and (4) which provides a lower bound of Z^*; $\bar{c}_{i,j}, \bar{c}_{i,j} \ge \mathbf{O}$ is the reduced cost of (i,j) associated with the resulting AP problem (see Appendix A for more details); $Z^*(\boldsymbol{s^p})$ denotes the optimal solution value of the ATSP that is subjected to containing $\boldsymbol{s^p}$ as a partial solution.

Proof. See Appendix A.

Proposition 1 means that any solution containing $\boldsymbol{s^p}$ cannot be better than $\boldsymbol{s}$ if the condition $f(\boldsymbol{s}) - \tilde{Z}^*_{AP} \le \sum_{(i,j) \in \boldsymbol{s^p}} \bar{c}_{i,j}$ holds. It provides a quantified criterion to indicate during the tour construction procedure whether the search alone the current path is desirable. In other words if a tour along the current partial path will lead to an inferior solution than the current best solution $\boldsymbol{s^{gb}}$, the march would be useless. Inspired from Proposition 1, we adjust transition probabilities by replacing the static biased weighting factor dependent on the actual cost with a dynamic one computed using the residual cost. Such that original transition probabilities are adjusted and become

$$p_{adj}(k/\boldsymbol{s^p}(r), \boldsymbol{s^p}) = \begin{cases} \dfrac{(\tau_{r,k})[w(\boldsymbol{s^p}(r), k)]^{\beta}}{\sum_{u \in \boldsymbol{J_{adj}}(\boldsymbol{s^p}(r))} (\tau_{r,u})[w(\boldsymbol{s^p}(r), k)]^{\beta}} & \text{if } k \in \boldsymbol{J_{adj}}(\boldsymbol{s^p}(r)) \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

where $w(\boldsymbol{s^p}(r),k)$ denotes a biased weighting factor which is dependent on $\boldsymbol{s^p}(r)$ and city k. Specifically, we define

$$\begin{gathered} w(\boldsymbol{s^p}(r), k) \overset{\text{def}}{=} \max\left\{1 - \frac{\sum_{(i,j) \in \boldsymbol{s^p}(k)} \bar{c}_{i,j}}{\Delta G}, w_{\min}\right\} \\ \boldsymbol{s^p}(k) \overset{\text{def}}{=} \boldsymbol{s^p}(r) \cup (r, k) \\ \Delta G \overset{\text{def}}{=} f(\boldsymbol{s^{gb}}) - \tilde{Z}^*_{AP} \end{gathered} \tag{12}$$

$$\boldsymbol{J_{adj}}(\boldsymbol{s^p}(r)) \overset{\text{def}}{=} \boldsymbol{J}(\boldsymbol{s^p}(r)) - \left\{k | \sum_{(i,j) \in \boldsymbol{s^p}(k)} \bar{c}_{i,j} \ge \Delta G\right\} \tag{13}$$

where w_{min} specifies the smallest biased weighting factor.

Remarks:

- ΔG is the current gap between the best upper bound and the lower bound. Without loss of generality, $\Delta G > \mathbf{O}$ is assumed since if $\Delta G = \mathbf{O}$ we have got the optimal solution already.
- The weighting factor defined in (12) indicates that ants favor choice of edges with small residual cost instead of small actual cost (see (6) for comparison).

The adjusted list of candidate cities defined in (13) shows that unvisited cites that will result in a inferior solution than $\boldsymbol{s^{gb}}$ must be left out of consideration. If many of such cites can be excluded, more attention would be centered on the search space that will produce solutions better than $\boldsymbol{s^{gb}}$ and the searching efficiency therein will be improved undoubtedly. Nevertheless, the number of excluding non-optimal cities is closely relevant to the tightness of the lower bound.

According to the newly generated transition probabilities, the choice of the next city to be added will be determined by

$$k = \begin{cases} arg\ max_{u \in \boldsymbol{J_{adj}}(\boldsymbol{s^p}(r))}\{(\tau_{r,u})[w(\boldsymbol{s^p}(r), u)]^{\beta}\} & if\ q \le q_0 \\ usingtransitionprobabilitiesgivenby & (11) \end{cases} \tag{14}$$

3.2. Terminal criterion setting of MIMM-ACO

The rule for terminal condition is derived from the proposition below.

Proposition 2. Let t^* be the first time when an optimal solution has been found s^*, then a constant value $t_0 = ln(\varphi)/ln(1-\rho)$ exists such that when $t \geq t^* + t_0$ the following holds

$$\tau_{i,j}(t) > \tau_{k,l}(t) \tag{15}$$

$$\begin{gathered}\tau_{k,l}(t) = \tau_{\min} \\ \forall (i,j) \in s^*, \quad \forall (k,l) \notin s^*\end{gathered} \tag{16}$$

where $\varphi = \tau_{min}/\tau_{max}$ is the ratio of minimum to maximum pheromone value

Proof. See Appendix B.

Proposition 2 shows a scenario that after a fixed number of iterations starting from the time when a optimal solution has been found, the pheromone matrix will keep a stable structure in which the pheromone level on the optimal solution is larger than that on any other connections and any connections not belonging to the optimal solution has the pheromone level with τ_{min}. Solutions generated from a stable pheromone structure will tend to be stable as well. Therefore, Proposition 2 provides a reasonable way to define a terminal condition for ACO, that is after a sequence of $\theta, \theta \geq t_0$ iterations without any improvement we terminate the ACO procedure. In our computational experiments, θ takes on the value of $(1.2–1.5)t_0$.

3.3. Convergence proof of MIMM-ACO

Corollary 1. *For an arbitrary choice of a small $\in > 0$ and for a sufficiently large t, it holds that $P^*(t) \geq 1 - \in$, and asymptotically $\mathbf{lim}_{t\to\propto} P^*(t) = \mathbf{1}$, where P^* is the probability that the algorithm finds an optimal solution at least once within the first t iterations.*

Proof. See Appendix C.

3.4. Overall algorithm of MIMM-ACO

We summarize the analysis above and present the overall pseudocode of MIMM-ACO in Fig. 2.

In MIMM-ACO method extra initialization procedures are included in contrast to conventional MM-ACO procedure. These procedures include

(1) the AP solver to solve the associated AP problem and compute the residual cost matrix;
(2) the so-called "PATCH" proposed by Karp [25] which was used to repair an AP solution to an ATSP solution. "PATCH" algorithm generates the first feasible solution $\boldsymbol{s_1}$.
(3) the calculation of t_0 using Proposition 2 and τ_{min} using its definition directly.

Specifically, we have

$$t_0 = \frac{\ln(\varphi)}{\ln(1-\rho)} \tag{17}$$

$$\tau_{\min} = \hat{\tau}_{\max}\varphi \tag{18}$$

where

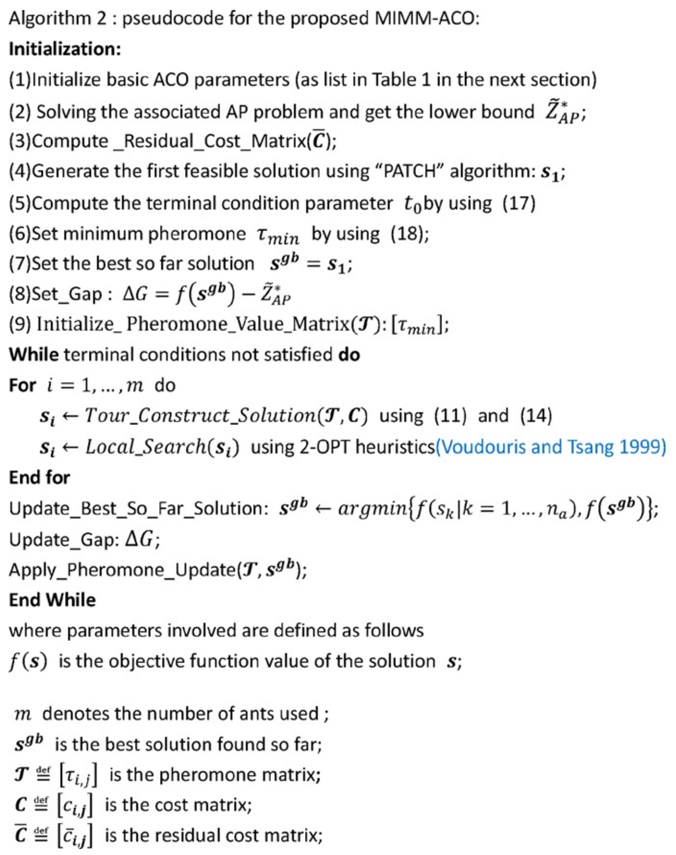
$$\hat{\tau}_{\max} = \frac{\hat{g}(\boldsymbol{s}^*)}{\rho} \tag{19}$$

$$\hat{g}(\boldsymbol{s}^*) \overset{\text{def}}{=} \frac{1}{\hat{f}(\boldsymbol{s}^*)} = \frac{1}{(1/2)f(\boldsymbol{s_1}) + (1/2)\tilde{Z}^*_{AP}} \tag{20}$$

φ and ρ are given parameters of ACO.

Note that $\hat{\tau}_{\max}$ in (18) denotes a estimate of τ_{max}. As shown by (9) it is hard to get the actual value of τ_{max} because the actual value of $g(s^*)$ is unavailable so far. So that, the estimate shown in (20) is used to give an approximation of $g(\boldsymbol{s}^*)$ that employs a weighted sum of lower bound and an upper bound.

The additional computational cost of MIMM-ACO compared with MM-ACO is incurred mostly from the AP solver and the "PATCH" algorithm. As is known to all, AP solver, for example the primal-dual algorithm [26], is of runtime complexity $O(n^3)$ and the same as the 'PATCH' algorithm. Since both the AP solver or the "PATCH" algorithm will be run once in the overall MIMM-ACO procedure, the computational cost is inexpensive. Our experiments also show that AP problem can be solved very quickly, even for the instance size up to 1000 the running time would be within several seconds.



```
Algorithm 2 : pseudocode for the proposed MIMM-ACO:
Initialization:
(1)Initialize basic ACO parameters (as list in Table 1 in the next section)
(2) Solving the associated AP problem and get the lower bound $\tilde{Z}^*_{AP}$;
(3)Compute _Residual_Cost_Matrix($\bar{\boldsymbol{C}}$);
(4)Generate the first feasible solution using "PATCH" algorithm: $\boldsymbol{s_1}$;
(5)Compute the terminal condition parameter $t_0$by using (17)
(6)Set minimum pheromone $\tau_{min}$ by using (18);
(7)Set the best so far solution $\boldsymbol{s^{gb}} = \boldsymbol{s_1}$;
(8)Set_Gap : $\Delta G = f(\boldsymbol{s^{gb}}) - \tilde{Z}^*_{AP}$
(9) Initialize_ Pheromone_Value_Matrix($\mathcal{T}$): $[\tau_{min}]$;
While terminal conditions not satisfied do
For $i = 1, \ldots, m$ do
    $\boldsymbol{s_i} \leftarrow Tour\_Construct\_Solution(\mathcal{T}, \boldsymbol{C})$ using (11) and (14)
    $\boldsymbol{s_i} \leftarrow Local\_Search(\boldsymbol{s_i})$ using 2-OPT heuristics(Voudouris and Tsang 1999)
End for
Update_Best_So_Far_Solution: $\boldsymbol{s^{gb}} \leftarrow argmin\{f(s_k|k = 1, \ldots, n_a), f(\boldsymbol{s^{gb}})\}$;
Update_Gap: $\Delta G$;
Apply_Pheromone_Update($\mathcal{T}, \boldsymbol{s^{gb}}$);
End While
where parameters involved are defined as follows
$f(\boldsymbol{s})$ is the objective function value of the solution $\boldsymbol{s}$;

$m$ denotes the number of ants used ;
$\boldsymbol{s^{gb}}$ is the best solution found so far;
$\mathcal{T} \overset{\text{def}}{=} [\tau_{i,j}]$ is the pheromone matrix;
$\boldsymbol{C} \overset{\text{def}}{=} [c_{i,j}]$ is the cost matrix;
$\bar{\boldsymbol{C}} \overset{\text{def}}{=} [\bar{c}_{i,j}]$ is the residual cost matrix;
```

Fig. 2. The pseudocode of the overall proposed MIMM-ACO algorithm for ATSP problem.

3.5. A numerical example

To increase the comprehensibility of the proposed definition of transition probabilities, a numerical example from Miller and Pekny [27] is used to show how the proposed method is conducted. The ATSP instance is with 8 vertices and the cost matrix is given by

$$[c_{i,j}] = \begin{bmatrix} \infty & 3 & 8 & 5 & 9 & 8 & 6 & 6 \\ 0 & \infty & 2 & 1 & 2 & 3 & 9 & 1 \\ 2 & 4 & \infty & 1 & 4 & 8 & 3 & 2 \\ 7 & 5 & 3 & \infty & 1 & 8 & 2 & 6 \\ 6 & 8 & 3 & 3 & \infty & 8 & 2 & 4 \\ 9 & 4 & 3 & 4 & 4 & \infty & 2 & 7 \\ 8 & 0 & 8 & 3 & 6 & 3 & \infty & 9 \\ 6 & 2 & 0 & 1 & 6 & 8 & 0 & \infty \end{bmatrix}$$

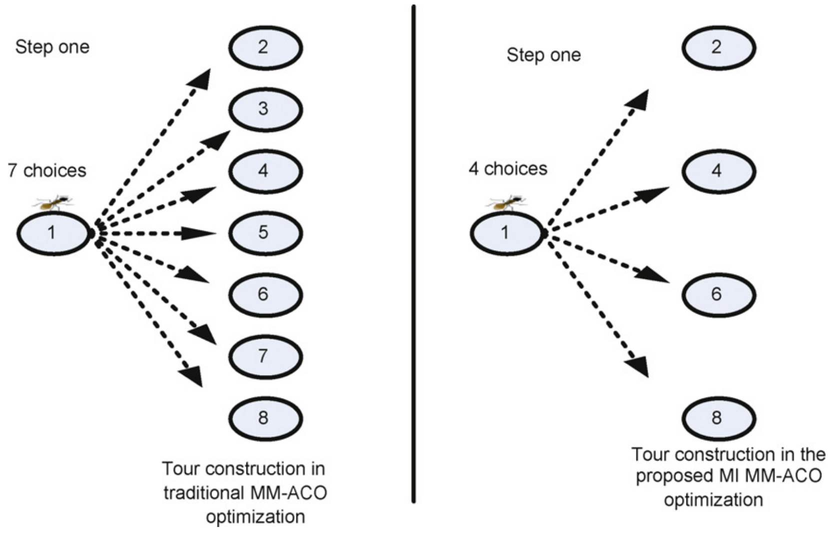

Step one
7 choices
1
2
3
4
5
6
7
8
Tour construction in traditional MM-ACO optimization
Step one
4 choices
1
2
4
6
8
Tour construction in the proposed MI MM-ACO optimization

Fig. 3. Comparison of the selection procedure for the next candidate vertex at step one between traditional MM-ACO (left side) and the proposed MIMM-ACO (right side).

First, the associated AP problem is solved and then we get that lower bound and the associated optimal dual variables (we refer to Dell'Amico and Toth [26] for more details):

$$\tilde{Z}^*_{AP} = 14$$

$$\begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \end{bmatrix} = \begin{bmatrix} 3 & 0 & 1 & 1 & 3 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & -1 & 1 \end{bmatrix}$$

Compute the residual cost matrix by $\bar{c}_{i,j} = c_{i,j} - u_i - v_j$, such that we have

$$[\bar{c}_{i,j}] = \begin{bmatrix} \infty & 0 & 5 & 2 & 6 & 2 & 4 & 2 \\ 0 & \infty & 2 & 1 & 2 & 0 & 10 & 0 \\ 1 & 3 & \infty & 0 & 3 & 4 & 3 & 0 \\ 6 & 4 & 2 & \infty & 0 & 4 & 2 & 4 \\ 3 & 5 & 0 & 0 & \infty & 2 & 0 & 0 \\ 6 & 1 & 0 & 1 & 1 & \infty & 0 & 3 \\ 8 & 0 & 8 & 3 & 6 & 0 & \infty & 8 \\ 6 & 2 & 0 & 1 & 6 & 5 & 1 & \infty \end{bmatrix}$$

We initialize $\boldsymbol{s^{gb}}$ = [1,2,4,5,8,7,6,3] obtained by using some certain heuristic algorithm. Note that $\boldsymbol{s^{gb}}$ can be updated in subsequent iterations. Obviously, we can get the currently best upper bound $f(\boldsymbol{s^{gb}})$ = 17 and the current gap between the lower bound and the upper bound $\Delta G = f(\boldsymbol{s}) - \tilde{Z}^*_{AP} = 3$. Assume that an ant is going to start to travel from vertex 1. At step one, we get immediately that:

$$\boldsymbol{s^p}(1) = \{1\};$$

$$\boldsymbol{J}(\boldsymbol{s^p}(1)) = \{2, 3, 4, 5, 6, 7, 8\};$$

Using (13) we get that

$$\left\{k|\textstyle\sum_{(i,j)\in \boldsymbol{s^p}(1)} \bar{c}_{i,j} \geq 3\right\} = \{3, 5, 7\}$$
$$\boldsymbol{J_{adj}}(\boldsymbol{s^p}(1)) = \boldsymbol{J}(\boldsymbol{s^p}(1)) \backslash \{3, 5, 7\} = \{2, 4, 6, 8\}$$

In traditional ACO, the next vertex will be selected from $\boldsymbol{J}(\boldsymbol{s^p}(1))$ while in the proposed MIMM-ACO algorithm the next vertex will be chosen from $\boldsymbol{J_{adj}}(\boldsymbol{s^p}(1))$. It is obvious that the size of $\boldsymbol{J_{adj}}(\boldsymbol{s^p}(1))$ is much smaller than that of $\boldsymbol{J}(\boldsymbol{s^p}(1))$ and this will definitely lead the ant to more promising tours. Fig. 3 presents a graphical illustration that in traditional MM-ACO the ant at vertex 1 will suffer from 7 choices while in MIMM-ACO the number of choices is 4.

Using (12) the biased weighting factors $w(\boldsymbol{s^p}(1),k)$ can be computed in turn:

$$w(\boldsymbol{s^p}(1), 2) = 1 - \frac{\sum_{(i,j)\in \boldsymbol{s^p}(2)} \bar{c}_{i,j}}{3} = 1$$

$$w(\boldsymbol{s^p}(1), 4) = 1 - \frac{\sum_{(i,j)\in \boldsymbol{s^p}(4)} \bar{c}_{i,j}}{3} = \frac{1}{3}$$

$$w(\boldsymbol{s^p}(1), 6) = 1 - \frac{\sum_{(i,j)\in \boldsymbol{s^p}(6)} \bar{c}_{i,j}}{3} = \frac{1}{3}$$

$$w(\boldsymbol{s^p}(1), 8) = 1 - \frac{\sum_{(i,j)\in \boldsymbol{s^p}(8)} \bar{c}_{i,j}}{3} = \frac{1}{3}$$

Using (14) the next city can be selected which completes the first step. The subsequent steps can be done in the same manner and the process is repeated until the overall tour construction procedure is finished completely.

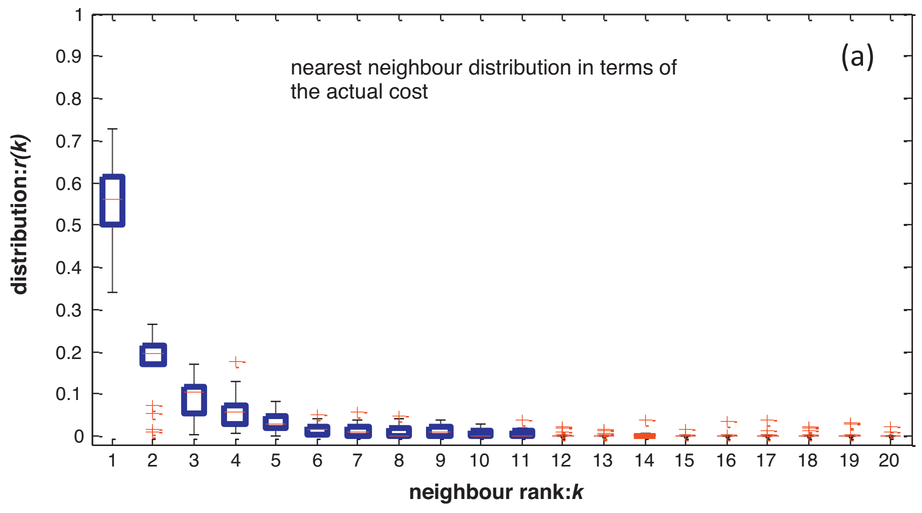
4. Computational results and discussions

The computational experiments consist of three parts, the first part aims to show the evidence that the residual cost is more significant than the actual cost for guiding the search of ants. This is conducted by means of the statistics on the nearest neighbor distribution (NND) of optimal solutions. The second one is to show the superiority of the MIMM-ACO compared with several existing state-of-the art algorithms, i.e., extremal optimization (EO) [28], cooperative genetic ant systems (CGAS) [16] and MM-ACO [23] based on benchmark problems. The third one is to do further experiments based on a number of randomly generated and large scale problems.

4.1. Experiments on nearest neighbor distribution

The NND has been studied for design of good TSP solutions [29]. For every feasible tour $\boldsymbol{s}$ of ATSP its NND can be obtained by statistics which is defined by

$$r(k) \overset{\text{def}}{=} \frac{\boldsymbol{s}(k)}{n}, \quad k = 1, \ldots, n-1 \qquad (21)$$
$$\textstyle\sum_k r(k) = 1$$



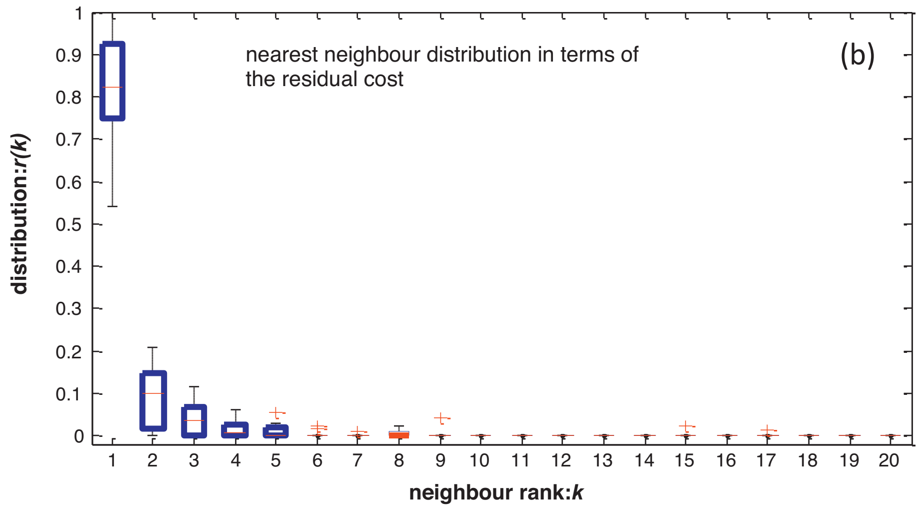


Fig. 4. The nearest neighbor distribution of optimal solutions from “ft17” to “rgb443” (overall 19 instances) in TSPLIB95 (http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/) measured by the actual cost matrix (a) and the residual cost matrix (b) respectively.

where $\boldsymbol{s}(k)$ is total number of the kth-nearest-neighbors for all arcs in this tour. For example $\boldsymbol{s}(1)$ means the number of arcs in the optimal solution $\boldsymbol{s}$ that belongs to nearest neighbor arcs.

Obviously, $r(k)$ is dependent on the cost type. Different cost definition between cities will produce different NND even for the same tour of ATSP. In Fig. 4, we present the box plot of $r(k)$ of optimal solutions for 19 benchmark instances in TSPLIB95 with the actual cost type (a) and residual cost type (b) respectively.

Fig. 4(a) shows that under the measurement of actual cost, only 56% on average arcs in optimal solutions are the nearest neighbor connections and other optimal arcs are mainly scattered through the 2nd to 11th nearest neighbor connections. With the measurement of residual cost as shown in Fig. 4(b), the number are 83% on average for the nearest neighbor connections and other optimal arcs are scattered only through the 2nd to 5th. It means that arc set belonging to optimal solutions under the measurement of residual cost tends to be more focused. This present significant evidence that the choice of edges with small residual cost is better than the choice for edges with small actual cost.

4.2. Experiments on benchmark problems

To validate the performance of the proposed algorithm, both benchmark and randomly generated instances are used as the test instances. The experiments on ATSP problems have been executed on a PC with Pentium(R) Dual 1.80 GHz processor and 2G RAM memory. The algorithm is coded using Microsoft Visual C++ (version 6.0). Some necessary parameters in MIMM-ACO are presented in Table 1. The first 4 items are quoted directly from Dorigo and Gambardella [5] due to the similarity of the problems. The 5th item is based on the statistic NND result directly. The 6th item is chosen so that under the worst case situation we have

$$\tau_{\max} w_{\min} = \frac{\tau_{\min}}{\varphi} w_{\min} = 1.001\tau_{\min} > w_{\max}\tau_{\min}$$

Table 1
Parameter setting of MIMM-ACO.

1	The population size of ants	$m = 10$
2	The relative importance of pheromone versus the biased weight	$\beta = 2$
3	The pheromone value decay parameter	$\rho = 0.1$
4	The ratio of minimum to maximum pheromone value	$\varphi = 1/n$
5	The exploitation ratio	$q_0 = 0.85$
6	The minimum value of biased weights	$w_{min} = 1.001\varphi$
7	Terminal condition parameters	$\theta = 1.5t_0$
8	Local search algorithm	2-opt

Table 2
Solution results for benchmark problems in TSPLIB95 (25 runs for each case).

Instance name	Performance comparison among algorithms							
	MIMM-ACO		MM-ACO		EO		CGAS	
	Deviation over optima (%)	Running time (s)	Deviation over optima (%)	Running time (s)	Deviation over optima (%)	Running time (s)	Deviation over optima (%)	Running time (s)
ft17	***0.0%***	***0.01***	***0.0%***	***0.01***	***0.0%***	***0.01***	***0.0%***	***0.01***
ft53	***0.0%***	***3.53***	0.22%	***3.17***	***0.0%***	3.85	0.35%	6.78
ft70	***0.0%***	9.85	1.71%	10.15	***0.0%***	***8.93***	***0.0%***	15.32
ftv33	***0.0%***	6.12	***0.0%***	9.75	***0.0%***	***4.78***	***0.0%***	28.73
ftv35	***0.0%***	***5.35***	***0.0%***	15.37	***0.0%***	7.35	***0.0%***	21.35
ftv38	***0.0%***	8.64	***0.0%***	10.96	***0.0%***	***7.83***	***0.0%***	29.79
ftv44	***0.0%***	9.37	***0.0%***	12.35	***0.0%***	***8.21***	***0.0%***	37.63
ftv47	***0.0%***	***7.52***	***0.0%***	10.08	***0.0%***	9.37	***0.0%***	29.70
ftv55	***0.0%***	***6.38***	***0.0%***	18.63	***0.0%***	***5.06***	***0.0%***	18.41
ftv64	***0.0%***	***15.37***	***0.0%***	27.65	***0.0%***	16.42	***0.0%***	29.25
ftv70	***0.03%***	64.53	5.78%	61.25	0.72%	***32.26***	0.75%	69.54
ftv170	0.05%	108.28	0.25%	***96.73***	0.28%	103.27	***0.0%***	128.76
kro124	***0.0%***	***33.25***	1.64%	54.21	0.35%	20.86	***0.0%***	78.52
p43	***0.0%***	8.35	0.08%	9.38	0.13%	***5.47***	***0.0%***	7.53
ry48p	***0.0%***	7.83	***0.0%***	7.97	***0.0%***	***5.45***	***0.0%***	12.35
rgb323	***0.0%***	***0.01***	1.3%	96.75	0.06%	87.12	0.13%	103.28
rgb358	***0.0%***	***0.01***	0.75%	75.37	***0.00%***	69.65	0.35%	96.49
rgb403	***0.0%***	***0.01***	1.35%	104.39	***0.00%***	85.32	0.31%	147.83
rgb443	***0.0%***	***0.01***	1.73%	90.65	***0.00%***	76.14	***0.0%***	143.76
Aver. val.	***0.004%***	***15.50***	0.78%	37.66	0.08%	29.33	0.10%	52.89

Note that Devitation over optima$\overset{\text{def}}{=}((f(s) - opt)/opt) \times 100\%$; the best results are highlighted in boldface italics.

where $w_{max} = 1$ (the largest biased weight) that is implied by (12). This means that we prefer an arc with high pheromone level to an arc with large biased weight if the extremal case occurs.

The computational results for the overall 19 benchmark problems are listed in Table 2. The relative deviation over optimum and the running time are reported as the criterion for comparison. To make the numerical results reliable 25 runs are implemented for each instance. Averaged performance results are presented. Table 2 illustrates that the proposed MIMM-ACO overwhelms over the MM-ACO in terms of both the relative deviation and the running time. As far as "relative deviation" is concerned MIMM-ACO is only slightly better than EO and CGAS. However, the running time of CGAS is much longer than that used by the proposed method and the EO. The reason is that CGAS is a kind of combination of AS (ant system) and GA. The advantage of this hybridization for ATSP is still ambiguous and more running time has to be used so as to exploit different characteristics of the two algorithms. For the last four instances, the lower bound solution happens to be a feasible solution and therefore is the optimal solution, which means that these problems are solved optimally in the preprocessing phrase of the MIMM-ACO.

4.3. Experiments on randomly generated problems

Aside from the last four large-scale problems the EO seems to compete with our method. Hence, the following numeric results on the randomly generated instances will be used to show that the improvement by our method is significant. We adopt the generating mode proposed by Carpaneto et al. [30]. Two classes of ATSP instances are considered as follows:

(***a1***) $c_{i,j}$ uniformly random in $[1,10^3]$;
(***t1***) $c_{i,j}$ uniformly random in $[1,10^3]$ with triangle inequality enforced;

Tables 3 and 4 show the results for class ***a1*** and class ***t1*** respectively. (Larger values of instance size for problems of class ***t1*** have not been considered because of the excessive computing time for the regularization of the cost matrices.) The performance

Table 3
Average solution results for class ***a1*** (25 runs for each case).

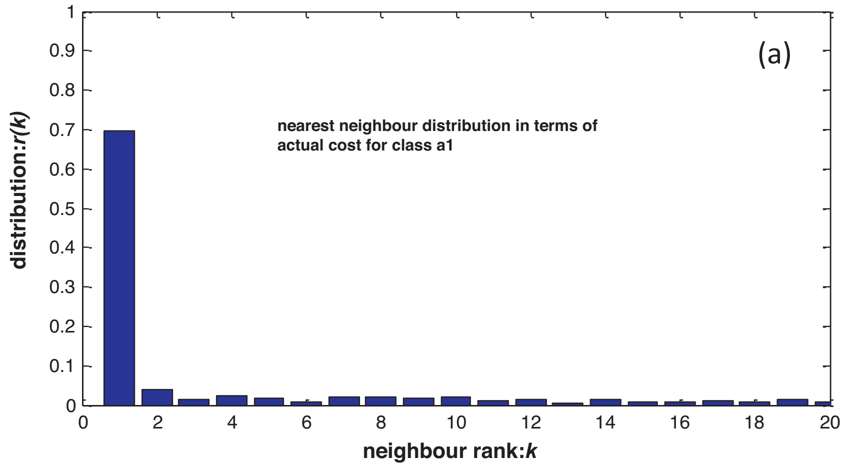
Problem size	Performance comparison among algorithms for class ***a1***							
	MIMM-ACO		MM-ACO		EO		CGAS	
	Deviation over LB. (%)	Running time (s)	Deviation over LB. (%)	Running time (s)	Deviation over LB. (%)	Running time (s)	Deviation over LB. (%)	Running time (s)
100	0.15%	***9.13***	0.17%	22.10	0.13%	15.31	***0.00%***	35.18
200	***0.03%***	***8.35***	0.04%	25.83	0.35%	23.47	0.31%	82.35
300	***0.02%***	***12.74***	0.03%	56.35	0.34%	27.85	0.54%	75.87
400	***0.03%***	***15.58***	***0.01%***	42.38	0.87%	30.29	0.06%	105.53
500	0.012%	***18.75***	0.35%	64.21	0.85%	43.27	***0.02%***	110.26
600	***0.01%***	***19.23***	0.87%	83.25	0.23%	53.91	0.08%	152.54
700	0.12%	***35.34***	0.43%	105.37	0.35%	87.65	***0.05%***	186.73
800	0.15%	***21.37***	0.23%	231.89	0.38%	76.32	***0.01%***	192.35
900	***0.01%***	***25.69***	0.56%	287.63	0.57%	101.35	0.05%	203.47
1000	***0.00%***	***10.01***	0.82%	301.24	0.32%	153.26	0.01%	187.31
Aver. val.	***0.05%***	***17.62***	0.35%	112.02	0.44%	62.27	0.11%	113.16

Table 4
Average solution results for class ***t1*** (25 runs for each case).

Problem size	Performance comparison among algorithms for class ***t1***							
	MIMM-ACO		MM-ACO		EO		CGAS	
	Deviation over LB. (%)	Running time (s)	Deviation over LB. (%)	Running time (s)	Deviation over LB. (%)	Running time (s)	Deviation over LB. (%)	Running time (s)
100	***0.03%***	***12.38***	0.15%	27.72	0.35%	30.14	0.06%	50.31
200	0.02%	***9.54***	0.17%	32.53	0.23%	35.87	***0.01%***	43.87
300	***0.01%***	***15.32***	0.03%	57.84	0.02%	47.65	0.04%	87.64
400	0.03%	***22.13***	***0.01%***	69.76	***0.01%***	53.29	0.02%	107.45
500	***0.02%***	***19.45***	0.05%	65.92	0.73%	87.65	0.15%	135.46
Aver. val.	***0.02%***	***15.76***	0.08%	50.75	0.27%	50.92	0.06%	84.94

Note that Deviation over LB.$\overset{\text{def}}{=}((f(s)-\tilde{Z}^*_{AP})/\tilde{Z}^*_{AP})\times 100\%$ where LB. is the abbreviation of lower bound and, the best results are highlighted in boldface italics.

measures are in terms of the deviation over the known optima (or the lower bound) and the running time. One can observe that the deviation difference among the algorithms is trivial when applied to class ***a1*** and class ***t1***. However, the difference in time efficiency among these algorithms is significant. MIMM-ACO holds remarkable searching efficient compared with the other three algorithms. The reason is that for randomly generated instance there would be a high probability of getting a tight lower bound $\tilde{Z}^*_{AP}$ (the detailed theoretical basis behind the scenario has been studied by Frieze and Sorkin [31]). The tight bound will lead to a significant reduction in the search space by removing non-optimal arcs out of consideration and that is exactly the domain knowledge that the model tells. By using the domain/model information ants are guided to search the most promising solution space by avoiding useless search. While for EO, its performance is strongly dependent on the assumption that the NND of optimal solution follows a scale-free distribution. If it is not the case, its performance would be degraded correspondently. Fig. 5 presents the averaged NND scenario of optimal solution for class ***a1*** (Fig. 4(a)) and for class ***t1*** (Fig. 4(b)). One can observe that the NND of optimal solutions to these kinds of instances do not subject to a scale-free distribution.



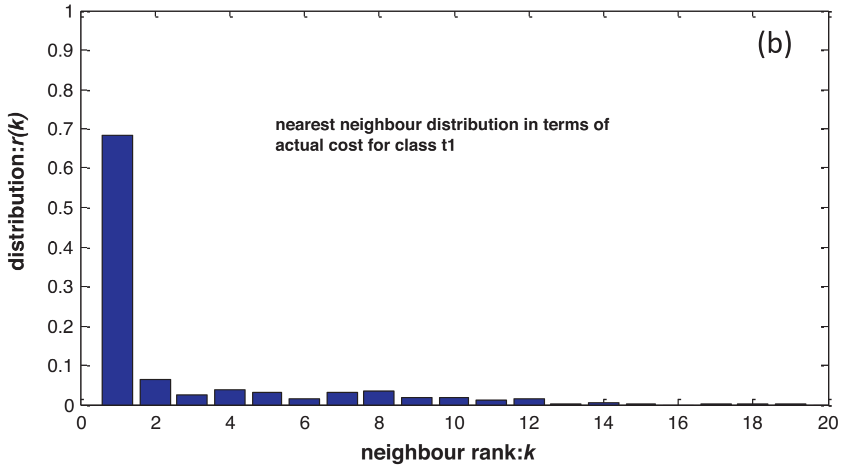


Fig. 5. The averaged nearest neighbor distribution of optimal solutions for 20 instances from class ***a1*** (a) and from class ***t1*** (b) respectively and both NNDs are obtained in terms of actual cost matrix.

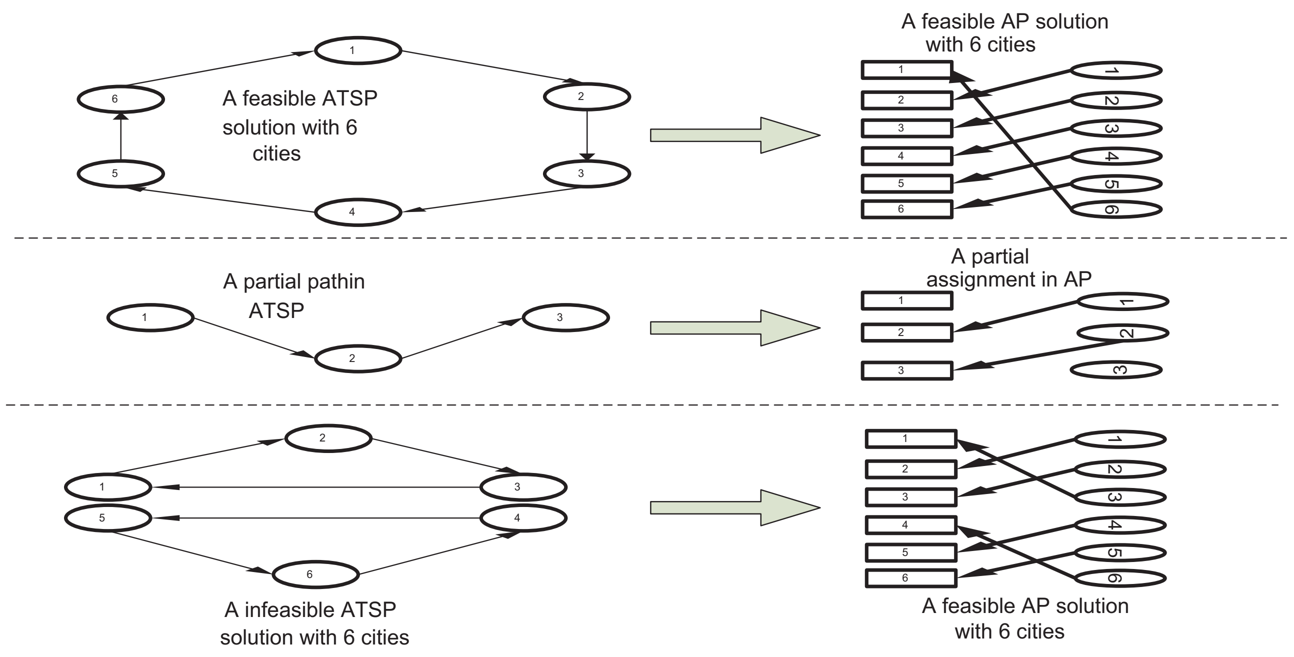

A feasible ATSP solution with 6 cities
A feasible AP solution with 6 cities
A partial pathin ATSP
A partial assignment in AP
A infeasible ATSP solution with 6 cities
A feasible AP solution with 6 cities
1
2
3
4
5
6

Fig. A1. The mapping of ATSP solution into AP solution with a example of 6 cities.

5. Conclusion

In this paper, we have proposed a novel hybrid metaheuristic algorithm for ATSP problem with analytical mechanisms. The MIMM-ACO optimization utilizes the information acquired from both the ATSP problem and the employed algorithm itself. It provides an analytical way to incorporate exact methods with ACO and thus is capable of improving the searching efficiency in a definite way. The idea behind it is that any well-suited algorithm requires one should have deep understanding and explicit knowledge of the problem. The more information one has, the more efficient the algorithm would be. However, the knowledge coming from the lower bound structure is always associated with the structure of the particular problem being solved. This puts forward a new interesting problem on selection of good lower bound structures to provide in-depth insight into the problem under consideration. This also points out a good direction for the future research.

Acknowledgments

We thank three anonymous referees whose many constructive suggestions helped improve this paper significantly. The authors would also like to thank the financial support provided by Nature Science Foundation of China under Grant 61074150 and the High Technology Research and Development Program of China under Grant 2009AA04Z162.

Appendix A.

Proof. We first give the formal definition of the associated AP problem before proving Proposition 1. The AP problem is formulated as

$$\tilde{Z}^*_{AP} = \min \sum\nolimits_{i\in \boldsymbol{V}} \sum\nolimits_{j\in \boldsymbol{V}} c_{i,j} x_{ij} \tag{A.1}$$

$$\boldsymbol{s.t.} \quad \begin{matrix} \sum\nolimits_{j\in \boldsymbol{V}} x_{i,j} = 1, & i \in \boldsymbol{V} \\ \sum\nolimits_{i\in \boldsymbol{V}} x_{i,j} = 1, & j \in \boldsymbol{V} \end{matrix} \tag{A.2}$$

$$0 \le x_{i,j} \le 1 \tag{A.3}$$

Because the constraint matrix defined by (A.2) is totally unimodular [26], The AP is actually a linear programming so that we can use (A.3) equivalently. Therefore, the dual problem can be defined by

$$\tilde{Z}^*_{AP} = \max \sum_{i\in \boldsymbol{V}} u_i + \sum_{j\in \boldsymbol{V}} v_j \tag{A.4}$$

$$\boldsymbol{s.t.} \quad c_{i,j} - u_i - v_j \ge 0, \quad i, j \in \boldsymbol{V} \tag{A.5}$$

Such that the residual cost $\bar{c}_{i,j} \overset{\text{def}}{=} c_{i,j} - u_i - v_j$ is obtained.

According to dual theory of the linear programming theory [32] the following properties hold

$$\begin{aligned} \tilde{Z}^*_{AP} &= \tilde{Z}^*_{AP} + \sum_{(i,j)\in \boldsymbol{R}} \bar{c}_{i,j} x_{i,j} \\ \bar{c}_{i,j} x_{i,j} &= 0, \quad \forall (i,j) \end{aligned} \tag{A.6}$$

where $\boldsymbol{R}$ is the index set of non-basic arcs.

Since any feasible, infeasible or partial solution in ATSP has a counterpart solution to AP problem (see Fig. A1 for an example with graphic illustration), we denote $\tilde{Z}^*_{AP}(\boldsymbol{s^p})$ as the optimal solution value of the AP that must contain $\boldsymbol{s^p}$ as a partial assignment solution. Because $\sum_{(i,j)\in \boldsymbol{s^p}} \bar{c}_{i,j}$ represents a lower bound on the increase of $\tilde{Z}^*_{AP}$ corresponding to the inclusion of $\boldsymbol{s^p}$ in the solution of AP, it yields immediately that

$$\begin{aligned} \tilde{Z}^*_{AP} + \textstyle\sum_{(i,j)\in \boldsymbol{s^p}} \bar{c}_{i,j} &\le \tilde{Z}^*_{AP}(\boldsymbol{s^p}) \\ \tilde{Z}^*_{AP} \le \tilde{Z}^*_{AP}(\boldsymbol{s^p}) &\le Z^*(\boldsymbol{s^p}) \\ Z^* &\le Z^*(\boldsymbol{s^p}) \end{aligned} \tag{A.7}$$

So, with (A.7) we have that

$$f(s) \le \tilde{Z}^*_{AP} + \sum_{(i,j)\in \boldsymbol{s^p}} \bar{c}_{i,j} \Rightarrow f(s) \le \tilde{Z}^*_{AP}(\boldsymbol{s^p}) \Rightarrow f(s) \le Z^*(\boldsymbol{s^p})$$

□

Appendix B.

Proof. The proof is a slightly modified version from results of Stutzle and Dorigo [24]. The proof is conducted by giving a bound on the length of t_0. Assuming the worst case situation that $\exists(i,j)\in s^*$, $\tau^*_{i,j}(t^*)=\tau_{\min}$ and $\exists(k,l)\notin s^*$, $\tau^*_{k,l}(t^*)=\tau_{\max}$, and using the global pheromone updating rule the pheromone trail at $(t^*+\hat{t})$ becomes

$$\tau^*_{i,j}(t^*+\hat{t}) = (1-\rho)^{\hat{t}}\tau_{\min} + \sum_{i=0}^{\hat{t}-1}(1-\rho)^i g(s^*)$$
$$= (1-\rho)^{\hat{t}}\tau_{\min} + \frac{g(s^*)}{\rho}(1-(1-\rho)^{\hat{t}}) \qquad (B.1)$$

Combining (9) it yields

$$\tau^*_{i,j}(t^*+\hat{t}) = (1-\rho)^{\hat{t}}(\tau_{\min}-\tau_{\max})+\tau_{\max} \qquad (B.2)$$

While the value of $\tau_{k,l}(t^*+\hat{t})$ equals

$$\tau_{k,l}(t^*+\hat{t}) = \max(\tau_{\min},(1-\rho)^{\hat{t}}\tau_{\max}) \qquad (B.3)$$

Solving the inequality $\tau^*_{i,j}(t^*+\hat{t}) > \tau_{k,l}(t^*+\hat{t})$ leads to

$$(1-\rho)^{\hat{t}}(\tau_{\min}-\tau_{\max})+\tau_{\max} > \max(\tau_{\min},(1-\rho)^{\hat{t}}\tau_{\max}) \qquad (B.4)$$

From our standpoint, we are more interesting in the case that $\tau_{k,l}(t^*+\hat{t}) = (1-\rho)^{\hat{t}}\tau_{\max}$, then we have

$$(1-\rho)^{\hat{t}}(\tau_{\min}-\tau_{\max})+\tau_{\max} > (1-\rho)^{\hat{t}}\tau_{\max} \Rightarrow \hat{t} > \frac{\ln(1/(2-\varphi))}{\ln(1-\rho)} \qquad (B.5)$$

Therefore, any transition period with the length larger than $ln(1/(2-\varphi))/ln(1-\rho)$ leads to (15).

For any $(k,l)\notin s^*$ under the extreme case the pheromone trail at iteration $t^*+\tilde{t}$ holds $\tau_{k,l}(t^*+\tilde{t}) = \max(\tau_{\min},(1-\rho)^{\tilde{t}}\tau_{\max})$

So,

$$\tau_{k,l}(t^*+\tilde{t}) = \tau_{\min} \Rightarrow (1-\rho)^{\tilde{t}}\tau_{\max} \le \tau_{\min} \Rightarrow \tilde{t} \ge \frac{\ln(\varphi)}{\ln(1-\rho)} \qquad (B.6)$$

Therefore, any transition period with the length larger than $ln(\varphi)/ln(1-\rho)$ leads to (16).

From (B.5) and (B.6) we conclude that (15) and (16) will be satisfied if we take

$$t_0 = \max\left\{\frac{\ln(\varphi)}{\ln(1-\rho)}, \frac{\ln(1/(2-\varphi))}{\ln(1-\rho)}\right\} = \frac{\ln(\varphi)}{\ln(1-\rho)} \qquad (B.7)$$

because $ln(\varphi) < ln(1/(2-\varphi))$ always holds when $0<\varphi<1$.

□

Appendix C.

Proof. The proof is quite simple and is similar to the proof in Stutzle and Dorigo [24]. The newly employed mechanism in ACO is thoroughly based on rigorous theoretical analysis. At least one optimal solution is preserved. We can guarantee that any feasible choice for some ant located in some city including the optimal choice is conducted with a probability $p_{min}>0$. See (11) and assume that the one ant positioned in city r, a trivial lower bound for $p_{min}(r)$ under the worst case situation can be calculated as

$$p_{\min}(r) \ge \hat{p}_{\min}(r) = \frac{w_{\min}\tau_{\min}}{w_{\min}\tau_{\min}+(n_r-1)\tau_{\max}} \qquad (C.1)$$

where n_r is the number of all allowable connections derived from city r or the neighborhood size of city r.

Rewrite (C.1) and then we have

$$\hat{p}_{\min}(r) = \frac{w_{\min}\varphi}{w_{\min}\varphi+(n_r-1)} \qquad (C.2)$$

See from (C.2) that $\hat{p}_{\min}(r)$ is a decreasing function on n_r which means the smaller the neighborhood size, the larger the probability bound. Let $n_{max}=max\{n_r|r=1,2,\ldots,n\}$. Then it leads to

$$\forall r, \quad \hat{p}_{\min}(r) > \bar{p}_{\min} = \frac{w_{\min}\varphi}{w_{\min}\varphi+(n_{\max}-1)} > 0 \qquad (C.3)$$

Then, any generic solution s', including any optimal solution s^*, can be produced with a probability $\hat{p} > (\bar{p}_{\min})^n > 0$. Such that a lower bound for $P^*(t)$ is given by

$$\hat{P}^*(t) = 1-(1-\hat{p})^t \qquad (C.4)$$

By choosing t sufficiently large, the results are proved.

□

References

[1] L. Tang, L. Huang, Optimal and near-optimal algorithms to rolling batch scheduling for seamless steel tube production, International Journal of Production Economics 105 (2) (2007) 357–371.
[2] C. Pan, G.K. Yang, A method of solving a large-scale rolling batch scheduling problem in steel production using a variant of column generation, Computers & Industrial Engineering 56 (1) (2009) 165–178.
[3] C. Theys, O. Braysy, et al., Using a TSP heuristic for routing order pickers in warehouses, European Journal of Operational Research 200 (3) (2010) 755–763.
[4] T. Keskinturk, M.B. Yildirim, et al., An ant colony optimization algorithm for load balancing in parallel machines with sequence-dependent setup times, Computers & Operations Research 39 (6) (2012) 1225–1235.
[5] M. Dorigo, L.M. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem, IEEE Transactions on Evolutionary Computation 1 (1) (1997) 53–66.
[6] E. Bonabeau, M. Dorigo, et al., Inspiration for optimization from social insect behaviour, Nature 406 (6791) (2000) 39–42.
[7] C. Blum, M. Dorigo, The hyper-cube framework for ant colony optimization, IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics 34 (2) (2004) 1161–1172.
[8] M. Dorigo, C. Blum, Ant colony optimization theory: a survey, Theoretical Computer Science 344 (2–3) (2005) 243–278.
[9] C. Voudouris, E. Tsang, Guided local search and its application to the traveling salesman problem, European Journal of Operational Research 113 (2) (1999) 469–499.
[10] C.-B. Cheng, C.-P. Mao, A modified ant colony system for solving the travelling salesman problem with time windows, Mathematical and Computer Modelling 46 (2007) 1225–1235.
[11] J. Yang, X. Shi, et al., An ant colony optimization method for generalized TSP problem, Progress in Natural Science 18 (11) (2008) 1417–1422.
[12] U. Aybars, et al., An interactive simulation and analysis software for solving TSP using ant colony optimization algorithms, Advances in Engineering Software 40 (5) (2009) 341–349.
[13] A. Puris, R. Bello, et al., Analysis of the efficacy of a two-stage methodology for ant colony optimization: case of study with TSP and QAP, Expert Systems with Applications 37 (7) (2010) 5443–5453.
[14] S. Ghafurian, N. Javadian, An ant colony algorithm for solving fixed destination multi-depot multiple traveling salesmen problems, Applied Soft Computing 11 (1) (2011) 1256–1262.
[15] S.-M. Chen, C.-Y. Chien, Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques, Expert Systems with Applications 38 (12) (2011) 14439–14450.
[16] G. Dong, W.W. Guo, Solving the traveling salesman problem using cooperative genetic ant systems, Expert Systems with Applications (2011), http://dx.doi.org/10.1016/j.eswa.2011.10.012.
[17] V. Maniezzo, Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem, INFORMS Journal on Computing 11 (4) (1999) 358–369.
[18] I.-C. Choi, S.-I. Kim, et al., A genetic algorithm with a mixed region search for the asymmetric traveling salesman problem, Computers & Operations Research 30 (5) (2003) 773–786.
[19] P.I. Cowling, R. Keuthen, Embedded local search approaches for routing optimization, Computers & Operations Research 32 (3) (2005) 465–490.
[20] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 1 (1) (1997) 67–82.
[21] C. Blum, J. Puchinger, et al., Hybrid metaheuristics in combinatorial optimization: a survey, Applied Soft Computing 11 (6) (2011) 4135–4151.
[22] L. Jourdan, M. Basseur, et al., Hybridizing exact methods and metaheuristics: a taxonomy, European Journal of Operational Research 199 (3) (2009) 620–629.
[23] T. Stutzle, H.H. Hoos, MAX-MIN ant system, Future Generation Computer Systems 16 (8) (2000) 889–914.
[24] T. Stutzle, M. Dorigo, A short convergence proof for a class of ant colony optimization algorithms, IEEE Transactions on Evolutionary Computation 6 (4) (2002) 358–365.

[25] R.M. Karp, A patching algorithm for the nonsymmetric traveling-salesman problem, SIAM Journal on Computing 8 (4) (1979) 561–573.
[26] M. Dell'Amico, P. Toth, Algorithms and codes for dense assignment problems: the state of the art, Discrete Applied Mathematics 100 (1–2) (2000) 17–48.
[27] D.L. Miller, J.F. Pekny, Exact solution of large asymmetric traveling salesman problems, Science 251 (4995) (1991) 754–761.
[28] Y.-W. Chen, Y.-J. Zhu, et al., Improved extremal optimization for the asymmetric traveling salesman problem, Physica A: Statistical Mechanics and its Applications 390 (23–24) (2011) 4459–4465.
[29] Y.-W. Chen, Y.-Z. Lu, et al., Optimization with extremal dynamics for the traveling salesman problem, Physica A: Statistical Mechanics and its Applications 385 (1) (2007) 115–123.
[30] G. Carpaneto, M. Dell'Amico, et al., Exact solution of large-scale, asymmetric traveling salesman problems, ACM Transactions on Mathematical Software 21 (4) (1995) 394–409.
[31] A. Frieze, G.B. Sorkin, The probabilistic relationship between the assignment and asymmetric traveling salesman problems, SIAM Journal on Computing 36 (5) (2007) 1435–1452.
[32] L.W. Wayne (Ed.), Operations Research Mathematical Programming, TsingHua Publishing Company, Beijing, 2002.