Available online at www.sciencedirect.com
ScienceDirect
Computers & Industrial Engineering xxx (2008) xxx–xxx
computers & industrial engineering
www.elsevier.com/locate/caie

Superior/Inferior Segment-Discriminated Ant System for combinatorial optimization problems

Feng-Cheng Yang *, Yon-Chun Chou

Graduate Institute of Industrial Engineering, National Taiwan University, 1, Section 4, Roosevelt Road, Taipei 106, Taiwan

Abstract

The Ant Colony Optimization method is a heuristic algorithm for solving various optimization problems, particularly the combinatorial optimization problems. Traditional ant-optimization methods might encounter search stagnation owing to a biased pheromone map that is dominated by local optimal trails. To overcome this drawback and lower the number of solution constructions for finding the optima, this paper presents an improving ant-optimization system, the Superior/Inferior Segment-Discriminated Ant System (SDAS). This system proposes a segment-based pheromone update strategy to deposit pheromone on superior segments and withdraw pheromone from inferior ones. The method uses the control-chart technique to define superior and inferior limits to partition the constructed solutions into superior, inferior, and ordinary solutions. Inferior and superior segments are then extracted from the superior and inferior solutions by stochastic set operations. Since the pheromone map is not easily dominated by any local optimal trail, the solution search is more efficient and effective. Several benchmarks from the TSP-LIB and OR-LIB were used as sample problems to test the proposed system against other ant-optimization systems, including the AS, ACS, AS_rank, AS_elite, and MMAS. Numerical results indicated that the SDAS obtains solutions that are similar to or better than others. Maturity index for the pheromone map was discussed and experimental results showed that the proposed method was able to prolong the time for the map to maturity to avoid earlier search stagnation.

Keywords: Ant Colony Optimization System; Pheromone withdraw; Segment-based pheromone update; Superior and inferior segments; Segment-Discriminated Ant System

1. Introduction

The Ant Colony Optimization (ACO) method is a population-based heuristic method that has been successfully applied to solve several NP-hard combinatorial optimization problems (Dorigo & Stützle, 2003, 2004; Dorigo, Caro, & Gambardella, 1999). The first ACO algorithm was the Ant System (Colorni, Dorigo, & Maniezzo, 1991) developed by Dorigo, Maniezzo, and Colorni (1996), which was applied to the traveling salesman problem. The development of the system was inspired by the food-searching behavior of real ants.

* Corresponding author. Tel.: +886 2 33669503; fax: +886 2 23625856.
E-mail address: iefcyang@ntu.edu.tw (F.-C. Yang).

0360-8352/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cie.2007.12.016

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

Ants communicate with each other by depositing pheromones on the trails they traverse. Remarkably, one will find a colony of ants traveling on the shortest path between the food source and their formicary to move the food home.

In general, an ACO method uses a set of artificial ants to construct solutions step by step and evolve the solutions toward the optima. Three algorithmic mechanisms are involved in constructing a solution. The first is a proportional probabilistic selection mechanism that guides the ants to select the next thing or object to participate in the process of solution construction. The second is a memory-recording mechanism that makes each ant record its own solution under construction and stores the best solution constructed so far by the colony of ants. The third is a pheromone update mechanism that maintains a pheromone map to guide the ants' solution search. The operations for constructing a solution are executed by the set of artificial ants iteration by iteration to evolve the best solution constructed so far.

After the debut of the Ant System (AS), several extended, enhanced, and improved ant systems were successively introduced. Typical systems include the Ant Colony System (ACS) (Dorigo & Gambardella, 1997), the elitism-based (AS_elite) and the rank-based (AS_rank) Ant Systems (Bullnheimer, Hartl, & Strauss, 1999), and the Max–Min Ant System (MMAS) (Stützle & Hoos, 2000). Some of the improved systems focus on better selection mechanisms, some on pheromone update operations, and some on solution constructions. To enhance the effectiveness of the ACO method, this paper presents an improved ant system, Superior/Inferior Segment-Discriminated Ant System (SDAS). The SDAS proposes a segment-based pheromone update strategy and an addition/subtraction discriminated pheromone update operation to avoid a quickly matured pheromone map (a biased map that is dominated by local optima). To examine the performance of the SDAS, we only implemented the system for the traveling salesman problems and bin-packing problems; although the applications can be applied to various kinds of combinatorial optimization problems.

We first briefly review several ACO methods and their approaches to solving the TSP and BPP in Section 2; then present our ant system, the SDAS, in Section 3. Sections 4 and 5 display and compare the numerical results of our method with other systems in solving the traveling salesman and bin-packing problems. In the final section, conclusive remarks are addressed.

2. Literature review and background to theory

The Ant Colony Optimization method is a constructive heuristic algorithm. The solution to an optimization problem is constructed step by step, by a population of agents – artificial ants. This method has been widely used in solving combinatorial optimization problems whose solutions can be constructed iteratively by heuristic algorithms. From a generalized viewpoint, a set of *objects* can be identified in a combinatorial optimization problem. Constructing a solution to the problem is a repeated procedure that deals with these objects one by one. For example, the objects for a Traveling Salesman Problem (TSP) are the cities to be visited by the salesman. An artificial ant composes a sequence of these cities (objects) one by one to form a traveling route. For a Bin-Packing Problem (BPP), the objects are the items (parcels or boxes) to be packed into a minimum number of bins. An artificial ant repetitively selects an object to place it to an allocated group (opened bin) or additionally creates a new group (opens a new bin) before the placement, until no object is left.

In constructing a solution, objects are processed by the ant one by one until all of them are processed and put into the solution. Normally, ants carry out this process by repeatedly selecting an object stochastically from an updated candidate set of objects. Objects in the candidate set are either selected from the set of non-processed objects stochastically or deterministically (for reducing the computation cost), or selected from the set subject to constraints of the problem. The selected object is directly appended or inserted into the partial solution under construction. The procedures of updating the candidate set, selecting an object, and placing it to the solution are then iteratively executed until all objects are processed.

Although the object selection procedures for the TSP and BPP are in general the same, their solution structures are different. Suppose that z is the number of objects and $\mathcal{O}$ is the set of the objects, $\mathcal{O} = \{1, 2, \ldots, z\}$. Let $\mathbf{S}$ represent a solution; then $\mathbf{S}$ for a TSP is an ordered sequence of objects, where

$$\mathbf{S} = \langle s_1, s_2, \ldots, s_z \rangle, \quad s_i \in \mathcal{O};\ s_i \neq s_j,\ \forall i \neq j. \tag{1}$$

In contrast, $\mathbf{S}$ for a BPP is a set of groups of objects, where

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

$$\mathbf{S} = \{S_1, S_2, \ldots, S_g\} = \{\{s_{11}, s_{12}, \ldots, s_{1|S_1|}\}, \ldots, \{s_{g1}, s_{g2}, \ldots, s_{g|S_g|}\}\}; \quad s_{kj} \in \mathcal{O}; \ k = 1, 2, \ldots, g; \\ j = 1, 2, \ldots, |S_k|; \ \text{and} \sum_{k=1}^{g} |S_k| = z. \tag{2}$$

For almost all of the ACO methods, a probability proportional selection method is used to select an object from the candidate set to construct a solution. The selection probability for each candidate is computed from a referenced pheromone value on a pheromone map and a computed heuristic value. Specifically, the probability of object j to be selected to succeed object i is calculated from a pheromone value τ_{ij} and a heuristic value η_{ij}. The heuristic value η_{ij} is calculated using a local optimization approach to guide the selection complying with the optimization objective. If the value of η_{ij} is irrelevant with the objective, the optimality quest will be ineffective. Detailed formulations of the probability calculations are problem-dependent, and different ant systems might have their own designs for them. The pheromone value τ_{ij} is defined in a pheromone map. The pheromone map is usually designed as a two-dimensional matrix, whose elements are constructed from the links between two objects or between one object and other item (e.g., a bin in BPPs). Different problems have their own designs of the matrix. An object-sequencing problem is to arrange the orders of objects to achieve an optimization goal, where the TSP is a typical example. The pheromone map is designed as a square matrix,

$$\mathbf{T} = [\tau_{ij}]_{z\times z} = \begin{bmatrix} \tau_{11} & \tau_{12} & \cdots & \tau_{1z} \\ \tau_{21} & \tau_{22} & \cdots & \tau_{2z} \\ \vdots & \vdots & \ddots & \vdots \\ \tau_{z1} & \tau_{z2} & \cdots & \tau_{zz} \end{bmatrix}. \tag{3}$$

The pheromone value τ_{ij} is thus defined as the tendency that object i is followed by object j in the ordered sequence of objects. For a object-grouping problem (e.g., the BPP), a pheromone item τ_{ij} may be defined as the tendency that object i and object j are grouped together in the same group or the tendency that object i is grouped into group j. For the former case, the map is a square matrix, $\mathbf{T} = [\tau_{ij}]_{z\times z}$. In contrast, $\mathbf{T} = [\tau_{ij}]_{z\times g}$ for the latter case, where g is the number of groups. Elements in the matrix are normally initialized with a constant value τ_0 and then updated by different pheromone update operations in different ACO methods.

To record *trails* of good solutions on the pheromone map, a solution is decomposed into a set of object-links and the pheromone values corresponding to these links are intensified. In this paper, an object-link decomposed from a solution is called a *segment*. Let $\boldsymbol{L} = \{l_{ij}\}$ be the segment set of all object-links of the problem, where l_{ij} is directly corresponding to τ_{ij}. Therefore, $\boldsymbol{L} = \{l_{ij} | i, j = 1, 2, \ldots, z\}$ for the TSP and the BPP with $\mathbf{T} = [\tau_{ij}]_{z\times z}$, while $\boldsymbol{L} = \{l_{ij} | i = 1, 2, \ldots, z; \ j = 1, 2, \ldots, g\}$ for the BPP with $\mathbf{T} = [\tau_{ij}]_{z\times g}$. To identify the pheromone items to which a solution corresponds or to decompose a solution into object-links, a segmentation operation on the solution is performed. Let $H^{\text{problem type}}(\cdot)$ be the *segmentation operator* of a problem type. The segmentation result is a set of segments, which is also a subset of $\boldsymbol{L}$. Therefore,

$$H^{\text{problem type}}(\mathbf{S}) \subseteq \boldsymbol{L}. \tag{4}$$

Once $\mathbf{S}$ is granted the rights for pheromone intensification, the pheromone items corresponding to the segment set are identified.

At the end of a computational iteration or after a solution construction step, pheromone values on the pheromone map are updated accordingly. Different update methods have been proposed in much ACO literature. In current ACO methods, pheromone values are usually intensified by one or a few elite solutions. The elitism and the amount of pheromone added are determined from evaluating their objective achievements by the objective function $f^{\text{problem type}}(\cdot)$. Conversely, all pheromone values are subjected to *evaporation* by deducting a certain amount from the values. In general, two kinds of pheromone intensification strategies are either individually or jointly conducted in ant-optimization systems: step-wise and trail-wise strategies. The step-wise pheromone intensification is conducted right after an ant has selected and added an object to the solution under construction. Since no elitism information is available during the construction, every ant is granted the rights to lay pheromone on the related segment in each step. The AS (Dorigo et al., 1996) executes this step-wise pheromone intensification operation to mimic the natural behavior of real ants. This strategy is also

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

called *online pheromone update*. The ACS adopts this strategy but names it *local pheromone update* (Dorigo & Gambardella, 1997). The trail-wise strategy, also called *offline pheromone update* in literature, allows one or a few ants to update the segments of their solutions when an iteration of solution construction is completed. The AS_elite and AS_rank systems use this strategy. The ACS also adopts this strategy but names it *global pheromone update*.

As noticed, a pheromone management demon is employed to monitor the trail-wise pheromone intensification strategy. This implementation is in fact deviated from natural behaviors of real ants, yet makes the system more intelligent. Moreover, most of the ACO methods focus on trail elitism, not segment elitism, and on pheromone deposit (intensification), not deduction. In contrast, Maniezzo (1999) fist proposed an update strategy that either increases or decreases the pheromone value based on the objective value of a solution. The proposed improving method was applied to solve the Quadratic Assignment Problem. Although the concept of pheromone deduction was introduced, the pheromone updates were applied to all the segments of the solution trail. This paper, however, suggests that not only the trails but also the segments on the trails should be further distinguished for either rewarded with pheromone addition or punished with pheromone deduction. Detailed design of this strategy is presented in Section 3.

List 1 outlines the main operations of a general ACO method. The solution construction procedure starts from line 5 to line 9; its details are problem-dependent. Line 8 executes the step-wise pheromone intensification strategy when an object is selected and added to the solution. Line 12 executes a user-specified local search to enhance solution quality; however, this procedure is optional. Line 15 executes the frequently used trail-wise pheromone intensification strategy to select a few elite trails for pheromone deposit. Line 17 optionally executes a pheromone-reset operation to escape from search stagnation.

List 1. The computational flow of a general ACO method.

```
1. Initialize the pheromone map.
2. Construct an initial solution and assign it the best solution so far.
3. For r ← 1 to iteration_limit
4.    For k ← 1 to number_of_ants
5.       For i ← 1 to number_of_objects
6.          Construct and update the candidate set of non-processed objects.
7.          Select one object from the candidate set and add it to the solution.
8.          If a step-wise pheromone intensification strategy is used, add pheromone to the related segment.
9.       End for
10.      If a trail-wise pheromone intensification strategy is used, add pheromone to the segments of the constructed trail.
11.   End for
12.   Optionally execute a local search algorithm to improve the solution quality.
13.   Evaluate the constructed solution and update the best solution so far.
14.   Check for stop conditions to exit the algorithm.
15.   If elitism pheromone update strategy is used, identify the elite trails and add pheromone to the segments of the trails.
16.   Evaporate a certain amount of pheromone from all pheromone items.
17.   If a pheromone-reset policy is used, check for search stagnation to reset the pheromone map.
18. End for
```

The following two subsections illustrate the ACO models for the TSP and BPP, which were adopted in the implementation of the proposed system.

2.1. Ant Colony Optimization methods for the traveling salesman problem

The TSP is essentially an object-sequencing problem. Many practical optimization problems are derived from the TSP. Examples include various vehicle routing problems, integrated-circuit chip mounting (onto PC boards) problems, assembly sequencing problems, job scheduling problems, etc. The TSP solution **S**

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

shown in Eq. (1) is a route of ordered cities (objects), where s_i is the ith visited city in the route. The set of segments extracted from the route is:

$$H^{\mathrm{TSP}}(\mathbf{S}) = \{l_{s_i s_{i+1}}, l_{s_{i+1} s_i} | i = 1, 2, \ldots, z-1\} \cup \{l_{s_z s_1}, l_{s_1 s_z}\}. \tag{5}$$

For example, the extracted segments for a solution $\mathbf{S} = \langle 1,2,5,3,4 \rangle$ are l_{12}, l_{21}, l_{25}, l_{52}, l_{53}, l_{35}, l_{34}, l_{43}, l_{41}, and l_{14}. The corresponding pheromone items to these segments are then the targets for pheromone intensification. The objective of the TSP is to minimize the route length

$$f^{\mathrm{TSP}}(\mathbf{S}) = \sum_{i=1}^{z-1} d_{s_i s_{(i+1)}} + d_{s_z s_1}, \tag{6}$$

where d_{jk} is the distance from city j to city k. In the AS, Dorigo et al. (1996) defined the probability that the ant chooses city j from the candidate set N_i to succeed city i as

$$P_{ij} = \frac{\tau_{ij}^{\alpha} \eta_{ij}^{\beta}}{\sum_{k \in N_i} \tau_{ik}^{\alpha} \eta_{ik}^{\beta}}, \quad \forall j \in N_i. \tag{7}$$

In solving large-scale TSPs, the size of the candidate set N_i is set fixed to reduce computation complexity. The candidate cities in N_i are then a fixed number of closest cities to city i. Since the objective of the TSP is to obtain a shortest route looping through all the cities, selecting a nearest city j to succeed city i is preferred. The heuristic value η_{ij} is then simply assigned as the inverse of distance d_{ij}; i.e., $\eta_{ij} = 1/d_{ij}$. Power factors α and β are used to intensify or relieve the influence of the values of τ_{ij} and η_{ij}, respectively. Therefore, a higher value of $\tau_{ij}^{\alpha} \eta_{ij}^{\beta}$ indicates a higher probability that object j will be chosen to succeed object i. If τ_{ij}^{α} dominates the value, the solution search relies heavily on previous construction experience. In this case, search stagnation occurs easily, since all ants refer to the same pheromone map and the map is likely to be biased by dominating routes. Conversely, if α is set to 0, the heuristic value η_{ij} dominates the selection and the solution search turns out to be a *greedy search*, since local optimization heuristics are usually implemented in the computation of η_{ij}. After all the ants have constructed their solutions, pheromone values τ_{ij} are updated by

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \Delta\tau_{ij}; \quad \forall i, j, \tag{8}$$

where ρ is an evaporation rate that removes a small amount of pheromone from every segment. The added amount of $\Delta\tau_{ij}$ depends on whether the corresponding segment l_{ij} of τ_{ij} belongs to the segment set resulted from the segmentation operation on a solution that deserve pheromone intensification. In the AS, since all solutions constructed from the y ants are eligible for pheromone intensification,

$$\Delta\tau_{ij} = \sum_{k=1}^{y} \Delta\tau_{ij}^{k}; \Delta\tau_{ij}^{k} = \begin{cases} \frac{Q}{f^{\mathrm{TSP}}(\mathbf{S}^k)} & \text{if } l_{ij} \in H^{\mathrm{TSP}}(\mathbf{S}^k) \\ 0 & \text{otherwise,} \end{cases} \tag{9}$$

where Q is a scalar and $\mathbf{S}^k$ is the solution constructed by ant k. Various pheromone update methods for the TSP were proposed in the successive ant systems for efficiency and quality improvements, such as the AS_elite, AS_rank, ACS, and MMAS.

2.2. Ant Colony Optimization methods for the bin-packing problem

The bin-packing problem can be regarded as an object-grouping problem. A one-dimensional bin-packing problem is to pack z objects (boxes or parcels with different sizes or weights) into a number of bins, subject to a size or weight capacity constraint of the bin. The goal is either to use a minimum number of bins to pack these objects, or to minimize the size or weight variances between the bins when the number of bins is given. The solution $\mathbf{S}$ for the BPP is represented in Eq. (2) as a set of subsets of objects, where

$$S_k = \{s_{k1}, s_{k2}, \ldots, s_{k|S_k|}\}; \quad s_{kj} \in \mathcal{O}; \ \ j = 1, 2, \ldots, |S_k|. \tag{10}$$

Note that $\cup_{k=1}^{g} S_k = \mathcal{O}$ and the capacity constraints are

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

$$W_k = \sum_{j \in s_k} w_j \leqslant W_0; \quad k = 1, 2, \ldots, g, \tag{11}$$

where w_j is the size or weight of object j and W_0 is the size or weight capacity of the bin. Levine and Ducatelle (2004) presented an ant-optimization method for the BPP and Cutting Stock Problem. The objective for the BPP defined in their work was originally proposed by Falkenauer (1996), which was to maximize

$$f^{\mathrm{BPP}}(\mathbf{S}) = \frac{1}{g} \sum_{k=1}^{g} \left(\frac{W_k}{W_0} \right)^{\lambda}. \tag{12}$$

In Eq. (12), g is the number of bins used, W_k is the total weight of the objects packed in bin k; and λ is an objective influence factor of the load fraction of bin to the number of bin used. Maximizing the average of load fractions of the bins can be interrupted as minimizing the number of bin used. Since the number of bins used, g, is not predetermined and is to be minimized, the pheromone map used is a square matrix; i.e., $\mathbf{T} = [\tau_{ij}]_{z \times z}$. Therefore, segmentation operation on a BPP solution focuses on the links of objects packed in the same bin, not on the links between the bin and the packed objects. Therefore, the segmentation operation on solution $\mathbf{S}$ is

$$H^{\mathrm{BPP}}(\mathbf{S}) = \bigcup_{k=1}^{g} H'(S_k), \tag{13}$$

$$H'(S_k) = \{ l_{ij} | i, j \in S_k, \ i \neq j \}. \tag{14}$$

$H'(S_k)$ is the segmentation operation on the packing results of bin k. The operation result is the complete set of links between objects in bin k. Fig. 1 depicts a BPP solution $\mathbf{S} = \{S_1, S_2, S_3, S_4\} = \{\{5,1\}, \{4,2\}, \{9,6,3\}, \{7,8\}\}$, which has nine objects packed into four bins. For each object depicted as a round-rectangle in the bin stack, the value shown within the parentheses is its weight. Segmentation results of the solution are $H'(S_1) = \{l_{51}, l_{15}\}$ for bin 1, $H'(S_2) = \{l_{42}, l_{24}\}$for bin 2, $H'(S_3) = \{l_{96}, l_{93}, l_{63}, l_{69}, l_{39}, l_{36}\}$ for bin 3, and $H'(S_4) = \{l_{78}, l_{87}\}$ for bin 4. Therefore, $\mathrm{H}^{\mathrm{BPP}}(S) = \{l_{51}, l_{15}, l_{42}, l_{24}, \ l_{96}, l_{93}, l_{63}, l_{69}, l_{39}, l_{36}, l_{78}, l_{87}\}$. If this solution is granted rights for pheromone intensification, the corresponding pheromone values are to be intensified. Note that if W_0 is 10 and the loading influence factor λ is 2, the load fractions of bins 1 to 4 are $((3+5)/10)^2 = 0.64$, $((3+7)/10)^2 = 1.0$, $((2+3+5)/10)^2 = 1.0$, and $((4+5)/10)^2 = 0.81$, respectively. The objective value of the solution is $f^{\mathrm{BPP}}(\mathbf{S}) = \frac{1}{4}(0.64 + 1.0 + 1.0 + 0.81) = 0.86$.

The discussed solution construction method, which was proposed by Levine and Ducatelle (2004), is an iterative approach. Initially, a bin is opened as the *current* bin. The unpacked objects are selected one by one and packed into the bin. The procedure subsequently opens a new bin as the current bin when none of

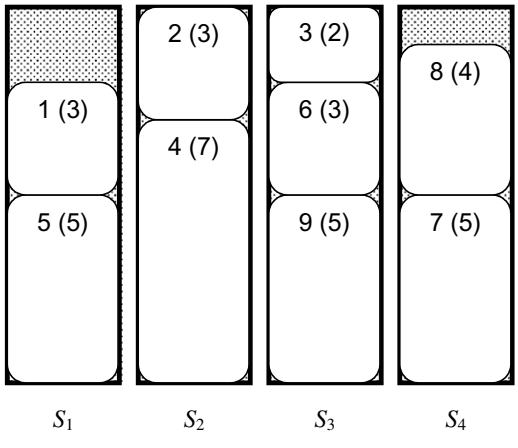

Fig. 1. Schematic diagram of a bin-packing solution.

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

the unpacked objects can be packed into the current bin. They defined the selection probability of choosing object j to pack into the current bin a as

$$P_{aj} = \frac{\tau(a,j)^{\alpha}\eta(j)^{\beta}}{\sum_{k\in N_a}\tau(a,k)^{\alpha}\eta(k)^{\beta}}, \quad \forall j \in N_a, \tag{15}$$

where N_a is the set of unpacked objects whose weights can be added to bin a without exceeding the weight capacity W_0; $\tau(a,j)$ is a pheromone equivalent reflecting the tendency of packing object j into bin a;

$$\tau(a,j) = \begin{cases} \frac{\sum_{k\in S_a}\tau_{kj}}{|S_a|} & \text{if } S_a \neq \varnothing \\ 1 & \text{otherwise,} \end{cases} \tag{16}$$

where S_a is the set of objects that are currently packed in bin a. The heuristic term $\eta(j)$ is defined to prefer an object with a larger weight to fully utilize the weight capacity left in the bin,

$$\eta(j) = \frac{w_j}{W_0}. \tag{17}$$

Levine and Ducatelle applied the traditional pheromone update method for the BPP, such that the pheromone was updated by

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \Delta\tau_{ij};\ \forall i,j;\ \Delta\tau_{ij} = \begin{cases} \frac{\rho}{f^{\mathrm{BPP}}(\mathbf{S}^*)} & \text{if } l_{ij} \in H^{\mathrm{BPP}}(\mathbf{S}^*) \\ 0 & \text{otherwise,} \end{cases} \tag{18}$$

where $\mathbf{S}^*$ is the best solution computed so far. In this paper, we adopt the above solution construction method to solve the BPP while using our own pheromone update method.

Local search methods have been widely used in heuristic methods, such as GA, simulated annealing, and particle swarm optimization method, to enhance solution qualities of larger-scale combinatorial optimization problems. The *k-opt* (especially the 3-*opt*) local search method for the TSP has been adopted by much literature for its effectiveness in helping the search for global optima. One effective local search method for the BPP was proposed by Falkenauer (1996) to first empty objects out of a few bins and then to swap the unpacked objects with packed ones to enlarge load fractions of the packed bins. These methods have been included in a few enhanced ACO methods to claim for their optimality searching capabilities. Generally, without the help of the local search, the global optimal solutions for large-scale combinational optimization problems are nearly unattainable for general heuristic methods, including the enhanced and improved ACO methods.

2.3. Maturity of the pheromone map and search stagnation

Ant-optimization methods face a common challenge of controlling the maturing development of the pheromone map, in order to effectively and efficiently find a near-optimal solution. Faster maturation might easily lead the solution search to a local minimum without exploring the solution space extensively. Slower maturation, however, usually results in an inefficient search whose solution quality is not guaranteed. Therefore, maintaining a *live* pheromone map in the optimizing process before search stagnation is crucial for obtaining a better solution in fewer searches (solution constructions).The pheromone maps of different pheromone update strategies have different value-update behaviors. Some might yield a dominating trail or trails quickly, while others might be able to maintain a *live* (or *immature*) pheromone map or prolong the time to maturity. Solution-search stagnation happens when the pheromone map is dominated by a trail or trails. A dominating trail can be identified by connecting two pheromone items that have the two highest values on each row of the pheromone map. Fig. 2 shows an example of the formation of a dominating trail. Note that the pheromone values in bold face indicate the dominating trail embedded in the graph.

This paper uses the *branch factor* proposed by Stützle and Hoos (2000) to investigate the maturity of the pheromone map. Suppose that z objects define the $z \times z$ pheromone map $\mathbf{T}$ in Eq. (3), where τ_{ij} is the

$$\mathbf{T} = \begin{bmatrix} 0.0 & \mathbf{0.9} & 0.2 & \mathbf{0.8} & 0.1 \\ \mathbf{0.9} & 0.0 & 0.2 & 0.1 & \mathbf{0.7} \\ 0.2 & 0.2 & 0.0 & \mathbf{0.8} & \mathbf{1.0} \\ \mathbf{0.8} & 0.1 & \mathbf{0.8} & 0.0 & 0.1 \\ 0.1 & \mathbf{0.7} & \mathbf{1.0} & 0.1 & 0.0 \end{bmatrix}$$

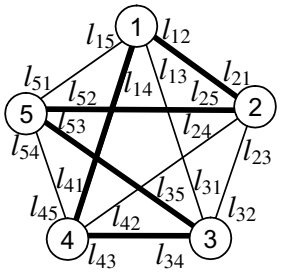

Fig. 2. Formation of a dominating trail on the pheromone map.

pheromone value of the link between objects i and j. A threshold is computed for each row to distinguish larger pheromone values from smaller ones. For row i the threshold is

$$\begin{aligned} &\chi_i = \tau_i^{\min} + \kappa(\tau_i^{\max} - \tau_i^{\min}),\ 0 \leqslant \kappa \leqslant 1, \\ &\tau_i^{\min} = \min_{j=1,2,\ldots,z}\{\tau_{ij}\}, \tau_i^{\max} = \max_{j=1,2,\ldots,z}\{\tau_{ij}\}, \end{aligned} \tag{19}$$

where κ is an interpolation parameter for the threshold. The branch factor of the pheromone map is then defined as

$$b = \frac{\sum_{i=1}^{z}\sum_{j=1}^{z} e_{ij}}{2z}, \quad e_{ij} = \begin{cases} 1 & \text{if } \tau_{ij} \geqslant \chi_i \\ 0 & \text{otherwise .} \end{cases} \tag{20}$$

Notice that when two values are relatively larger than others in each row, they will be filtered out by the threshold, and the branch factor of the map will ideally approach 1.0. Parameter κ affects the evaluation of the number of relative-large values in a row and the resulted branch factor. Although a small value 0.005 of κ was suggested in Stützle and Hoos (2000), in some ACO methods we found it is too hard (tight) to let the branch factor approach 1.0 for maturity recognition. Fortunately, we still can identify the maturity from leveled changes of branch factors. Therefore, we can only state that a pheromone map becomes matured when its branch factor is converged to a positive value. The pheromone map maturing behaviors of various ACO methods are further discussed in Section 4, where numerical data are available for comparison.

3. Superior/Inferior Segment-Discriminated Ant System

This paper proposes an ant-optimization system that uses a segment-based pheromone update strategy, Superior/Inferior Segment-Discriminated Ant System (SDAS). The object selection mechanism of the SDAS is the same as the ACS, where an *exploitation fraction* is defined and a run-time random variable is generated to determine whether using the probability proportional method (a stochastic approach) or deterministically selecting an object j with the highest value of $\tau_{ij}^{\alpha}\eta_{ij}^{\beta}$. Therefore the selected object j^* to succeed object i is

$$j^* = \begin{cases} \arg\max_{j\in N_i}\{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}\} & \text{if } q \leqslant q_0 \\ J & \text{otherwise,} \end{cases} \tag{21}$$

where q is a run-time random number normally distributed within $[0,1]$, q_0 is the exploitation fraction ($0 \leqslant q_0 \leqslant 1$), and J is the object stochastically selected from the probability proportional method whose selection probabilities are defined in Eq. (7).

Instead of using the entire trail to update its pheromone values, the SDAS investigates every segment of the chosen solution trails to build sets of superior and inferior segments. Superior segments merit a pheromone deposit, while inferior ones are subject to pheromone withdrawal. First, the SDAS constructs a superior and an inferior limit for evaluating the performance of ants, using the techniques of control charts. Solution trails constructed by the ants are therefore classified into sets of superior, inferior, and ordinary trails. The

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

segments of the superior and inferior trails are further investigated and classified into superior and inferior segments.

3.1. Evaluation of superior, inferior, and ordinary trails

Most of the existing ACO methods let all ants or a few elite ants deposit pheromone on their trails. This strategy can not sufficiently distinguish good segments from poor ones. For instance, when all ants (including inferior ones) are allowed to deposit pheromone on their trails, the inferior trails might indistinct the pheromone strengthened by the superior ones. Conversely, when only a few elite ants are allowed to deposit pheromone, their trails are not guaranteed to be the truly superior, since some poor segments might be embedded in them. On the other hand, if inferior trails are totally excluded from pheromone deposit and good segments are embedded in these trails, the solution search would not be guided to the global optimum. Therefore, a trail should not be indiscreetly labeled as inferior or superior for pheromone update, we need to investigate more deeply into the segment level.

To overcome the pitfall of treating all segments of an elite trail as good segments, the SDAS focuses on the segments. Pheromone values are not updated on all trails or all elite trails, but on individual segments. Trails are evaluated first to identify superior and inferior trails. Their segments are further examined and extracted to compose a set of superior segments and a set of inferior ones. Moreover, the pheromone-update process is not simply one of depositing (intensifying) the pheromone. Instead, the pheromone value corresponding to a superior segment is intensified, while the value to an inferior segment is reduced. Superior segments earn their rewards and inferior ones incur punishment. Therefore, segments are the focus of the SDAS and are discriminated in pheromone updating.

To evaluate the solution trails, the SDAS first uses the control-chart technique to establish a superior limit F_{best} and an inferior limit F_{worst} on the objective values. Let y be the number of ants deployed and $f_{\max}$, $f_{\min}$, and $\bar{f}$ be the minimum, maximum, and average of the objective values obtained from all the trails:

$$\bar{f} = \frac{1}{y}\sum_{k=1}^{y} f^k = \frac{1}{y}\sum_{k=1}^{y} f^{\text{problem type}}(\mathbf{S}^k), \quad f_{\min} = \min_{k=1,2,\ldots,y}\{f^k\}, \quad f_{\max} = \max_{k=1,2,\ldots,y}\{f^k\}, \tag{22}$$

where $\mathbf{S}^k$ is the solution constructed by ant k and f^k is the objective value of trail k (solution $\mathbf{S}^k$). For minimization problems, the superior and inferior limits

$$F_{\text{best}} = \bar{f} - \omega(\bar{f} - f_{\min}), \tag{23}$$

$$F_{\text{worst}} = \bar{f} + \omega(f_{\max} - \bar{f}), \quad 0 < \omega \leqslant 1.0, \tag{24}$$

where ω is a range factor for the limits. The trails are then divided into sets of superior, inferior, and ordinary trails by these limits. Let U be the superior set of solution trails, $U = \{k | f^k \leqslant F_{\text{best}}, k \in \{1,2,\ldots,y\}\}$, and V the inferior set of solution trails, $V = \{k | f^k \geqslant F_{\text{worst}}, k \in \{1,2,\ldots,y\}\}$. Note that F_{best} approaches $f_{\max}$ and F_{worst} approaches $f_{\min}$ when ω approaches 1.0, where fewer trails are regarded as superior or inferior.

When the superior and inferior trails are identified, superiority or inferiority can be assigned to their segments. In general, a segment of a superior trail is likely to be a *good* segment that makes a positive contribution to the objective value. If a segment appears on both superior and inferior trails, it cannot be classified unambiguously as *good* or *bad*. Therefore, particular operations are required to extract superior and inferior segments from the superior and inferior trails.

3.2. Determination of superior and inferior segments

The pheromone update strategy of the SDAS is a superior-added, inferior-subtracted (SAIS) strategy, discriminating in favor of superior segments and against inferior ones. Superior and inferior segments are determined by a *stochastic set operation*.

Let $\mathbf{S}^*$ represent the best solution so far. The set of corresponding segments to the best solution so far is

$$G = H^{\text{problem type}}(\mathbf{S}^*). \tag{25}$$

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

Applying the segmentation operation on the superior and inferior trails, the segment set of all of the superior trails is

$$A'' = \bigcup_{k \in U} H^{\text{problem type}}(\mathbf{S}^k) \tag{26}$$

and the segment set of all of the inferior trails is

$$B'' = \bigcup_{k \in V} H^{\text{problem type}}(\mathbf{S}^k). \tag{27}$$

The Venn diagram of Fig. 3(a) depicts a general compositional relation between G, A'', and B'', where $\mathbf{L}$ is the set of all segments on the pheromone map. Note that segments belonging to the superior set A'' might also appear in B''. The segments in $A'' \cap B''$ are neither superior nor inferior and they should be excluded from pheromone deposit or deduction. In the SDAS, segments of the best solution so far will be given a certain amount of pheromone separately at the end of the pheromone update procedure. Therefore, they are excluded from the superior and inferior segment sets. Consequently, the set of superior segments that deserve pheromone deposit is

$$A' = A'' - G - B'', \tag{28}$$

as depicted in Fig. 3(b). However, although segments in A' can be regarded as superior segments, a stochastic process is executed for each segment to determine whether it will be finally accepted as a superior segment. Therefore, the superior segment set A is defined as

$$A = \{l_{ij} | l_{ij} \in A', \theta_{ij} < \theta^A\}, \tag{29}$$

where θ_{ij} is a run-time-generated normally distributed random value within $[0, 1]$, and θ^A is a user-specified threshold, $0 < \theta^A \leqslant 1.0$. A conceptual diagram of set A is shown in Fig. 3(c). The definition of Eq. (29) is based on an assumption (or a belief) that the superior segments are usually located on superior trails. Setting a higher θ^A implies that the user has a stronger belief in this assumption. As a result, θ^A can be regarded as the probability that the assumption is true.

Likewise, the inferior segment set B is extracted from B'', whose pheromone might be deducted. At first, segments that appear in both G and B'' are excluded from consideration to avoid subtracting pheromone from segments of the best route so far. Then, similarly, the segments appearing both in B'' and A'' are excluded. Therefore, the segment set

$$B' = B'' - G - A'', \tag{30}$$

as shown in Fig. 3(b). Again, there is an assumption that inferior segments usually come from inferior routes. Therefore, the inferior segment set B is defined as

$$B = \{l_{ij} | l_{ij} \in B', \theta_{ij} < \theta^B\}, \tag{31}$$

where θ^B is the threshold for being inferior, $0 < \theta^B \leqslant 1.0$. A schematic diagram of set B is shown in Fig. 3(c). Consequently, using the presented *stochastic set operation*, the SDAS can extract superior and inferior segments from the trails constructed by the ants, to have a basis for discriminated pheromone update.

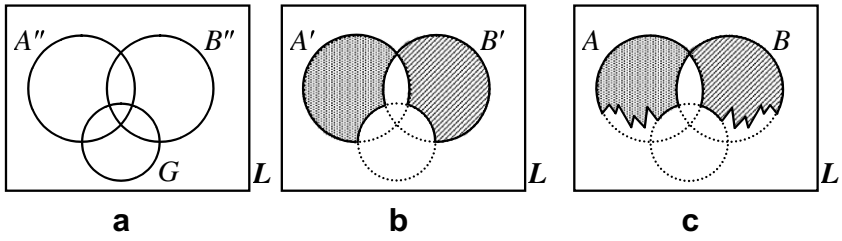

Fig. 3. Schematic diagrams of superior and inferior segment sets.

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

3.3. Pheromone update process of the SDAS

The superior segments in set A receive additional pheromone, given by

$$\Delta\tau_{ij} = \frac{\sigma}{f(\mathbf{S}^*)}, \text{ if } l_{ij} \in A, \tag{32}$$

and the inferior ones lose some of their pheromone, given by

$$\Delta\tau_{ij} = -\sigma \cdot \tau_{ij}, \text{ if } l_{ij} \in B, \tag{33}$$

where σ is the *discrimination factor*, $0 < \sigma \leqslant 1.0$. Note that pheromone update computations in Eqs. (32) and (33) are similar to the pheromone dropping and evaporation computations in traditional ant systems. However, they do not need to be designed in this way; i.e., variations of the SDAS can use different computations to implement the SAIS (superior-added, inferior-subtracted) strategy. The last step of our pheromone update procedure is to evaporate the pheromone of all segments and update the segments on the best solution so far. Therefore, the pheromone of the SDAS is updated by

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \Delta\tau_{ij};\ \forall i,j;\ \Delta\tau_{ij} = \begin{cases} \frac{\sigma}{f(\mathbf{S}^*)} & \text{if } l_{ij} \in A \\ -\sigma \cdot \tau_{ij} & \text{if } l_{ij} \in B \\ \frac{\rho}{f(\mathbf{S}^*)} & \text{if } l_{ij} \in G \\ 0 & \text{otherwise,} \end{cases} \tag{34}$$

where ρ is the *evaporation fraction*, $0 < \rho \leqslant 1.0$.

3.4. Pheromone reset

Since the SDAS allows the best solution so far to add pheromone on its segments to accelerate optimum searching, as shown in Eq. (34), search stagnation might not be completely avoidable for problems having several local optima. When the ants get stuck on a few solutions of local optima, a pheromone *reset* operation can be optionally performed to discard the biased guiding information on the pheromone map. In this operation, the amounts of pheromone on all segments are reset to the initial value τ_0, while the best solution so far is kept invariant. The purpose of pheromone reset is to erase the biased memory that crowds ants together on the same trail or a few trails. The SDAS uses three indices to check for search stagnation: the branch factor of the pheromone map; the number of successive iterations without solution improvement; and the number of successive iterations without any renewal of the iteration-best solution.

The branch factor b is defined in Eq. (20). The SDAS defines a branch factor limit $\bar{b}$ to detect search stagnation. When $b \leqslant \bar{b}$, the pheromone map is regarded as biased by a dominating trail, and search stagnation is assumed. A value for κ of 0.05 and a value for $\bar{b}$ of 1.1 were suggested in Stützle and Hoos (2000). Counters and limits for the numbers of successive iterations without improvements on the iteration-best solution and the best solution so far are implemented to detect for search stagnation. Once stagnation is detected, SDAS resets all values in the pheromone map to the initial value τ_0.

4. Tests for solving the traveling salesman problem

In order to test and compare the performance of the proposed SDAS with other methods, we selected 15 TSPs from the OR-LIB as sample problems, ranging from small to large size. The ACO methods under evaluation included the AS, AS_elite, AS_rank, ACS, and MMAS. We reconstructed these systems following the description in their original literature.

Most of the TSP benchmarks in the OR-LIB have known minimum route lengths. The goal of this test was not to evaluate the capability of finding the minima, but the optimality searching capability. To reveal the intrinsic algorithmic merits of an ACO method, this test simply set an execution time limit for all the methods to solve these 15 problems. Local search was not activated in any method, neither the pheromone-reset operation. For each benchmark, pilot runs were executed to determinate a pertinent execution time limit

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

for all methods. The time limit was set long enough for all methods to unfold their optimality searching capabilities.

The five ACO systems and our SDAS system for the TSP were programmed in C# programming language and executed on the .NET Framework. Coding techniques were impartially implemented for these systems, without particularly favoring any one. All the systems counted the number of route length evaluations for each run, for cross references. The number was actually the number of solution constructions (or searches) in every run, since no any local search was allowed or executed in the test. In addition, when a new shortest route was found, the number of solution constructions was recorded and updated for the measurement of the speediness of route length minimization.

The number of runs on a problem was ranging from 10 to 50, depending on the number of cities defined in the problem. Numerical results were therefore averaged for comparison. For easier comparison on different-size problems, an *error percentage* of the average length relative to the known optimum was calculated as

$$\varepsilon = \frac{\text{the average shortest length-the known shortest length}}{\text{the known shortest length}} \times 100(\%). \tag{35}$$

Since the parameters defined in each ACO method have different functions and effects on its optimality search, their values should be rigorously tuned for solving a particular problem or a particular type of problem. The aim of this test was not to tune the best parameters for each method on each problem to have an absolutely fair and equitable comparison. Therefore, the parameter setting for each method on each problem was simply based on the suggestions or sample values provided in their original literature and the number of cities of the TSP. Table 1 lists the parameter settings on each problem for each method. Note that the execution time and the number of candidate cities were set based on the scale of the problem and the value of the optimum route length. The test was executed on a personal computer with a 3.40 GHz Intel(R) Pentium(R) 4 CPU and 1GB RAM.

The 15 TSPs were separated into three sets. The first set contained *oliver30*, *att48*, *eil51*, *berlin52*, and *rat99*. The numerical results are shown in Table 2. The second set included *kroA100*, *a280*, *lin318*, *att532*, and *rat783*. Their numerical results are listed in Table 3. All the six ACO methods were tested to solve these two sets of benchmarks. The benchmarks in the last set were *pcb1173*, *d1291*, *fl1577*, *pr2392*, and *fl3795*. In this set, to save computation time, only our SDAS, AS, and MMAS were tested for the last two problems. Table 4 lists the numerical results.

Figs. 4 and 5 compare the average error percentages obtained from the six methods for each problem. Note that since minimum lengths of benchmarks *rat99*, *lin318*, *d1291*, and *fl3795* are not available, their error percentages are not available. The averages and standard deviations of the minimum route lengths computed from all methods for each problem are listed in Tables 2–4 for cross references. As shown in Fig. 4, the performance of the SDAS is close to that of the MMAS in solving small-scale TSPs and both methods outperformed other methods. For medium- to large-scale TSPs, the results obtained from the SDAS are better than that from other methods, as shown in Fig. 5.

Notably there are controversies over the rightness of comparison on different heuristics. Computation complexities of exact algorithms can be analytically computed for comparison. However, for stochastic factors and parameters involved algorithms, conducting an absolutely fair and right comparison is an elaborate challenge and an impartial comparison basis is hard to establish. Therefore, the comparison resulted from the test does not suggest that the SDAS outperforms other ACO methods in every aspect. Nevertheless, it indicates that the proposed segment discriminated strategy has a rightful potential for optimality search.

4.1. Discussion on the computational overhead

Referring to Tables 2–4, for each problem, the average numbers of solutions constructed by the six methods within the same execution time limit are not far from each other. The differences between these values were mainly due to the algorithmic complexities and parameter value settings of the ACO methods. Our programming for these systems did not particularly favor any one. In fact, no any method always constructed more or less solutions than others in the same execution time limit for all the problems.

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

Table 1
Parameter settings of tested ACO methods for the 15 TSPs

Parameters	*oliver30*	*att48*	*eil51*	*berlin52*	*rat99*	*kroA100*	*a280*	*lin318*	*att532*	*rat783*	*pcb1173*	*d1291*	*fl1577*	*pr2392*	*fl3795*
Number of runs	50	30	30	30	30	30	30	30	30	30	30	10	10	10	10
Number of candidate cities	10	10	20	20	20	20	20	20	10	10	13	10	20	10	10
Execution time limit (s)	1	6	12	6	15	15	100	200	240	300	600	660	660	1800	2400
Number of ants	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
Pheromone factor α	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Heuristic factor β	3	3	2	3	3	3	3	3	3	3	5	5	3	4	4
Initial pheromone value τ_0	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
SDAS															
Range factor ω	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.8	0.85	0.85	0.85	0.85
Threshold θ^A	0.8	0.8	0.8	0.85	0.9	0.9	0.9	0.9	0.9	0.9	0.7	0.8	0.75	0.75	0.75
Threshold θ^B	0.5	0.5	0.3	0.5	0.7	0.5	0.6	0.6	0.6	0.6	0.4	0.4	0.5	0.5	0.5
Evaporate rate ρ	0.06	0.03	0.1	0.1	0.1	0.1	0.15	0.15	0.25	0.35	0.5	0.5	0.35	0.6	0.6
Discrimination factor σ	0.08	0.15	0.3	0.2	0.28	0.2	0.3	0.3	0.25	0.1	0.3	0.3	0.3	0.4	0.4
Exploitation fraction q_0	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8
AS															
Evaporation rate	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95
Pheromone quantity	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
AS_elite															
Elite number	3	3	3	3	20	20	20	20	20	20	20	5	5	5	5
AS_rank															
Pheromone quantity	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
ACS															
Exploitation fraction	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8
Local update par.	0.06	0.03	0.1	0.1	0.1	0.1	0.15	0.15	0.25	0.35	0.5	0.5	0.35	0.6	0.6
Global update par.	0.08	0.15	0.3	0.2	0.28	0.2	0.3	0.3	0.25	0.1	0.3	0.3	0.3	0.4	0.4
MMAS															
Best probability	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
Swap period	5	5	5	5	5	5	1	1	1	1	1	1	5	5	5

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

Table 2
Computational results for the first set of TSPs

Benchmark (shortest route length)	Methods	Avg. of the best route length	Best route length STDEV	Error percentage (%)	Avg. of the number of solution constructions to find the best route	Avg. of the number of solution constructions
oliver30.tsp (424.869)	SDAS	424.99	0.527	0.029	2148.8	4884.4
	AS	429.52	2.927	1.095	2678.0	4737.6
	ACS	427.59	2.383	0.641	3352.0	4752.0
	MMAS	425.01	0.665	0.034	3312.8	4819.2
	AS_rank	426.38	3.802	0.356	1585.6	4750.4
	AS_elite	425.96	1.110	0.256	3266.0	4652.0
att48.tsp (10628.000)	SDAS	10,665.50	38.938	0.353	6700.0	13,324.7
	AS	11,075.63	74.924	4.212	6010.7	13,137.3
	ACS	11,211.53	104.721	5.491	7606.0	13,232.7
	MMAS	10,663.53	38.248	0.334	7449.3	13,420.0
	AS_rank	10,743.83	81.080	1.090	2854.0	12,965.3
	AS_elite	10,976.30	90.366	3.277	7305.3	13,105.3
eil51.tsp (429.983)	SDAS	432.70	2.682	0.631	13,680.0	17,460.0
	AS	450.37	6.901	4.742	11,388.0	17,500.0
	ACS	457.94	5.439	6.502	8192.0	17,770.0
	MMAS	431.68	2.048	0.395	11,748.0	18,514.0
	AS_rank	439.10	4.496	2.121	4112.0	17,634.0
	AS_elite	444.11	4.206	3.286	11,876.0	17,634.0
berlin52.tsp (7544.366)	SDAS	7575.46	71.568	0.412	4680.0	8613.3
	AS	7743.07	80.183	2.634	5551.3	8467.3
	ACS	7758.10	106.289	2.833	4800.0	8673.3
	MMAS	7571.12	70.583	0.355	6154.7	9039.3
	AS_rank	7634.71	107.760	1.197	3362.0	8584.7
	AS_elite	7631.28	72.755	1.152	5642.0	8464.7
rat99.tsp (n/a)	SDAS	1250.58	15.392	n/a	7093.3	7737.3
	AS	1309.10	11.737	n/a	4856.7	7455.3
	ACS	1322.82	13.048	n/a	3653.3	7727.3
	MMAS	1261.18	21.619	n/a	7214.0	7980.7
	AS_rank	1285.46	14.390	n/a	5754.7	7693.3
	AS_elite	1265.21	16.301	n/a	6651.3	7810.0

In general, the computational overhead of the SDAS is insignificant comparing to others. Note that, all methods, except the AS, sort the objective values of all solutions to rank them or to select elite one or ones. In the SDAS, no significant computation overhead is incurred to identify the inferior and superior solutions. Segmentation operations on those elite solutions are unavoidable for all methods to update the corresponding pheromone items. In the SDAS, an extra flag map was implemented to indicate each segment as a superior segment, an inferior segment, a normal segment, or a segment on the best solution so far, during the segmentation operations. The SDAS algorithm then loops through each pheromone item once to evaporate pheromone and to either subtract or intensify its value according to the flag. Other methods also need to evaporate all pheromone items and to deposit pheromone on the elite trails, where a segment might be computationally traversed many times if it appears in several elite trails. Therefore, extra memory resource is required in the SDAS, while the computational overhead is insignificant.

4.2. Discussion on the pheromone maturing behaviors

One simple way to study the optimization behaviors of an ACO method is to investigate the variation of the computed best route length. Fig. 6 shows an example of the evolutions of the six ACO methods in solving the *a280* benchmark. Fig. 7 displays the branch factors of these runs. For easier and closer comparisons, the figure

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

Table 3
Computational results for the second set of TSPs

Benchmark (shortest route length)	Methods	Avg. of the best route length	Best route length STDEV	Error percentage (%)	Avg. of the number of solution constructions to find the best route	Avg. of the number of solution constructions
kroA100.tsp (21,285.443)	SDAS	21,819.89	247.551	2.511	6708.0	7171.3
	AS	23,034.84	264.405	8.219	3954.7	7150.0
	ACS	23,205.53	257.860	9.021	3535.3	7294.0
	MMAS	21,835.32	318.050	2.583	7353.3	7582.0
	AS_rank	22,498.61	203.694	5.700	5589.3	7104.0
	AS_elite	22,165.35	344.798	4.134	6226.7	7156.0
a280.tsp (2586.770)	SDAS	2856.93	115.661	10.444	7397.3	8015.3
	AS	3067.82	40.085	18.597	4446.7	7966.0
	ACS	2981.63	58.350	15.264	4950.0	8364.0
	MMAS	2915.54	84.239	12.710	8337.3	8584.7
	AS_rank	3027.98	48.254	17.056	5769.3	8008.7
	AS_elite	2952.95	54.730	14.156	7296.7	8427.3
lin318.tsp (n/a)	SDAS	45,582.60	553.542	n/a	11,405.3	12,409.3
	AS	47,523.44	357.726	n/a	7851.3	12,569.3
	ACS	47,539.37	294.375	n/a	5478.0	12,856.0
	MMAS	48,690.05	1625.284	n/a	11,717.3	12,454.7
	AS_rank	46,787.66	448.003	n/a	8086.7	12,505.3
	AS_elite	46,256.99	583.013	n/a	10,002.0	12,779.3
att532.tsp (27,686.000)	SDAS	32,231.50	559.337	16.418	4853.3	5536.7
	AS	33,290.03	263.675	20.241	3900.7	5686.0
	ACS	32,526.90	227.463	17.485	2196.7	5294.0
	MMAS	35,286.83	785.648	27.454	4456.7	5239.3
	AS_rank	32,992.10	278.888	19.165	4033.3	5740.0
	AS_elite	32,611.83	505.634	17.792	4318.7	5284.0
rat783.tsp (8806.000)	SDAS	10,442.04	235.421	18.579	3188.7	3482.0
	AS	10,803.16	61.881	22.680	2884.0	3720.0
	ACS	10,589.86	75.739	20.257	1656.7	3362.7
	MMAS	11,513.48	116.832	30.746	3051.3	3375.3
	AS_rank	10,755.10	81.596	22.134	2650.7	3737.3
	AS_elite	10,714.30	98.052	21.670	2701.3	3411.3

is partially enlarged in Fig. 8. The interpolation factor κ of Eq. (19) was set to 0.005 for branch factor evaluations. The curve distributions shown in these figures are typical for all benchmarks solved in the test.

As depicted in Fig. 6, the SDAS persisted in improving the solution for a long period of time before the solution stagnation happened. In addition, the continuity and depth of route length decreasing were smoother and deeper than others. Among the methods, although the ACS had the fastest length decreasing rate initially, it encountered search stagnation the earliest. The MMAS had a similar behavior as the SDAS but with a lower decreasing rate and a shallow decreasing depth. The AS, AS_rank, and AS_elite shared a similar evolution behavior: the objective value decreases with a different amount in a slower rate than that of the ACS.

Figs. 7 and 8 show the variation of the branch factor for the six ACO methods. For all methods, the branch factor first increases to a maximal value, then continuously decreases, and finally levels off. The pheromone map with the highest branch factor might play an important role in the successive optima search. We regard the time period from the beginning to the time reaching the maximal branch factor as an *information accumulation* stage or *self organizing* stage. During this stage, the map is more gathering the solution construction experience than guiding the solution search. Notice that, the time values elapsed to reaching the maximal branch factor of the SDAS and MMAS are longer than those of the rest. In this case the maximal branch factor for the SDAS is 138.95, which is close to one half of the number of cities, 280, of the *a280* benchmark. This is due to the earlier pheromone deductions on the inferior segments and the value of κ in Eq. (19) was set small (0.005) to have each row threshold close to the minimal pheromone value of the row.

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

Table 4
Computational results for the third set of TSPs

Benchmark (shortest route length)	Methods	Avg. of the best route length	Best route length STDEV	Error percentage (%)	Avg. of the number of solution constructions to find the best route	Avg. of the number of solution constructions
pcb1173.tsp (56,892.000)	SDAS	68,100.33	863.449	19.701	2810.0	3223.3
	AS	71,501.02	755.193	25.679	2594.0	3014.0
	ACS	68,700.09	388.885	20.755	1814.0	3194.0
	MMAS	75,951.74	826.772	33.502	3148.0	3314.0
	AS_rank	71,769.91	394.407	26.151	2760.0	3340.0
	AS_elite	71,521.70	374.366	25.715	2802.0	3340.0
d1291.tsp (n/a)	SDAS	61,363.03	837.163	n/a	2190.0	2870.0
	AS	59,680.17	603.799	n/a	2266.0	2760.0
	ACS	58,232.76	537.835	n/a	2198.0	2984.0
	MMAS	64,409.44	568.871	n/a	2706.0	2962.0
	AS_rank	59,811.86	386.568	n/a	2412.0	2982.0
	AS_elite	59,337.51	434.469	n/a	2498.0	2980.0
fl1577.tsp (22,249.000)	SDAS	26,222.18	302.145	17.858	1454.0	2014.0
	AS	27,309.92	240.343	22.747	1638.0	2100.0
	ACS	26,378.26	273.728	18.559	1386.0	2100.0
	MMAS	31,399.05	395.484	41.126	2014.0	2078.0
	AS_rank	27,571.13	202.879	23.921	1732.0	2094.0
	AS_elite	27,380.95	203.192	23.066	1654.0	2100.0
pr2392.tsp (378,062.826)	SDAS	469,619.20	6717.911	24.217	1700.0	2302.0
	AS	476,091.58	2482.945	25.929	2024.0	2338.0
	MMAS	516,234.22	2585.069	36.547	2254.0	2330.0
fl3795.tsp (n/a)	SDAS	34,696.69	295.095	n/a	818.0	1164.0
	AS	36,886.96	228.992	n/a	1020.0	1188.0
	MMAS	44,197.29	705.204	n/a	1138.0	1200.0

The pheromone map with the highest branch factor starts guiding the search for optimal solutions (local or global ones) with different organizing and adjusting capabilities in different methods. During this *guiding* stage, the branch factor decreases until the pheromone is *matured* with dominated trails. If the map does not bear comprehensive optimality information or the map already has prejudiced items, wider explorations are impossible and the objective improvement is quickly leveled off. This unfavorable case applies to the AS, AS_rank, and AS_elite. As shown in Fig. 8, the branch factor decreasing stage for the SDAS lasts for iterations 55–370 and yields a better final solution than others. The branch factor might not be a good index to reveal the *liveliness* of the pheromone map, yet its variation helps to figure out the maturing status. From

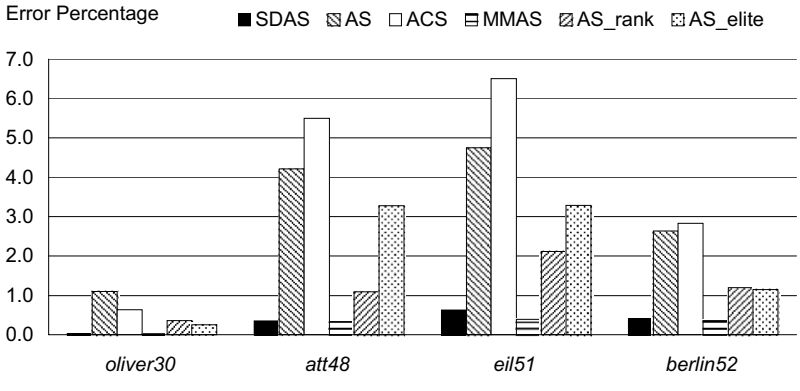

Fig. 4. Error comparisons for small-scale TSPs.

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

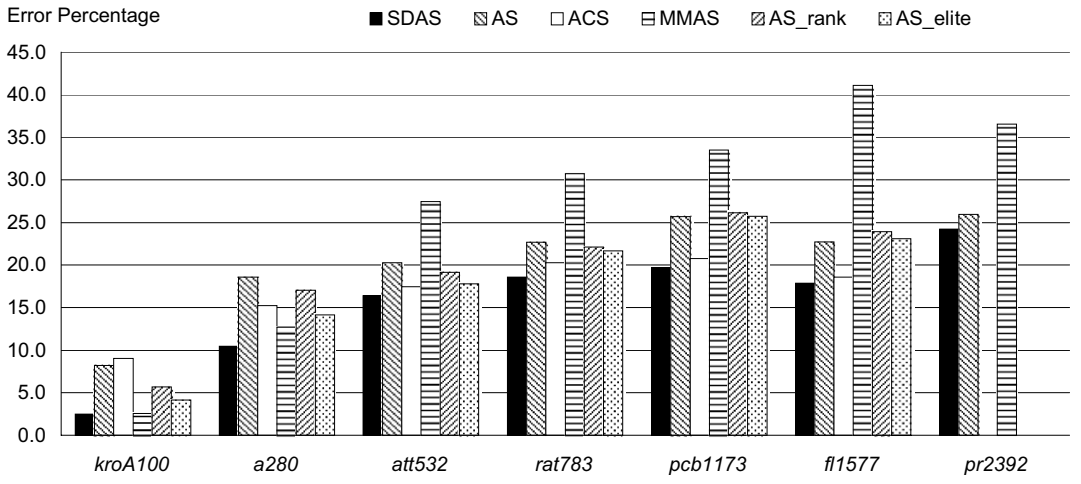

Fig. 5. Error comparisons for medium- to large-scale TSPs.

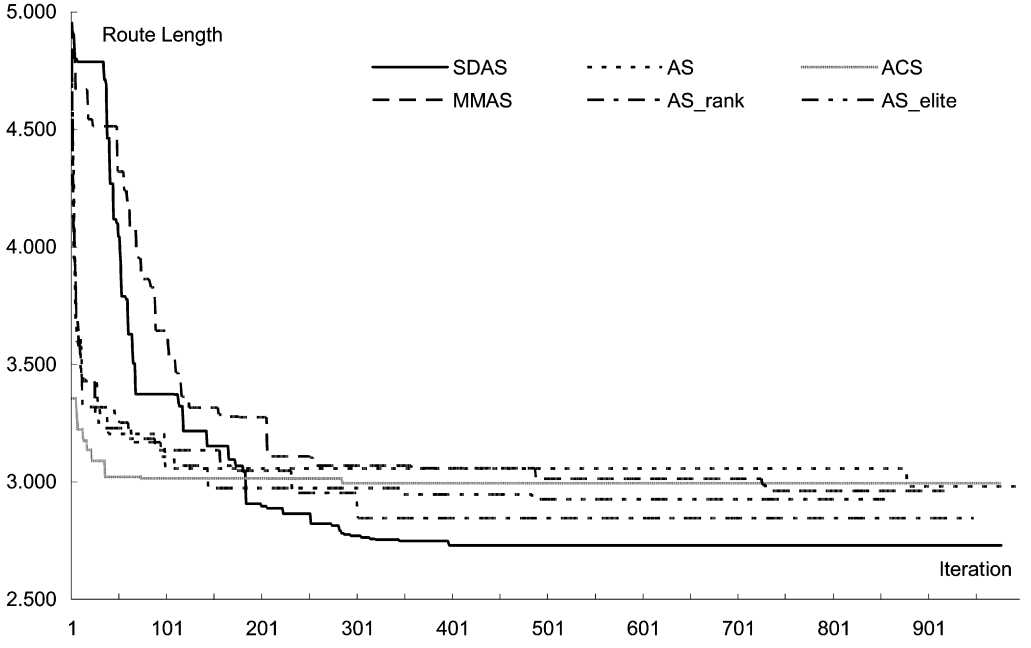

Fig. 6. Route length minimization progresses on benchmark *a280*.

the numerical data obtained in the test, it seems that the pheromone map of the SDAS is more *live* in solution search than others.

5. Tests for solving the bin-packing problem

We tested 80 bin-packing benchmarks from the OR-LIB; they are classified into four sets of problems with different numbers of objects: 120, 250, 500, and 1000. Each set consists of 20 problems that are originally generated stochastically. These problems, defined in files *binpack1*, *binpack2*, *binpack3*, and *binpack4*, were designated and contributed by Falkenauer (1996). The weights of the objects are uniformly distributed in (20,100) to be packed into bins of a weight capacity of 150. Each problem is identified as *u Y_X*, where the *u* stands for

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

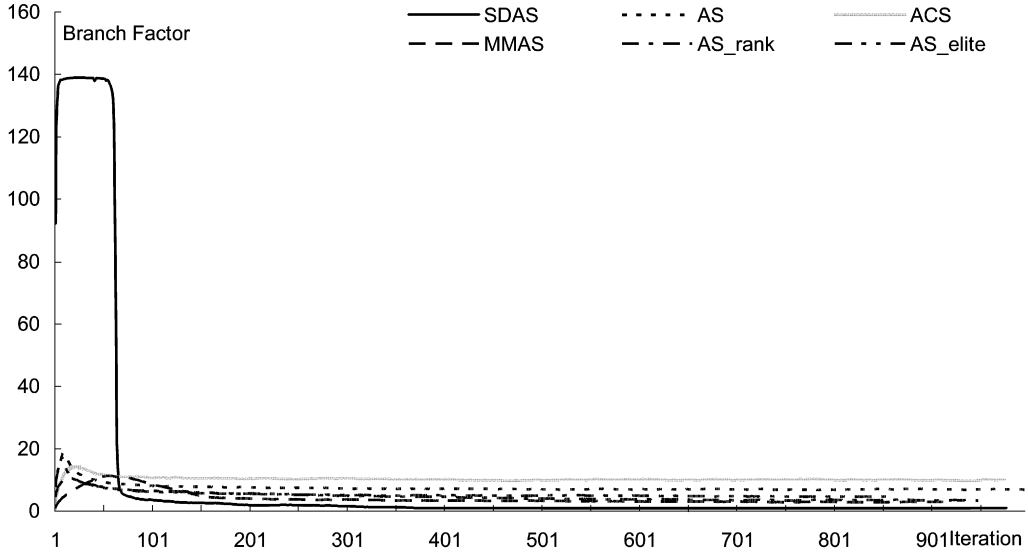

Fig. 7. Comparisons of branch factor convergence on benchmark *a280*.

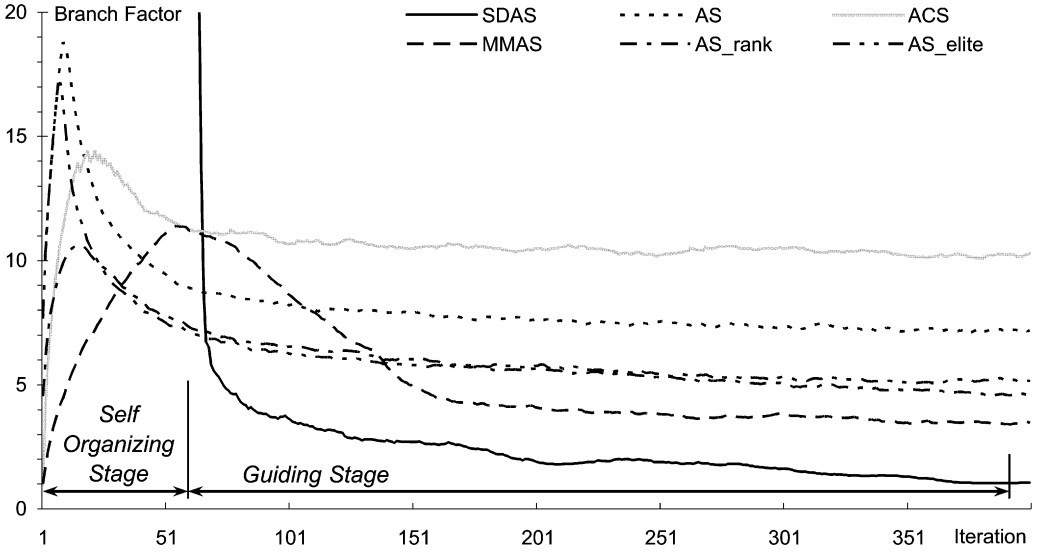

Fig. 8. Comparisons of branch factor convergence on benchmark *a280*.

uniform distribution, the Y is the number of objects, and the X is the serial number in each set; e.g., *u120_00*, *u250_07*, *u1000_19*, etc. In addition, the minimum number of bins for each problem was known and provided in Falkenauer (1996).

We reconstructed all the six ACO systems for the BPP, using the Levine and Ducatelle's ACO method that has been discussed in Section 2.2. Although various local search methods can be used to enhance the BPP solutions, they were not adopted in the test. Some literature claimed optimum was found quicker or easier than other methods, yet the optimality searching capability was not all due to the proposed method, but largely the local search. Therefore, finding the minimum numbers of bins was not the goal of this test; instead, intrinsic solution search capabilities of these ACO methods for the BPP were of interest. The test simply focused on their objective achievements against the same limit of execution time, without any oper-

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

Table 5
Parameter settings for the four categories of BPPs

Parameters	*u120_X*	*u250_X*	*u500_X*	*u1000_X*
Number of runs	10	10	10	2
Number of candidate cities	20	120	240	1200
Execution time limit (s)	120	250	500	1000
Number of ants	20	20	20	20
Pheromone factor α	1	1	1	1
Heuristic factor β	3	3	3	3
Initial pheromone value τ_0	0.05	0.05	0.05	0.05
SDAS				
Range factor ω	0.85	0.85	0.85	0.85
Threshold θ^A	0.8	0.8	0.8	0.8
Threshold θ^B	0.6	0.6	0.6	0.6
Evaporate rate ρ	0.1	0.13	0.15	0.1
Discrimination factor σ	0.15	0.25	0.25	0.1
Exploitation fraction q_0	0.8	0.8	0.8	0.8
AS				
Evaporation rate	0.95	0.95	0.95	0.95
Pheromone quantity	0.01	0.01	0.01	0.01
AS_elite				
Elite number	3	3	3	3
AS_rank				
Pheromone quantity	0.005	0.005	0.005	0.005
ACS				
Exploitation fraction	0.8	0.8	0.8	0.8
Local update par.	0.1	0.13	0.15	0.1
Global update par.	0.15	0.25	0.25	0.1
MMAS				
Best probability	0.05	0.05	0.05	0.05
Swap period	5	5	5	5

ation of local search or pheromone reset. The execution time limit was set to a value that allows all of the methods to unfold their optimality searching capabilities. Table 5 lists the ACO parameters setting for each set of BPPs.

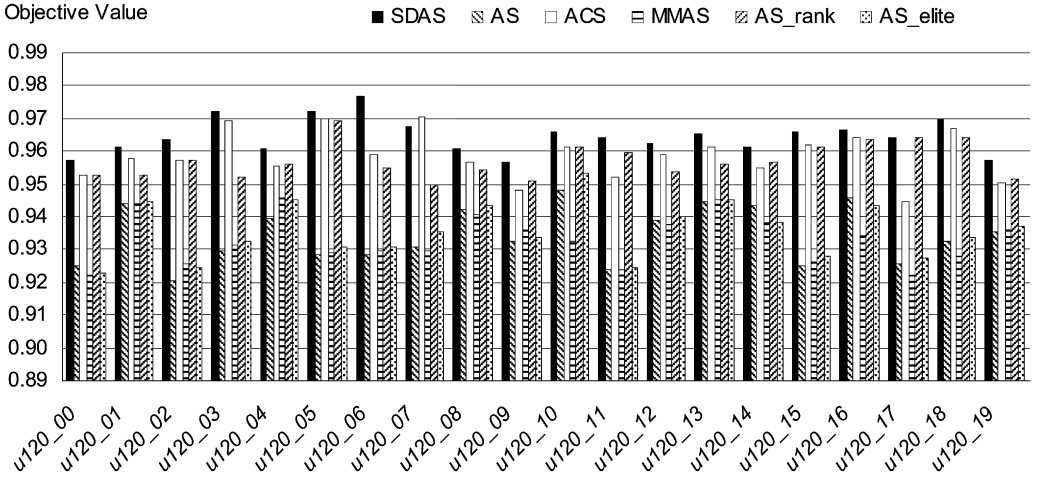

Fig. 9. Objective comparisons between ACO methods for *u120_X* BPPs.

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

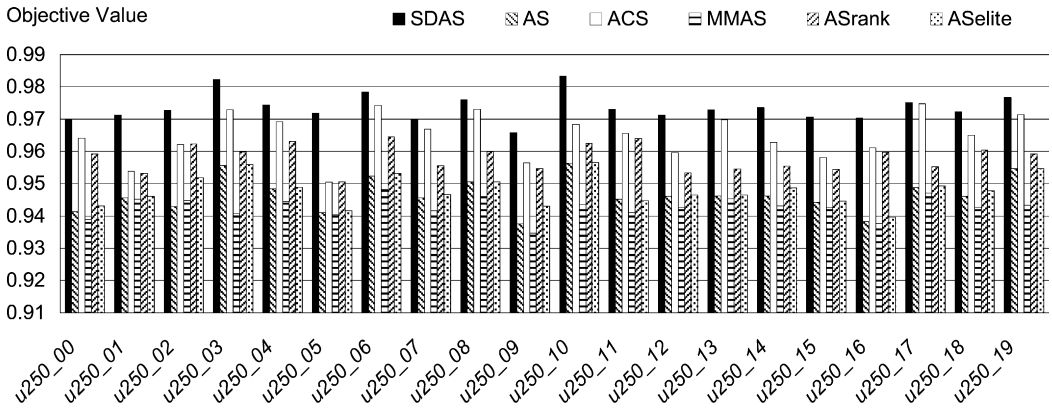

Fig. 10. Objective comparisons between ACO methods for *u250_X* BPPs.

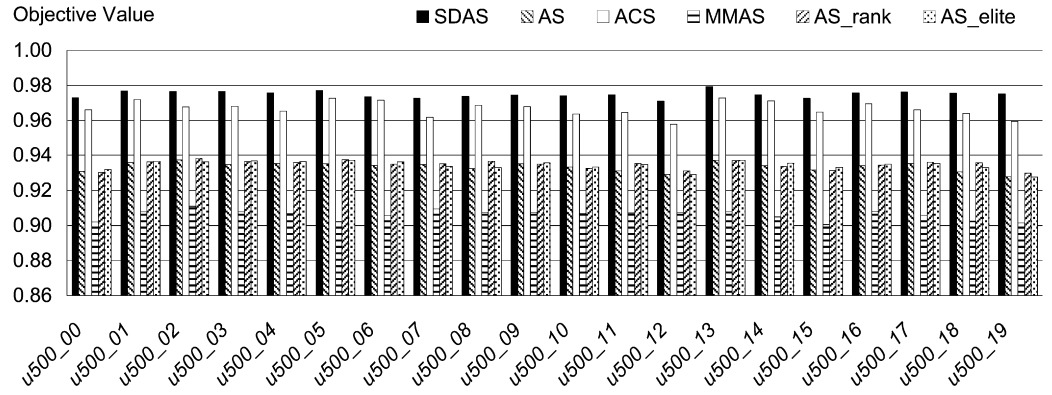

Fig. 11. Objective comparisons between ACO methods for *u500_X* BPPs.

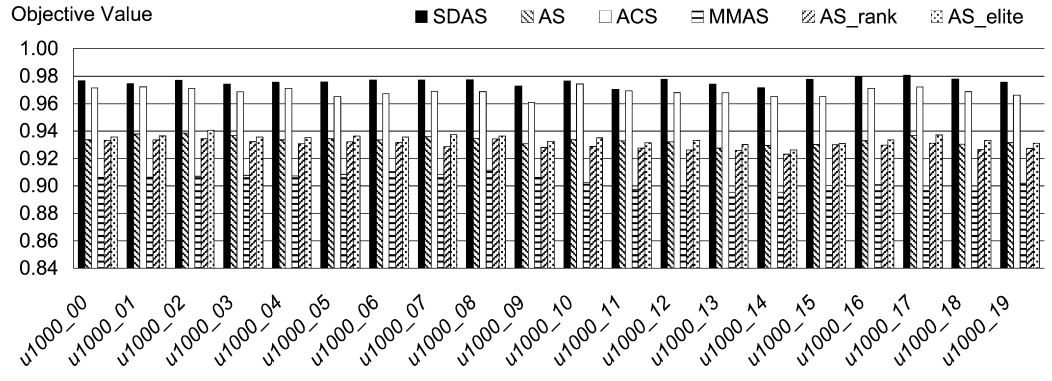

Fig. 12. Objective comparisons between ACO methods for *u1000_X* BPPs.

Figs. 9–12 display the average objective values for each problem obtained from the six ACO methods, where the objective value is calculated by Eq. (12), with λ set to 3.0. In this test, all of the problems were executed 10 times, except the problems in the *u1000_X* set, which were executed only twice for time saving. When we executed these problems, most of the methods could quickly obtain a number of bins close to but not exactly the known minimum. This achievement was mainly due to the effective solution construction

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016

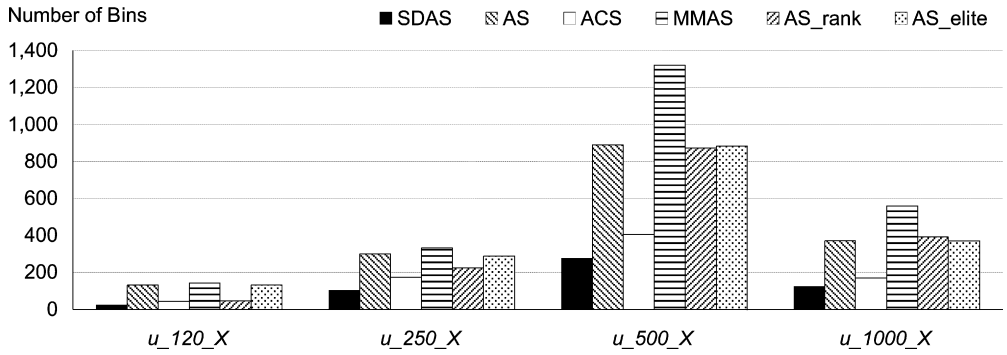

Fig. 13. Total excess number of bins resulted from the ACO methods for the four problem sets.

algorithm. Without the aids from any local search, every method had difficulties in obtaining the minimum number of bins. From the figures shown, our SDAS generated better results than others (except the *u120_07*). Our non-biased pheromone map could prolong the time to search stagnation and guided the computation to better objective values than others. For each run on each benchmark, the number of bins above the known minimum was calculated and cumulated for each set of the BPPs. Fig. 13 compares the total excess numbers of bins on each problem set for the methods. Note that there were 200 runs (20 problems $\times$ 10 runs) executed for the first 3 sets and 40 runs (20 problems $\times$ 2 runs) for the last set. From the comparisons, our SDAS showed a superior optimality searching capability than others.

6. Conclusion

This paper has proposed a segment-based pheromone update strategy for the ACO method. The presented SDAS adopts both pheromone deduction and intensification to attempt to maintain a *live* pheromone map to guide the solution search. Numerical results have shown that this method prolongs the time toward pheromone maturity and the time to search stagnation. In addition, the segment-based optimality information stored in the pheromone map can lead the solution searches to a better one than other methods. Although, it is hard to claim that the SDAS outperforms other ACO methods in every aspect, the proposed pheromone update strategy is worth of further attention.

References

Bullnheimer, B., Hartl, R. F., & Strauss, C. (1999). A new rank-based version of the ant system: A computational study. *Central European Journal of Operations Research and Economics, 7*, 25–38.

Colorni, A., Dorigo, M., Maniezzo, V. (1991). Distributed optimization by ant colonies. In *Proceedings of European conference on artificial life Paris* (pp. 134–142). France.

Dorigo, M., Caro, G. D., & Gambardella, L. M. (1999). Ant algorithms for discrete optimization. *Artificial Life, 5*, 137–172.

Dorigo, M., & Gambardella, L. M. (1997). Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation, 1*, 53–66.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 26*, 29–41.

Dorigo, M., & Stützle, T. (2003). The ant colony optimization metaheuristic: Algorithms, applications, and advances. In F. Glover & G. A. Kochenberger (Eds.), *Handbook of metaheuristics* (pp. 250–285). New York: Springer.

Dorigo, M., & Stützle, T. (2004). *Ant colony optimization*. MIT Press.

Falkenauer, E. (1996). A hybrid grouping genetic algorithm for bin packing. *Journal of Heuristics, 2*, 5–30.

Levine, J., & Ducatelle, F. (2004). Ant colony optimization and local search for bin packing and cutting stock problems. *Journal of the Operational Research Society, 55*, 705–716.

Maniezzo, V. (1999). Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem. *INFORMS Journal on Computing, 11*, 358–369.

Stützle, T., & Hoos, H. H. (2000). Max–min ant system. *Future Generation Computer Systems, 16*, 889–914.

Please cite this article in press as: Yang, F. -C., & Chou, Y. -C., Superior/Inferior Segment-Discriminated Ant System ..., *Computers & Industrial Engineering* (2008), doi:10.1016/j.cie.2007.12.016