Journal of the Chinese Institute of Industrial Engineers

Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/tjci20

AN IMPROVED ANT COLONY SYSTEM ALGORITHM FOR THE VEHICLE ROUTING PROBLEM

Chia-Ho Chen [a] & Ching-Jung Ting [b]

[a] Department of Industrial Engineering and Management , Yuan Ze University , 135 Yuan-Tung Road, Chung-Li, Taoyuan, 320, R.O.C.

[b] Department of Industrial Engineering and Management , Yuan Ze University

Published online: 09 Feb 2010.

To cite this article: Chia-Ho Chen & Ching-Jung Ting (2006) AN IMPROVED ANT COLONY SYSTEM ALGORITHM FOR THE VEHICLE ROUTING PROBLEM, Journal of the Chinese Institute of Industrial Engineers, 23:2, 115-126, DOI: 10.1080/10170660609509001

To link to this article: http://dx.doi.org/10.1080/10170660609509001

AN IMPROVED ANT COLONY SYSTEM ALGORITHM FOR THE VEHICLE ROUTING PROBLEM

Chia-Ho Chen
Department of Industrial Engineering and Management
Yuan Ze University
135 Yuan-Tung Road, Chung-Li, Taoyuan 320, R.O.C.
Ching-Jung Ting*
Department of Industrial Engineering and Management
Yuan Ze University

ABSTRACT

The vehicle routing problem (VRP), a well-known combinatorial optimization problem, holds a central place in logistics management. Many meta-heuristic approaches like Simulated Annealing (SA), Genetic Algorithms (GA), Tabu Search (TS), and Ant Colony Optimization (ACO) have been proposed to solve VRP. Ant Algorithm is a distributed meta-heuristic approach that has been applied to various combinatorial optimization problems, including traveling salesman problem, quadratic assignment problem. In this research, we proposed an improved ant colony system (IACS) algorithm that possesses a new state transition rule, a new pheromone updating rule and diverse local search approaches. The computational results on 14 VRP benchmark problems show that our IACS yields better solutions than those of other ant algorithms in the literature and is competitive with other meta-heuristic approaches in terms of solution quality.

Keywords: vehicle routing problem, meta-heuristic, ant colony system

1. INTRODUCTION

The vehicle routing problem (VRP) holds a central place in logistics management. This problem involves the design of a set of minimum cost delivery routes, start and end at a depot, which serves a set of customers. Each customer must be supplied exactly once by one vehicle. Each vehicle must not exceed its capacity and a pre-specified route length. The VRP is a well-known NP-hard problem [18] that is very difficult to solve to optimality. Exact methods like Dynamic Programming and Branch and Bound cannot obtain the optimal solution for large VRP within reasonable time, thus, many researchers have used heuristic approaches to solve the VRP. Many meta-heuristic approaches developed according to artificial intelligence, biological evolution and/or physics phenomenon have been reported and applied to the VRP, such as Simulated Annealing (SA) [1, 26, 27, 33], Genetic Algorithms (GA) [2, 23], Tabu Search (TS) [19, 26, 28, 31, 34], Neural Network [20, 30] and Ant Colony Optimization (ACO) [3, 4, 15].

Among these meta-heuristic approaches, Ant System (AS) is a new distributed meta-heuristic first introduced by Colorni et al. [8]. The AS is based on the behavior of real ants searching for food. Real ants communicate with each other using an aromatic essence, called pheromone, that they laid down on the path they traversed. The pheromone will accumulates when more and more ants pass through the same path. Nevertheless, the pheromone will evaporate if no ants continue to pass. The selection of the pheromone trail reflects the length of the paths as well as the quality of the food source found. Dorigo et al. [13] reported the AS to solve traveling salesman problem (TSP), quadratic assignment problem (QAP) and job-shop scheduling. Dorigo and Gambardella [11, 12] developed the ant colony system (ACS) to improve the performance of AS. They used a different state transition rule and added a local pheromone updating rule.

Ant algorithms have been applied to many combinatorial problems successfully, including traveling salesman problem [5, 11, 12, 13], quadratic assignment problem [13, 17, 25, 29, 32], job-shop scheduling [9, 13], vehicle routing problem [3, 4, 15], sequential ordering problem [14] and graph coloring problem [10]. Bullnheimer et al. [3] were the first researchers that used AS to solve the VRP. They presented a hybrid Ant System algorithm (HAS) that added the 2-opt heuristic and then based on Saving Algorithm to construct routes. However, the results of

* Corresponding author: ietingcj@saturn.yzu.edu.tw

HAS were not as good as other meta-heuristic approaches. Then, Bullnheimer et al. [4] developed an improved AS (IAS) for the VRP. They applied the idea of candidate lists [11, 12] to construct vehicle routes. Candidate lists can concentrate the search on promising nodes thus saving computational effort that can be better used for further iterations. Results of a set of standard problems showed that IAS was significantly better than AS and outperformed SA and Neural Network. Gambardella et al. [15] defined a hybrid Ant System algorithm for the VRP (HAS-VRP), which was inspired by ACS. Results obtained by HAS-VRP were competitive with those of the best-known algorithms and new upper bounds had been found for well-known problem instances. Furthermore, Gambardella et al. [16] proposed a multiple Ant Colony System to vehicle routing problem with time windows (MACS-VRPTW) and improved some of the best-known solutions in the literature. In MACS-VRPTW, one colony minimized the number of vehicles while another colony minimized the traveled distances.

The rest of the paper is organized as follows. Section 2 describes the improved ant colony system that incorporates with a new state transition rule, a new pheromone updating rule and diverse local search approaches: 2-opt [24] and swap. Computation results on fourteen VRP benchmark problems with IACS and comparison against other ant-based algorithms and meta-heuristics are reported in section 3. Finally, concluding remarks are made.

2. THE IMPROVED ANT COLONY SYSTEM

Our Improved ACS (IACS) algorithm is based on the ACS algorithm that proposed by Dorigo and Gambardella [12]. In ACS, m ants are initially positioned on n vertices chosen according to some a priori assignment procedure (e.g., randomly). Each ant builds a tour by repeatedly applying a probabilistic nearest neighbor heuristic. While constructing a tour, an ant modifies the pheromone level on the visited edges by applying a local updating rule. When all ants have completed their tours, the pheromone level on each edge is modified again by applying the global updating rule which favors the edges associated with the best tour found from the start. Our IACS modifies the local and global pheromone updating rules and adds a local search on one ant. The IACS mainly consists of the iteration of the following three steps.

Step 1: Each ant builds the solution independently and carries out local pheromone update.

Step 2: Apply the local search to improve the solution.

Step 3: Update the global pheromone information.

In the following sections, we describe each step in more details and the flowchart of the IACS is shown in Figure 1.

2.1 Route Construction

In the original ACS, an ant, say k, moves from the present node i to the next node v according to the state transition rule given by (1).

$$v = \begin{cases} \arg \max\limits_{j \in U_k} \left[(\tau_{ij})(\eta_{ij})^{\beta} \right] & q \le q_0 \\ V & q > q_0 \end{cases} \tag{1}$$

$$V : P_{ij} = \frac{(\tau_{ij})(\eta_{ij})^{\beta}}{\sum\limits_{j \in U_k} (\tau_{ij})(\eta_{ij})^{\beta}} \tag{2}$$

where U_k is the set of nodes that remain to be visited by ant k positioned on node i, τ_{ij} is the pheromone level on edge (i, j), η_{ij} is the inverse of the length of edge (i, j). Here we define η_{ij} as the savings of combining two nodes i and j on one tour as opposed to serving them on two different tours. Thus, η_{ij} is calculated as follows:

$$\eta_{ij} = d_{i0} + d_{0j} - d_{ij} \tag{3}$$

where d_{ij} denotes the distance between nodes i and j, and node 0 is the depot; and β is the parameter that determines the relative influence of pheromone versus distance savings ($\beta > 0$). Moreover, q is a random number uniformly distributed in [0, 1], and q_0 is a pre-defined parameter ($0 \le q_0 \le 1$). This state transition rule favors transitions toward nodes connected by short edges and with a large amount of pheromone. If $q \le q_0$ then the best next node v, according to Eq. (1) (exploitation), is chosen, otherwise, a node is chosen according to Eq. (2) (exploration). If the vehicle capacity constraint is met, the ant will return to the depot before selecting the next node. This selection process continues until each node is visited and the tour is complete.

To keep the information of the best found solution, we adopt the concept similar to the elitism of the genetic algorithm to preserve the best ant of a generation into the next generation. As a consequence, the pheromone of the edges belonging to the global best solution will be reinforced at least twice to speed up the search toward the better solution. Thus, we only reconstruct m - 1 solutions (m is the number of ants) in each generation.

2.2 Local Search

In the original ACS, after the ants have constructed their solutions but before the pheromone is local updated, each ants' solution is improved by applying a local search. However, local search is a time-consuming procedure of ACS. To save the computation time, we will only apply local search to the best S ants (S = 1 in this paper) among those m-1 ants built in this iteration. The idea here is that better solution may have better chance to find a local optimum via local search. We first apply a local search based on swap moves to the ant. Following Bullnheimer et al. [4] we then apply the 2-opt algorithm.

2.3 Pheromone Update

In ant algorithms, the pheromone of all edges belonging to the route obtained by ants will be updated. The pheromone updating of ACS includes local and global updating rules. The local updating rule of IACS in Eq. (4) is applied to change pheromone level of edges after an ant completes its route.

$$\tau_{ij}^{new} = \tau_{ij}^{old} + \rho\tau_0 \tag{4}$$

where $0 \le \rho \le 1$ is a user-defined parameter called evaporation coefficient, and $\tau_0 = (n \times L_{nn})^{-1}$ is the initial pheromone level of edges, where n is the number of nodes and L_{nn} is the tour length produced by the Nearest Neighbor heuristic. If the best solution till now does not improve within a given number of generations (20 generations in this study), the pheromone level of each edge is reset to the initial pheromone level, τ_0.

Dorigo et al. [13] and Bullnheimer et al. [5] used the elitist strategy on the trail updating in ant system. The idea of the elitist strategy in the context of the ant system is to give extra emphasis to the best path found so far after every iteration. Such a strategy will direct the search of all the other ants in probability toward a solution composed by some edges of the best tour itself. In our IACS, the best elitist ants of the iteration, including the global-best and iteration-best ants, are allowed to lay pheromone on the arcs they traversed. The idea here is to balance between exploitation (through emphasizing global-best ant) as well as exploration (through the emphasis to iteration-best ant). The global updating rule is described as follow:

$$\tau_{ij}^{new} = (1-\gamma)\tau_{ij}^{old} + \gamma\Delta\tau_{ij} \tag{5}$$

where $0 \le \gamma \le 1$ is a user-defined parameter, and

$$\Delta\tau_{ij} = \frac{(L_3 - L_g) + (L_3 - L_l)}{L_3} \tag{6}$$

where L_g and L_l denote the tour length of global-best solution and iteration-best solution respectively, and L_3 is the 3rd best solution at current iteration. Edges do not belong to the global-best solution and iteration-best solution just lose pheromone at the rate γ, which constitutes the trail evaporation. This choice is intended to make ants search in a neighborhood of the two best tours instead of the globally best tour to avoid the algorithm being trapped in a local optimum without finding very good solutions.

2.4 Overall Procedure

The procedures of our IACS are described as follows:

Step1: Set parameters.
Step2: Generate an initial solution using Nearest Neighbor heuristic.
Step3: Apply the local search (2-opt and Swap) to the initial solution and let it to be the solution 1 of population. g = 1, h = 2.
Step4: Construct solutions base on the route construction rule and progress local pheromone update by Eq. (4). h = h + 1.
Step5: If h > m, then h = 2 and go to Step 6. Otherwise, go to Step 4.
Step6: Sort the solutions 2~m in ascending order and apply local search (2-opt and Swap) to the 2nd solution.
Step7: Apply the global pheromone update rule by Eq. (5).
Step8: Record the best solution so far and let it to be the solution 1 in the next generation. g= g + 1.
Step9: If the stopping criterion (maximum number of generations, G, in this paper) is met, then stop and output the best solution. Otherwise, go to Step 4.

3. NUMERICAL ANALYSIS

In this section, fourteen VRP benchmark problems described in Christofides et al. [6] are tested by our IACS. We compare the computational results with those of two published AS [3, 4] as well as different meta-heuristics such as SA [26, 33], TS [19, 34] and GA [2]. These problems contain between 50 and 199 customers as well as the depot. The customers in problems 1-10 are randomly distributed in the plane, while they are clustered in problems 11-14. Problems 1-5 and 6-10 are identical, except that the total length of each vehicle route is limited for the latter problems. Problems 13-14 are the counterparts of problems 11-12 with additional route length constraint. In addi-

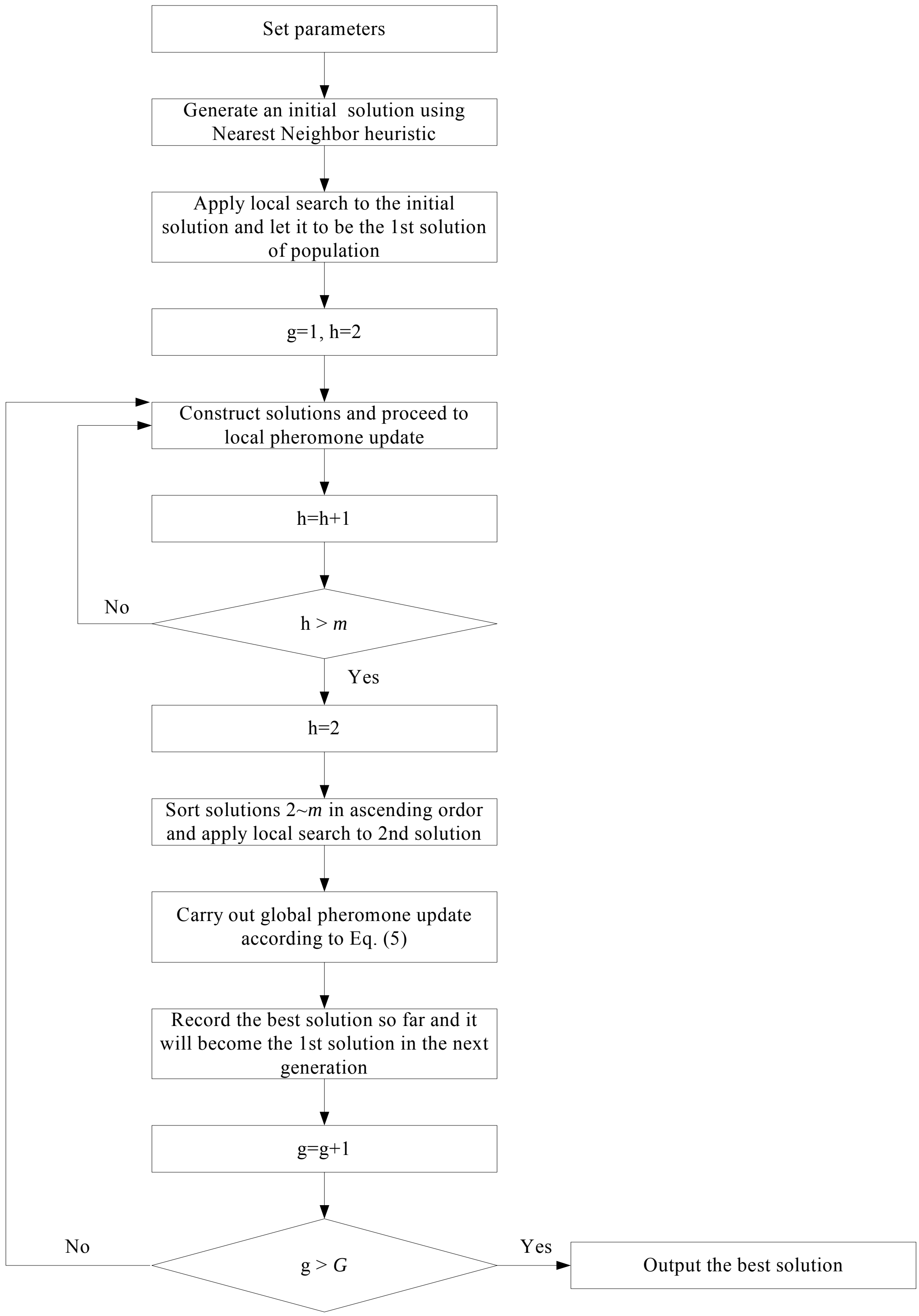

Figure 1. Flowchart of the IACS

tion, the best known solutions of C1 and C12 had been proved to be optima [21, 22]. Information on these instances is summarized in Table 1. Columns 2-7 show the problem size n, the vehicle capacity Q, the service times s, the maximum route length T, the best known solutions BKS, the authors and the methods used by the authors.

Table 1. VRP benchmark problems

NO.	*N*	*Q*	*s/T*	BKS	Reference	Method
C1	50	160	0/∞	524.61	Taillard (1993)	TS
C2	75	140	0/∞	835.26	Taillard (1993)	TS
C3	100	200	0/∞	826.14	Taillard (1993)	TS
C4	150	200	0/∞	1028.42	Taillard (1993)	TS
C5	199	200	0/∞	1291.45	Rochat and Taillard (1995)	TS
C6	50	160	10/200	555.43	Taillard (1993)	TS
C7	75	140	10/160	909.68	Taillard (1993)	TS
C8	100	200	10/230	865.94	Taillard (1993)	TS
C9	150	200	10/200	1162.55	Taillard (1993)	TS
C10	199	200	10/200	1395.85	Rochat and Taillard (1995)	TS
C11	120	200	0/∞	1042.11	Taillard (1993)	TS
C12	100	200	0/∞	819.56	Taillard (1993)	TS
C13	120	200	50/720	1541.14	Taillard (1993)	TS
C14	100	200	90/1040	866.37	Taillard (1993)	TS

3.1 Computational Results

The IACS is coded in Borland C++ Builder 5.0 and executed on a PC equipped with 128 MB of RAM and an Intel Pentium Ⅲ processor running at 1000 MHz. In the preliminary experiment, the parameter settings used for IACS are $m = n/10$, $\beta = 4$, $\rho = \gamma = 0.5$, $q_0 = 0.5$, $S = 1$, and $G = 2n$. We will test several values for each parameter while all the others are held constant in the following sections. We summarize the results that include the best, the worst, average solutions, standard deviation (std. dev.) and average computational time over 10 runs for each problem in Table 2. We find the best known solutions in 6 out of 14 problem instances.

Table 2. The computational results of IACS

NO.	Best	Worst	Average	Std. Dev.	Time (sec.)
C1	**524.61**[a]	536.13	528.17	4.05	3
C2	836.18	858.13	846.90	5.45	26
C3	835.6	854.00	844.00	5.72	101
C4	1038.22	1083.90	1057.12	13.80	617
C5	1327.07	1367.00	1347.63	12.10	3080
C6	**555.43**	567.75	563.17	4.36	5
C7	**909.68**	949.42	928.85	12.10	41
C8	**865.94**	891.83	880.33	8.00	115
C9	1173.76	1193.70	1185.98	7.18	853
C10	1413.83	1496.30	1449.25	22.70	4223
C11	**1042.1**	1051.00	1046.61	3.19	204
C12	832.67	850.52	842.43	5.66	88
C13	1547.07	1575.00	1558.21	8.21	428
C14	**866.37**	869.15	867.89	1.00	125
Avg.	983.47	1010.3	996.18	-	708

[a]: the solution equals to the best-known solution.

3.2 Computation Efficiency

We present the overall average relative percentage deviation (RPD) from the best known-solutions of 14 benchmark problems at different iterations in Figure 2. As can be observed in Figure 2, the average RPD is reduced from 10% in the initial solution to 2% within the first $n/2$ iterations. The increment of improvement of RPD decreases as the iteration progresses. This indicates that IACS spends more than half of the computational time only to improve the RPD from 2% to 0.66%.

3.3 Parameter Sensitivity

In this section, we discuss the impact of different parameters of IACS. We test different values for each parameter on those problems that we already found the best known solutions (C1, C6, C7, C8, C11, and C14) and summarized the average objective function value, the average computational time in seconds over ten runs. The best average tour length

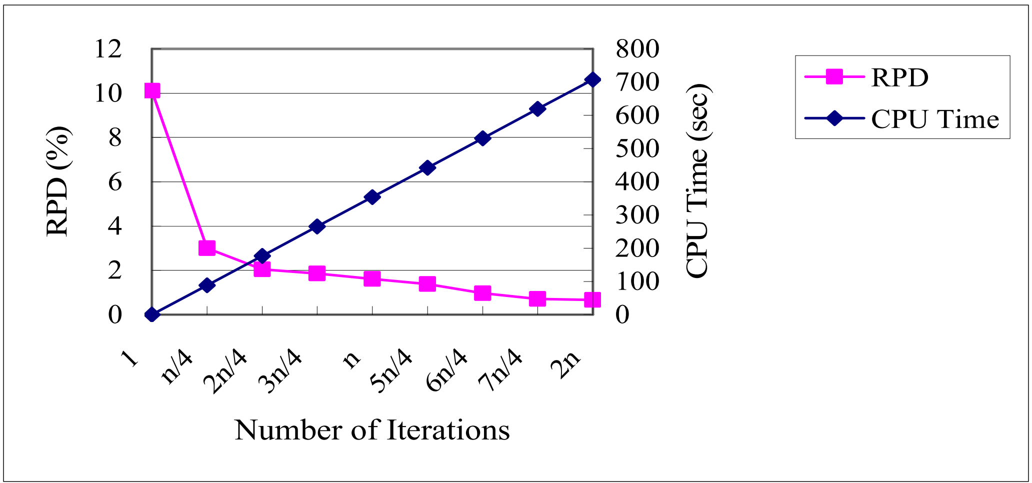

Figure 2. Average RPD and computation time at different iterations

for each problem is in boldface. When the value of one parameter is tested, the others are set at their default values in the preliminary experiment. The values tested were: $\beta \in \{1, 2, 3, 4, 5\}$, $\rho \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$, $\gamma \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$, $q_0 \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$, $m \in \{n/10, n/5, n/3, n/2, n\}$, and $S \in \{1, 2, 3, 4\}$.

Table 3 presents the impact of parameter β on solution quality. β is the parameter that determines the relative effect of pheromone versus distance savings. When β increases the ant favors larger savings value. Nevertheless, if the high value of β is used, the IACS will be easily trapped in a local optimum. A larger number of best results are obtained with values around $\beta = 4$.

Table 3. Impact of parameter β on solution quality

Problem	**1**	**2**	**3**	**4**	**5**
C1	536.76[a]/2.3[b]	532.68/2.3	533.24/2.3	**528.17**[c]/2.6	531.45/2.6
C6	568.63/3.9	566.38/4.0	564.46/4.4	**563.17**/4.0	564.78/4.4
C7	938.79/26.6	931.30/29.7	926.78/29.6	**922.45**/33.8	931.09/36.2
C8	886.89/97	885.10/104	882.70/102	**877.29**/116	881.44/125
C11	1045.20/167	1046.27/195	**1044.57**/217	1044.91/207	1045.18/246
C14	**867.13**/116	867.34/107	867.41/117	867.73/122	867.37/118
Avg.	807.23/69	804.85/74	803.19/79	**800.62**/81	803.55/89

[a]: the average objective value
[b]: the computational time (in seconds)
[c]: the best average tour length

Table 4 shows the impact of parameter q_0 on solution quality. Through this parameter, it determines the relative importance of exploitation versus exploration. With high value of q_0, the IACS tends to exploit the neighbor solution space of the previous best solution. The smaller the q_0 is, the higher the probability to use the probabilistic rule described with Eq. (1). Therefore, the ant performs a biased exploration of the arcs and will explore the new possible solution space. Better solutions are found with smaller values of q_0 between 0.1 and 0.5.

Table 5 reports the impact of parameter ρ on solution quality. This parameter is used in the local updating rule, when some pheromone is added to the edges of the current tour to exploit the search. Higher values of ρ may lead to early converge in the local optimum as shown in table 5.

Table 6 shows the impact of parameter γ on solution quality. The deposited pheromone is discounted by a factor γ, this results in the new pheromone trail being a weighted average between the old pheromone value and the amount of pheromone deposited. Clearly, the best results are obtained with values around $\gamma = 0.1$. This makes sense, as larger values tend to strongly bias the search toward the best solutions and lead to premature convergence.

Table 7 presents the impact of parameter m on solution quality. We find that increasing the number of ants generally increases the computational times. However, the objective function value is not necessary improved as shown in Table 7. Clearly, the best results are obtained with values around $m = n/10$.

Table 4. Impact of parameter q_0 on solution quality

Problem	0.1	0.3	0.5	0.7	0.9
C1	530.31[a]/2.7[b]	**527.13**[c]/2.5	528.17/2.6	531.62/2.3	535.03/2.3
C6	563.80/4.6	565.96/4.2	**563.17**/4.0	564.55/4.0	566.04/4.0
C7	926.63/40.8	933.64/31.9	**922.45**/33.8	931.56/31.8	932.38/28.4
C8	885.01/130	882.73/122	**877.29**/116	879.67/99	885.77/91
C11	1044.91/317	**1044.86**/257	1044.91/207	1045.54/191	1047.36/168
C14	**867.21**/138	867.79/127	867.73/122	867.46/118	867.79/96
Avg.	802.98/106	803.69/91	**800.62**/81	803.40/74	805.73/65

[a]: the average objective value
[b]: the computational time (in seconds)
[c]: the best average tour length

Table 5. Impact of parameter ρ on solution quality

Problem	0.1	0.3	0.5	0.7	0.9
C1	530.04[a]/2.5[b]	533.66/2.6	**528.17**[c]/2.6	532.07/2.4	531.13/2.6
C6	565.16/4.4	565.67/4.2	**563.17**/4.0	563.22/4.2	563.94/4.4
C7	**922.00**/31.9	927.41/31.2	922.45/33.8	925.86/34.0	928.40/31.2
C8	878.98/114	883.75/111	**877.29**/116	879.86/111	884.22/103
C11	**1044.02/233**	1045.36/233	1044.91/207	1046.90/241	1044.68/226
C14	**867.10/118**	867.55/119	867.73/122	867.45/118	867.12/112
Avg.	801.22/84	803.90/84	**800.62**/81	802.56/85	803.25/80

[a]: the average objective value
[b]: the computational time (in seconds)
[c]: the best average tour length

Table 6. Impact of parameter γ on solution quality

Problem	0.1	0.3	0.5	0.7	0.9
C1	**526.65**[c]/3.1[b]	532.35[a]/2.7	528.17/2.6	533.68/2.3	534.08/2.1
C6	**562.53**/5.4	564.44/4.8	563.17/4.0	564.08/4.3	564.41/4.0
C7	923.07/41.1	927.63/36.3	922.45/33.8	931.82/30.0	925.47/28.0
C8	877.59/142	883.65/118	**877.29**/116	881.91/104	885.46/94
C11	**1044.64**/311	1046.92/252	1044.91/207	1045.18/207	1046.14/187
C14	867.62/136	**867.58**/125	867.73/122	**867.58**/111	867.86/106
Avg.	**800.35**/106	803.76/90	800.62/81	804.04/76	803.90/70

[a]: the average objective value
[b]: the computational time (in seconds)
[c]: the best average tour length

Table 7. Impact of parameter m on solution quality

Problem	***n*/10**	***n*/5**	***n*/3**	***n*/2**	***N***
C1	**528.17**[c]/2.6[b]	533.34[a]/2.5	538.18/2.6	541.59/3.1	538.13/4.3
C6	**563.17**/4.0	567.46/3.8	564.99/3.9	569.98/4.1	568.40/5.3
C7	**922.45**/33.8	928.24/29.3	934.16/28.3	934.81/27.8	934.49/34.8
C8	**877.29**/116	883.03/90	883.13/83	884.46/90	886.40/111
C11	**1044.91**/207	1045.58/203	1046.27/204	1047.16/210	1047.14/291
C14	867.73/122	**867.35**/94	867.55/86	867.91/100	867.52/132
Avg.	**800.62**/81	804.17/70	805.71/68	807.65/73	807.01/96

[a]: the average objective value
[b]: the computational time (in seconds)
[c]: the best average tour length

Table 8 presents the impact of S on solution quality. The solutions are clearly improved with more ants go through the local search. However, the objective function value is not reduced significantly while the computational time is significant longer for higher value of S. Figure 3 illustrates the objective function value and run time at different S in problem C14. The run time for S = 4 is over four times of that for S = 1, but the objective value only is reduced by 0.94. This leads to our intension to use S = 1 for IACS.

Table 8. Impact of S on solution quality

Problem	1	2	3	4
C1	528.17[a]/2.6[b]	529.02/6.6	527.61/10.9	**527.44**[c]/15.2
C6	563.17/4.0	563.75/8.9	**563.00**/14.2	563.21/19.6
C7	922.45/33.8	922.46/72.9	921.13/120	**918.46**/168
C8	877.29/116	878.36/197	880.52/321	**877.06**/447
C11	1044.91/207	1044.41/480	1043.23/743	**1042.98**/1074
C14	867.73/122	867.26/239	867.29/395	**866.79**/544
Avg.	800.62/81	800.88/167	800.46/267	**799.32**/378

[a]: the average objective value
[b]: the computational time (in seconds)
[c]: the best average tour length

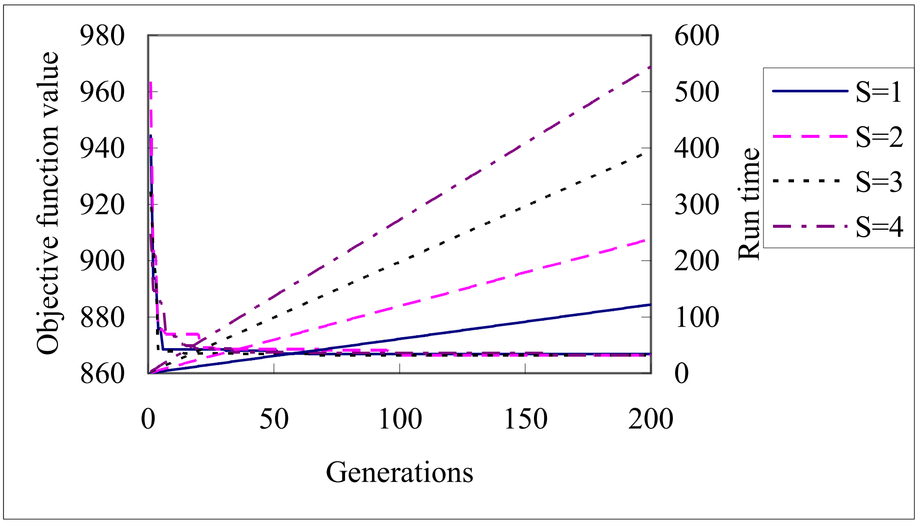

Figure 3. The objective function value and run time at different S for problem C14

Given the results in the previous tables, the values $\beta = 4$, $q_0 = 0.5$, $\rho = 0.5$, $\gamma = 0.1$, $m = n/10$ and $S = 1$ are chosen. The results obtained with this combination of parameters on the 14 problems are shown in table 9. Compared with the quality of the solutions in table 2, it appears that no significant undesirable cross-effects are observed with this combination of values.

Table 9. The computational results at best combination of parameters

NO.	Best	Worst	Average	Std. Dev.	Time (sec.)
C1	**524.61**[a]	535.28	526.65	4.16	3
C2	835.32	860.10	850.27	8.28	26
C3	831.20	860.38	849.32	9.33	88
C4	1037.02	1065.29	1051.90	9.97	512
C5	1320.50	1373.35	1351.54	16.64	2756
C6	**555.43**	571.44	562.53	5.51	5
C7	**909.68**	940.26	923.07	11.54	41
C8	**865.94**	890.08	877.59	8.99	142
C9	1169.69	1196.65	1182.97	8.58	980
C10	1409.61	1500.55	1453.92	29.72	4632
C11	**1042.11**	1047.45	1044.64	1.81	311
C12	831.83	855.40	843.93	8.26	86
C13	1545.68	1569.43	1553.75	8.15	494
C14	**866.36**	869.15	867.62	0.97	136
Avg.	981.79	1009.63	995.69	-	729

[a]: the solution equals to the best-known solution.

3.4 Comparison of IACS with other meta-heuristics

In Table 10, IACS is compared with two previous AS results and five other meta-heuristic approaches (2 SA, 2 TS and 1 GA) in terms of RPD. The numbers in bold indicate the best solution among eight algorithms. The average RPD of all benchmark problems obtained by IACS is the lowest among eight algorithms. IACS also yields the best solutions among all algorithms in 8 out of 14 problems.

Table 10. Comparison of RPD obtained by IACS with other meta-heuristics

No.	AS[1]	IAS[2]	SA[3]	SA[4]	TS[5]	TS[6]	GA[7]	IACS
C1	**0.00**[a]	**0.00**	0.65	**0.00**	**0.00**	**0.00**	0.04	**0.00**
C2	4.23	1.08	0.40	0.69	**0.06**	**0.00**	1.74	0.01
C3	6.45	0.75	0.37	0.47	0.40	**0.00**	1.76	0.61
C4	11.57	3.22	2.88	3.36	0.75	**0.11**	2.67	0.84
C5	14.09	4.03	6.55	5.31	2.42	**0.55**	6.76	2.25
C6	1.35	0.87	**0.00**	**0.00**	**0.00**	**0.00**	0.87	**0.00**
C7	4.23	0.72	**0.00**	1.13	0.39	6.15	0.49	**0.00**
C8	2.34	0.09	0.09	0.47	**0.00**	1.75	0.79	**0.00**
C9	3.39	2.88	**0.14**	2.96	1.31	-	2.62	0.61
C10	7.80	4.00	1.58	4.74	1.62	3.11	6.25	**0.99**
C11	2.91	2.22	12.85	**0.00**	3.01	**0.00**	1.74	**0.00**
C12	0.05	**0.00**	0.79	0.18	**0.00**	**0.00**	7.11	1.50
C13	3.20	1.22	0.31	1.74	2.12	5.02	1.37	**0.29**
C14	0.40	0.08	2.73	0.07	**0.00**	5.64	0.69	**0.00**
Avg.	4.43	1.51	2.09	1.36	0.86	1.60	2.49	**0.51**

[a]: the best among all algorithms
[1]: AS by Bullnheimer et al. (1998)
[2]: IAS by Bullnheimer et al. (1999a)
[3]: SA by Osman (1993)
[4]: SA by Van Breedam (1995)
[5]: TS by Gendreau et al. (1994)
[6]: TS by Xu and Kelly (1996)
[7]: GA by Baker and Ayechew (2003)

Table 11 summarizes NBKS (the number of solutions achieving best-known solutions), computational time in seconds, the average RPD, computer used and scaled time of each algorithm. Regarding computational time, Osman used a VAX 8600, Gendreau et al. used a 36 MHz Silicon Graphics (about 5.7 MFlop/s), Van Breedam used a Pentium 266 MHz PC (55 Mflop/s), Xu and Kelly used an unspecified DEC Alpha (26–43 MFlop/s, about 5 times faster than the Silicon Graphics according to Gendreau), Bullnheimer et al. used a Pentium 100 MHz PC (8 Mflop/s). The 1000MHz PC running the GA has an estimated power of 75 MFlop/s. The scaled time is computed as follows: If T_a is the computational time and P_a the computer power (Mflop/s) for one algorithm a, its scaled time is $(T_a/T_g) \times (P_a/P_g)$, with g standing for IACS. The IACS is 3.7 times faster than the TS heuristic from Xu and Kelly (1996), but 6 times slower than the IAS from Bullnheimer et al. (1998, 1999a). Overall, the results of table 11 show that IACS achieves comparable results since it produces the lowest RPD and the largest NBKS.

Table 11. Comparison of computational time

Method	NBKS	RPD	CPU(sec)	Computer Used	Scaled Time
AS[1]	1	4.43	1086	Pentium 100 MHz PC (8 Mflop/s)	0.16
IAS[2]	2	1.51	1106	Pentium 100 MHz PC (8 Mflop/s)	0.16
SA[3]	2	2.09	9081	VAX 8600 workstation	-
SA[4]	3	1.36	7710	80386 compatible computer 40 MHz	-
TS[5]	5	0.86	2808	Silicon Graphics workstation 36 MHz (5.7 Mflop/s)	0.29
TS[6]	6	1.60	7896	DEC Alpha workstation (26 Mflop/s)	3.75
GA[7]	0	2.49	1598	Pentium 266 MHz PC (55 Mflop/s)	1.61
IACS	6	0.51	708	Pentium 1000 MHz PC (75 Mflop/s)	1

[1]: AS by Bullnheimer et al. (1998)
[2]: IAS by Bullnheimer et al. (1999a)
[3]: SA by Osman (1993)
[4]: SA by Van Breedam (1995)
[5]: TS by Gendreau et al. (1994)
[6]: TS by Xu and Kelly (1996)
[7]: GA by Baker and Ayechew (2003)

4. CONCLUSION

In this research, we have proposed an improved ant colony system (IACS) that adopts a new state transition rule, a pheromone updating rule and two local search methods, for solving the vehicle routing problem. The main idea is to give extra emphasis to the global-best and iteration-best solutions. The computational study of fourteen benchmark problems shows that the IACS performs better than the two previous Ant System algorithms. Furthermore, the performance of IACS is competitive when compared with other meta-heuristic approaches, such as SA, GA and TS. Though the run time is not favor in IACS, our IACS still produces the largest number of best know solutions and yields the lowest average RPD among all metaheuristic approaches. Regarding the computation efficiency, we find that the IACS can find very good solutions in a short time. This strength can be combined with other metaheuristic approaches in the future work.

REFERENCE

1. Alfa, A. S., S. S. Heragu and M. Chen, "A 3-opt based simulated annealing algorithm for vehicle routing problems," *Computers & Industrial Engineering*, **21**, 635-639 (1991).
2. Baker, B. M. and M. A. Ayechew, "A genetic algorithm for the vehicle routing problem," *Computers & Operations Research*, **30**, 787-800 (2003).
3. Bullnheimer, B., R. F. Hartl and C. Strauss, "Applying the ant system to the vehicle routing problem," In S. Voss, S. Martello, I.H. Osman, and C. Roucairol, editors, Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization, Kluwer, Boston, MA, 109-120 (1998).
4. Bullnheimer, B., R. F. Hartl and C. Strauss, "An improved ant system for the vehicle routing problem," *Annals of Operations Research*, **89**, 319-328 (1999a).
5. Bullnheimer, B., R. F. Hartl and C. Strauss, "A New Rank Based Version of the Ant System - A Computational Study," *Central European Journal of Operations Research*, **7**, pp. 25-38 (1999b).
6. Christofides, N., A. Mingozzi and P. Toth, *The Vehicle Routing Problem*. In: Combinatorial Optimization, ed. N. Christofides, A. Mingozzi, P. Toth, & C. Sandi. Wiley, Chichester (1979).
7. Colorni A., M. Dorigo, F. Maffioli, V. Maniezzo, G. Righini and M. Trubian, "Heuristics from nature for hard combinatorial problems," *International Transactions in Operational Research,* **3(1)**, 1-21 (1996).
8. Colorni, A., M. Dorigo and V. Maniezzo, "Distributed optimization by ant colonies," In F. Varela and P. Bourgine, editors, Proceeding of the European Conference on Artificial Life, Elsevier, Amsterdam (1991).
9. Colorni, A., M. Dorigo, V. Maniezzo and M. Trubian, "Ant system for job-shop scheduling," *Belgian Journal of Operations Research, Statistics and Computer Science*, **34(1)**, 39-53 (1994).
10. Costa, D. and A. Hertz, "Ants can colour graphs," *Journal of the Operational Research Society,* **48**, 295-305 (1997).
11. Dorigo, M. and L. M. Gambardella, "Ant colonies for the traveling salesman problem," *BioSystems,* **43**, 73-81 (1997a).
12. Dorigo, M. and L. M. Gambardella, "Ant colony system: A cooperative learning approach for the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, **1**, 53-66 (1997b).
13. Dorigo, M., V. Maniezzo and A. Colorni, "Ant system: Optimization by a colony of cooperating agents," *IEEE Transactions on Systems*, Man and Cybernetics Part B, **26**, 29-41 (1996).
14. Gambardella, L. M. and M. Dorigo, "HAS-SOP: A hybrid ant system for the sequential ordering problem," *Tech. Rep. No. IDSIA 97-11,* IDSIA, Lugano, Switzerland (1997).
15. Gambardella, L. M., E. Taillard and G. Agazzi, "Ant colonies for vehicle routing problems," In D. Corne, M. Dorigo, and F. Glover, editors, New Ideas in Optimization. McGraw-Hill (1999a).
16. Gambardella, L. M., E. Taillard and G. Agazzi, "MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows," In D. Corne, M. Dorigo and F. Glover (Eds.), New ideas in optimization, Maidenhead, UK: McGraw-Hill (1999b).
17. Gambardella, L. M., E. Taillard and M. Dorigo, "Ant colonies for the quadratic assignment problem," *Journal of the Operational Research Society,* **50**, 167-176 (1999).
18. Garey, M. R. and D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP completeness, W.H. Freeman & Co., New York (1979).
19. Gendreau, M., A. Hertz and G. Laporte, "A tabu search heuristic for the vehicle routing problem," *Management Science*, **40**, 1276-1290 (1994).
20. Ghaziri, H. Supervision in the self-organizing feature map: application to the vehicle routing problem. In I. Osman, and J. Kelly, editors, *Meta-heuristics: Theory & Application*, Kluwer Academic Publishers, 651-660 (1996).
21. Golden, B. L., E. A. Wasil, J. P. Kelly and I-M. Chao, "Metaheuristic in vehicle routing," In T. G. Crainic and G. Laporte, editors. *Fleet Management and Logistics*, Kluwer, Boston, 33-56 (1998).
22. Hadjiconstantinou, E., N. Christofides and A. Mingozzi, "A new exact algorithm for the vehicle routing problem based on q-paths and k-shortest paths relaxations," In M. Gendreau and G. Laporte, editors, *Freight Transportation*, Volume 61 of Annals of Operations Research, 21-43 (1995).
23. Jaszkiewicz, A. and P. Kominek, "Genetic local search with distance preserving recombination operator for a vehicle routing problem," *European Journal of Opera-*

tional Research, **151**, 352-364 (2003).
24. Lin, S., "Computer solution of the traveling salesman problem," *Bell System Computer Journal*, **44**, 2245-2269 (1965).
25. Maniezzo, V., A. Colorni and M. Dorigo, "The ant system applied to the quadratic assignment problem," Tech. Rep. IRIDIA/94-28, Université Libre de Bruxelles, Belgium (1994).
26. Osman, I. H., "Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem," *Annals of Operations Research*, **41**, 421-451 (1993).
27. Robuste, F., C. F. Daganzo and R. Souleyrette, "Implementing vehicle routing models," *Transportation Research B*, **24**, 263-586 (1990).
28. Rochat, Y. and E. Taillard, "Probabilistic and intensification in local search for vehicle routing," *Journal of Heuristics*, **1**, 147-167 (1995).
29. Stützle, T. and M. Dorigo, "ACO algorithms for the quadratic assignment problem**,"** In D. Corne, M. Dorigo and F. Glover, editors, *New Ideas in Optimization,* McGraw-Hill, 33-50 (1999).
30. Su, C. T. and H. H. Chen, "Vehicle routing design of physical distribution center," *Journal of the Chinese Institute of Industrial Engineers*, **16(3)**, 405-417 (1999).
31. Taillard, E., "Parallel iterative search methods for vehicle routing problems," *Networks*, **23**, 661-673 (1993).
32. Talbi, E. G., O. Roux, C. Fonlupt and D. Robillard, "Parallel ant colonies for the quadratic assignment problem," *Future Generation Computer Systems*, **17(4)**, 441-449 (2001).
33. Van Breedam, A., "Improvement heuristics for the vehicle routing problem based on simulated annealing," *European Journal of Operational Research*, **86**, 480-490 (1995).
34. Xu, J. and J. P. Kelly, "A network flow-based tabu search heuristic for the vehicle routing problem," *Transportation Science*, **30**, 379-393 (1996).

ABOUT THE AUTHORS

Chia-Ho Chen is a Ph.D. student of Industrial Engineering and Management at Yuan Ze University, Taiwan. He received his M.S. degree in Department of Business and Operations Management from the Chang Jung Christian University. His primary research interests include vehicle routing problem, location problem and simulation.

Ching-Jung Ting is an assistant professor in the Industrial Engineering and Management Department at Yuan Ze University, Taiwan. He received his B.B.A. from National Chiao Tung University, Taiwan, M.S. in Civil Engineering from Northwestern University, and Ph.D. in Civil Engineering from University of Maryland, College Park. His primary research interests include supply chain management, logistics management, metaheuristics, and transportation system analysis.

(Received October 2004; revised March 2005; accepted July 2005)

應用改良式群蟻系統演算法於車輛途程問題之研究

陳家和
元智大學工業工程與管理學系
320 桃園縣中壢市遠東路 135 號
丁慶榮*
元智大學工業工程與管理學系

摘要

車輛途程問題(Vehicle Routing Problem; VRP)是近五十年來，一個非常著名的組合最佳化問題，過去已有許多學者針對 VRP 進行求解與探討。很多萬用啓發式演算法(Meta-heuristic)已被應用於求解 VRP，如模擬退火法(Simulated Annealing; SA)、基因遺傳演算法(Genetic Algorithms; GA)、禁忌搜尋法(Tabu Search; TS)與螞蟻演算法(Ant Algorithm)等。其中，螞蟻演算法爲 90 年代根據螞蟻族群搜尋食物的現象，所發展而成的萬用啓發式演算法。現今，已有越來越多的學者針對螞蟻演算法進行改良與修正，並應用於許多組合最佳化問題。因此，本研究提出一改良式的群蟻系統演算法(Ant Improved Colony System, IACS)，其採用新的途程建構準則、費洛蒙(Pheromone)更新方式與多種區域搜尋法(Local Search)，以改善傳統螞蟻演算法求解VRP 之品質。本研究求解 14 題 VRP 之標竿測試例題，並將求解結果與 SA、GA、TS 及其它已發表之螞蟻演算法進行比較。比較結果顯示，IACS 之求解品質優於其它已發表之螞蟻演算法，且不遜於其它比較之萬用啓發式演算法。

關鍵詞：車輛途程問題、萬用啓發式演算法、群蟻系統演算法
(*聯絡人: ietingcj@saturn.yzu.edu.tw)