Appl Intell (2011) 34: 64–73
DOI 10.1007/s10489-009-0179-6

Study on hybrid PS-ACO algorithm

Bing Shuang · Jiapin Chen · Zhenbo Li

Published online: 14 May 2009

Abstract Ant colony optimization (ACO) algorithm is a recent meta-heuristic method inspired by the behavior of real ant colonies. The algorithm uses parallel computation mechanism and performs strong robustness, but it faces the limitations of stagnation and premature convergence. In this paper, a hybrid PS-ACO algorithm, ACO algorithm modified by particle swarm optimization (PSO) algorithm, is presented. The pheromone updating rules of ACO are combined with the local and global search mechanisms of PSO. On one hand, the search space is expanded by the local exploration; on the other hand, the search process is directed by the global experience. The local and global search mechanisms are combined stochastically to balance the exploration and the exploitation, so that the search efficiency can be improved. The convergence analysis and parameters selection are given through simulations on traveling salesman problems (TSP). The results show that the hybrid PS-ACO algorithm has better convergence performance than genetic algorithm (GA), ACO and MMAS under the condition of limited evolution iterations.

Keywords Ant colony optimization · Particle swarm optimization · Hybrid PS-ACO · TSP

1 Introduction

Ant colony optimization (ACO) is a recently developed, evolution based approach which has been inspired by the behavior of real ant colonies, in particular, by their foraging behavior [1, 2]. ACO has been successfully applied to several NP-hard combinatorial optimization problems [3], such as traveling salesman problems (TSP) [3, 4], quadratic assignment [5], vehicle routing [6], sequential ordering [7] and scheduling [8].

The ACO algorithm uses pheromone as an indirect communication medium among the individuals of a colony of ants, and the procedure of converging to the global optimum is a dynamic positive feedback of pheromone. Once a better solution is found, exploitation of the solution will be made by increasing the associated pheromone trails, while the pheromones trails of other solutions will be evaporated. If one of the exiting solution-components has a much higher pheromone trails level than the others, an ant will prefer the solution-component over all alternatives due to the probabilistic selection and further reinforcement will be given to the solution-component in the pheromone trails update. Then, as the evolution goes on, some better solution-components will obtain more pheromone trails, and the difference of pheromone trails between different solution-components will be more significant. Therefore, the ant will construct the same solution that contains solution-components with higher pheromone trails levels over and over again and the exploration of the search space stops.

Although the ACO algorithm is useful for discovering near-optimal solutions for some optimization problems, the long time required to find such results and premature convergence of ACO make it infeasible for large problems. Therefore, several improvements have been presented. In the ant colony system (ACS) of Dorigo [9], an additional global updating rule is applied to components which belong to the best ant tour, which will give more chances for those components with more pheromone trails to be visited by following ants and accelerate the convergence. The MAX- MIN ant

B. Shuang (✉) · J. Chen · Z. Li
National Key Laboratory of Nano/Micro Fabrication Technology, Key laboratory for Thin Film and Microfabrication of Ministry of Education, Research Institute of Micro/Nano Science and Technology, Shanghai Jiao Tong University, Shanghai 200030, China
e-mail: chenjp@sjtu.edu.cn

system (MMAS) of Stutzle [3] allows only the best solutions to add pheromone trails during the pheromone trails update, as well as it uses a rather simple mechanism for limiting the strengths of the pheromone trails effectively to avoid premature convergence of the search.

All improved ACO algorithms have one important feature in common: they exploit the best solutions found during the search much more than the basic ACO. They use local search to improve the solutions constructed by the ants. The fact that additional exploitation of the best found solutions provides the key for an improved performance, is certainly related to the shape of the search space of many combinatorial optimization problems [3].

While exploitation of the best solutions can direct the search processes, it can also, however, restrict the search processes by making the search depend more on the best solutions and ignore the exploration of new search spaces. The key to achieve best performance of ACO algorithms is to combine the search space exploration with best solutions exploitation.

Particle swarm optimization (PSO) [10–13], inspired by social behavior simulation, was originally designed and developed by Eberhart and Kennedy [10]. In PSO, a swarm consists of individuals, called particles, which change their positions (*adaptable velocity*) over time. Each particle represents a potential solution to the problem, and searches around in a multi-dimensional search space. During its search, each particle adjusts its position according to its own experience and the experience of its neighboring particles, making use of the best position encountered by itself and its neighbors. The effect is that particles move towards the better solution areas while still having the ability to search a wide area around the better solution space.

Being illuminated by the idea of PSO, we employ the mechanism of PSO to improve the pheromone updating rules of ACO. Therefore, the primary contribution of this paper is combining the mechanisms of PSO and ACO to create the new hybrid PS-ACO algorithm which can perform better convergence. Through two acceleration parameters, the local search experience (i.e., the local best solutions) and the global search experience (i.e., the global best solutions) of the ant colony is integrated into the pheromone updating rules of ACO. Accordingly, the search space is expanded by local exploration around the local best tours and the best global-searched solution is reinforced to direct the search process, which are beneficial to improve the convergence performance of ACO.

The remainder of this paper is structured as follows. In Sect. 2, the hybrid PS-ACO algorithm is proposed. The convergence performance of the PS-ACO is compared with GA, ACO and MMAS in Sect. 3. Section 4 simulates the parameters selection. Finally, Sect. 5 concludes.

2 Hybrid PS-ACO algorithm

2.1 Basic of ACO algorithm

The ACO algorithm imposes a definition of the optimization problem into a graph, in which the ants move along every branch from one node to another and construct the paths representing the solutions. Let graph $G = (V, C)$ denotes the solution space, where $V = \{v_1, v_2, \ldots, v_m\}$ is the set of nodes in the graph G, and $C = \{(v_i, v_j) | v_i \in V, v_j \in V\}$ is the set of arcs among those nodes, (v_i, v_j) is the edge between v_i and v_j, $i \in \{1, 2, \ldots, m\}$, $j \in \{1, 2, \ldots, m\}$.

Initially, n ants are placed on m nodes randomly. Then, in each construction step, each ant moves to a node it has not yet visited based on a probabilistic decision. When it completes a tour, it lays a substance called pheromone trail on the edges. In ACO, the transferring probability of an ant $k (k \in \{1, 2, \ldots, n\})$ currently located at node i chooses node j as next node is given as (1), which indicates that the transferring probability is associated with the pheromone trails and locally available heuristic information [4]

$$p_{ij}^k(t) = \frac{[\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta}{\sum_{l \in N_i^k} [\tau_{il}(t)]^\alpha [\eta_{il}]^\beta} \quad \text{if } j \in N_i^k \tag{1}$$

where $\tau_{ij}(t)$ is the quantity of pheromone trails at edge (v_i, v_j) at time t. η_{ij} is the heuristic information and most of the ACO algorithms for TSP use $\eta_{ij} = 1/d_{ij}$, d_{ij} is the length of edge (v_i, v_j). α and β are two parameters which determine the relative importance of the pheromone trails and the heuristic information. N_i^k is the feasible selections of ant k, that is, the set of nodes which ant k has not visited yet. Equation (1) implies that ants prefer nodes which are closer and connected by arcs with higher pheromone trails.

Once the ant completes a tour of the graph, it deposits pheromone trails on the arcs it traveled to update the pheromone trails. In ACO algorithm, the pheromone updating rule is defined as follows:

$$\tau_{ij}(t + \Delta t) = (1 - \rho) \cdot \tau_{ij}(t) + \Delta\tau_{ij} \tag{2}$$

where $\rho (0 < \rho < 1)$ is the pheromone trails evaporation coefficient. $\Delta\tau_{ij}$ defines the pheromone trails increment of the edge (v_i, v_j) during the given period Δt.

In ACO algorithm, $\Delta\tau_{ij}$ is defined by the following equation:

$$\Delta\tau_{ij} = \sum_{k=1}^{n} \Delta\tau_{ij}^k \tag{3}$$

where $\Delta\tau_{ij}^k$ is the increment of pheromone trails laid on edge (v_i, v_j) by ant k between time t and $t + \Delta t$. It is defined by:

$$\Delta\tau_{ij}^k = \begin{cases} \frac{Q}{f_k} & (v_i, v_j) \in \pi_k, \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where Q is a constant and f_k is the fitness value of the sequence π_k toured by ant k in Δt. Equations (3) and (4) indicate that the pheromone increments only relate to the current search of the ant colony, which means that history experience is ignored and the valuable solutions have not been reinforced enough.

2.2 PSO algorithm

The PSO algorithm is similar to evolutionary computation (EC) techniques in that, a population of particles (i.e., potential solutions) to the problem under consideration is used to probe the search space [13]. However, in PSO, each particle of the population has an adaptable velocity, according to which it moves in the search space. Moreover, each particle remembers the best position of the search space it has ever visited. Thus, its movement is an aggregated acceleration towards its best previously visited position and towards the best individual of neighborhood.

For a D-dimensional search space, the i-th particle of the swarm can be represented by a vector $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})^T$. The velocity of this particle can be defined as another D-dimensional vector $V_i = (v_{i1}, v_{i2}, \ldots, v_{iD})^T$. The best previously visited position of the i-th particle is denoted as $P_i = (p_{i1}, p_{i2}, \ldots, p_{iD})^T$. Defining g as the index of the best particle in the swarm (i.e., the g-th particle is the best), then the swarm is manipulated according to the following two equations [10, 11]:

$$v_{id}^{n+1} = w v_{id}^n + c_1 r_1^n (p_{id}^n - x_{id}^n) + c_2 r_2^n (p_{gd}^n - x_{id}^n), \quad (5)$$

$$x_{id}^{n+1} = x_{id}^n + v_{id}^{n+1} \quad (6)$$

where, n is the iteration number, $d = 1, 2, \ldots, D$, $i = 1, 2, \ldots, N$, and N is the number of particles. w is the inertia weight. c_1 and c_2 are positive constants, called acceleration constant, which regulate the trade-off between the local experience and global experience of the swarm. r_1 and r_2 are random numbers uniformly distributed in [0, 1], which are used to maintain the diversity of the population.

2.3 Hybrid PS-ACO algorithm

In the PS-ACO algorithm, the transferring probability and the pheromone updating rules have the same expressions as (1) and (2), but the pheromone trails increment $\Delta\tau_{ij}$ of PS-ACO is improved with the idea of PSO algorithm and it is defined by:

$$\Delta\tau_{ij} = \sum_{k=1}^{n} \Delta\tau_{ij}^k + c_1 \cdot r_1 \cdot \sum_{k=1}^{n} \Delta\tau_{ij}^{local_k} + c_2 \cdot r_2 \cdot \Delta\tau_{ij}^{global} \quad (7)$$

where $\Delta\tau_{ij}^k$ has the same definition as (4), $\Delta\tau_{ij}^{local_k}$ is the local update value of ant k at edge (v_i, v_j), and it is defined by:

$$\Delta\tau_{ij}^{local_k} = \frac{Q}{f_{best\_so\_far}^k} \quad (8)$$

where $f_{best\_so\_far}^k$ is the fitness value of the best solution toured by ant k from the beginning.

$\Delta\tau_{ij}^{global}$ is the global update value of edge (v_i, v_j) which is given by:

$$\Delta\tau_{ij}^{global} = \frac{Q}{f_{best\_so\_far}} \quad (9)$$

where $f_{best\_so\_far}$ is the fitness value of the best solution toured by all ants from the beginning.

The parameters c_1 and c_2 are acceleration coefficients determining the influences of $\sum_{k=1}^{n} \Delta\tau_{ij}^{local_k}$ and $\Delta\tau_{ij}^{global}$, proper values of the two parameters may result in faster convergence and alleviation of local minima. r_1 and r_2 are random numbers uniformly distributed in [0,1] which can weaken the influence of the local best values and global best values in a random way and maintain the diversity of the search, it can also prevent the relative differences between the pheromone trails from becoming extremely large in the evolution of the algorithm. Therefore, r_1 and r_2 can balance the effects of local and global update, they are helpful for avoiding over strengthen the influence of any of the two update factors and reducing the probability of prematurity in search process.

Equation (7) shows the reinforcements of the best solutions encountered by ants and the experience of the whole ant colony. The local update rule emphasizes the local best solutions of ants. Local best solutions in the TSP are concentrated around a small region of the whole search space [3], so the search space will be distributed or expanded with additional pheromone trails deposit on different local best solutions of ants. The global update rule emphasizes the global best solutions of the search procedure, which exploits and guides the search of the whole colony with reinforcing the globally found best solutions. Accordingly, the new hybrid PS-ACO algorithm makes the search move towards the better solution spaces, and maintain the ability to search wider areas around the better solutions synchronously, that is, the PS-ACO may converge more quickly and precisely.

3 Convergence analysis of PS-ACO

To test the converging performance of PS-ACO, we compare it with another three algorithms, genetic algorithm (GA) [14], ACO and MMAS, through the standard travel salesman problems (TSP). The four algorithms are fully implemented

in MATLAB using Pentium 4 3.0 GHz PC with 1 GB memory, and all the instances are taken from the TSPLIB [15] and the optimal solutions are known in advance.

For the three ant-based algorithms, ACO, MMAS and PS-ACO, the colony parameters are same: the ant number equals to the city number, and the maximal iteration number is 2000. The ACO algorithm is implemented as Sect. 2.1 of this paper, and the other parameters are set as $\alpha = 1.0$, $\beta = 5.0$, $Q = 100$, $\rho = 0.01$. For MMAS, the algorithm is implemented as Ref. [3] and the parameters are set as $\alpha = 1.0$, $\beta = 2.0$, $\rho = 0.02$, $p_{best} = 0.05$ and in every 10th iteration the global update is used to reinforce the pheromone trails as given in [3]. For PS-ACO, unless explicitly indicated, the following parameters are set as default: $\alpha = 1.0$, $\beta = 2.0$, $Q = 100$, $\rho = 0.015$, $c_1 = 2.0$, $c_2 = 2.0$ and r_1, r_2 are set randomly within each iteration. The GA is implemented according to [14] and path representation is used to represent the sort of cities, position based crossover and exchange mutation strategies are used as genetic operators. The generation number is set to 5000 and the population number is equal to the city number. The crossover probability, mutation probability and selection probability are 0.3, 0.1 and 0.7, respectively. Several standard TSP instances are selected to evaluate the algorithms, and each algorithm runs 20 times to get the average performance values.

The mean value, the best value, and the percentage error of the mean value to the optimal value of the four algorithms are listed in Table 1. The mean number of converging iterations of the three ant-based algorithms, ACO, MMAS and PS-ACO, at which they stagnated, are also presented in the table. We can find the mean values of PS-ACO are closest to the optimal values in the 20 runs of the four algorithms for different TSP instances. The errors of ACO are obviously larger than that of PS-ACO, up to about forty times for the att48.TSP, and GA has the same result as ACO. The errors of PS-ACO are also better than those of MMAS within the simulations. As to the best values, the PS-ACO can get the optimal values for the first two TSPs and obtain the most nearest values to the optimums for the others.

As to the mean number of converging iterations, the values of ACO are obviously smaller than the other two ant-based algorithms and the values of MMAS are the largest which mean that the ACO will stagnate earlier and the MMAS converges slower than the PS-ACO in the given evolution iterations. For the generation number of GA is different with the maximal iteration number of the three ant-based algorithms, the mean iterations of GA are ignored in the table.

Figure 1 illustrates the evolution curves of mean values and best values for certain execution of the PS-ACO and ACO, in which the mean values are average values of all ants in the evolution. Plot (a) gives the evolution curves of PS-ACO algorithm and Plot (b) shows the curves of ACO. Compares plot (a) with plot (b), the best values of the PS-ACO can converge with evolution goes on and reach the best at about the 1380^{th} iteration, the mean values of PS-ACO keep on turning down quickly until the maximal evolution iteration is reached. While the best values of the ACO will

Table 1 Comparisons of GA, ACO, MMAS and PS-ACO

Instance name (optimal value)	Algorithm	Mean value	Error (%)	Best Value	Mean iteration
att48.TSP	GA	35172.40	4.92	34099	–
(33522)	ACO	35066.25	4.61	34123	646.90
	MMAS	33739.76	0.65	33524	1375.35
	PS-ACO	33560.70	0.12	**33522**	1187.65
eil51.TSP	GA	442.20	3.86	433	–
(426)	ACO	441.45	3.62	436	631.90
	MMAS	433.15	1.67	430	1537.05
	PS-ACO	427.40	0.32	**426**	1157.30
pr76.TSP	GA	114678.13	6.03	112380	–
(108159)	ACO	117390.55	8.54	114083	991.10
	MMAS	113055.50	4.53	112372	1873.24
	PS-ACO	110162.10	1.85	109744	1563.70
rd100.TSP	GA	8251.76	4.32	8138	–
(7910)	ACO	8410.40	6.33	8255	1199.40
	MMAS	8183.2	3.45	8043	1904.52
	PS-ACO	8017.40	1.36	7967	1408.40

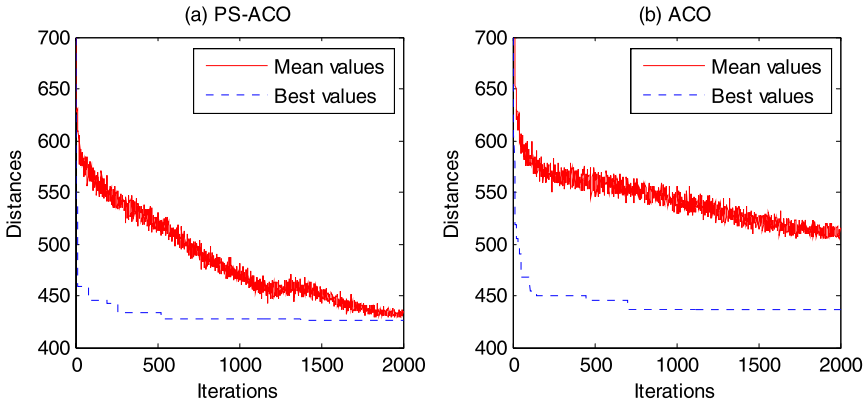

Fig. 1 Evolution curves of PS-ACO and ACO for eil51.TSP

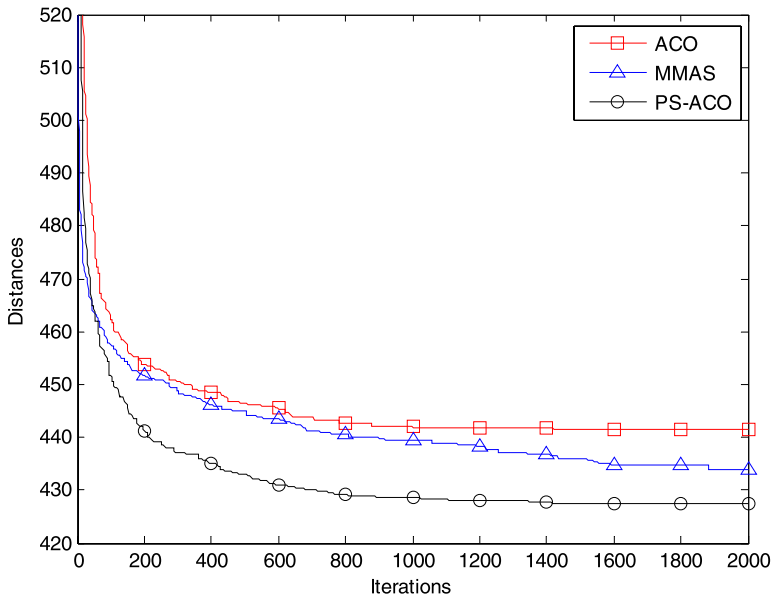

Fig. 2 Comparison of mean evolution curves of ACO, MMAS and PS-ACO for eil51.TSP

stagnate at about the 700th iteration and the mean values decrease fast at the beginning and become much slower than the mean values of the PS-ACO with the algorithm running. We can see that the PS-ACO can keep on optimizing and finding better solutions during the evolution, but it is difficult to improve the search for ACO, because search stagnation may occur in ACO and converge prematurely.

The mean evolution curves of best values for ACO, MMAS and PS-ACO are compared in Fig. 2, each curve is an average of evolutionary values for 20 independent executions of the associated algorithm. As indicated in Fig. 2, the mean best value of PS-ACO is better than the values of MMAS and ACO. The ACO converges slowly and stagnates early; the MMAS converges faster than the PS-ACO at the beginning and then it is exceeded by the PS-ACO. The PS-ACO converges quickly and reaches the best at about the 1400th iteration, while the MMAS will not stop converging until reach the given maximal evolution iteration.

Analyzing the mechanisms of the PS-ACO algorithm, we can find that the specific reinforcement of local update of the PS-ACO can use the local experience of ants and drive the algorithm to search in a much more parallelizing way, which will accelerate the search process to explore wider search space and increase the probability of finding the optimal solutions. By exploiting the global best solutions, the global reinforcement will increase the differences of pheromone trails on components of solutions and make the algorithm converge more quickly. The local exploration expands the search space and makes it easy and fast to find the optimal solutions, while the global exploitation reinforces the best solutions and accelerates the search. The stochastic combination of local and global update will balance the influence of search space exploration and optimum exploitation on the search.

In MMAS, there are two kind of ants can update pheromone trails, one is called iteration-best ant, which found the best solution in the current iteration, the other is called global-best ant, which found the best solution from the beginning of the algorithm, and can update pheromone trails with an increasing frequency during the search. Conse-

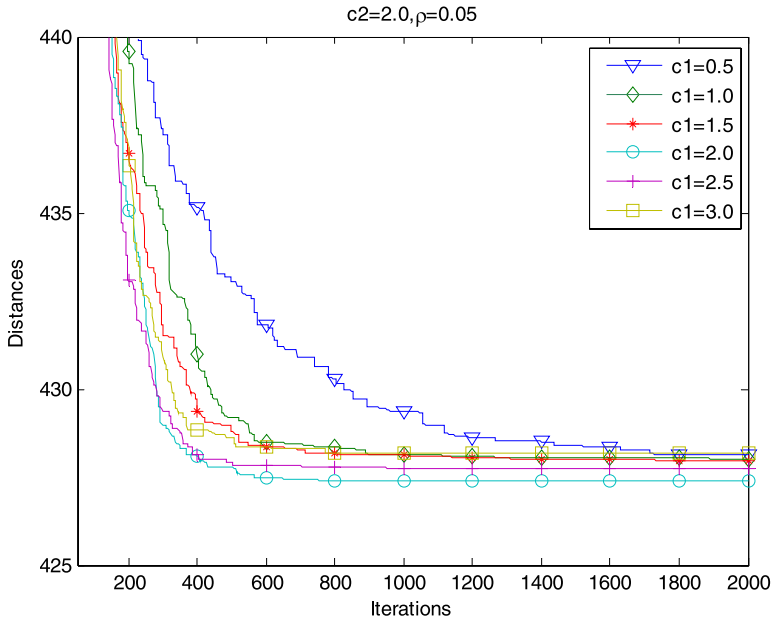

Fig. 3 Evolution curves of PS-ACO for different values of c_1

quently, the pheromone update is limited in the solution-components of the iteration-best ant after each iteration, only a few ants are affected by the local update for selecting these solution-components in the following search, most will tour the solutions randomly without any guide of previous experience, which will expand the search space widely and benefit for finding better solutions, but it is not good for improving the search speed. Although the global-best ant updates the pheromone trails dynamically with decreasing iteration intervals, the global reinforcement is not strong enough and which is more helpful for keeping the wide search space than improving the search efficiency. Therefore, MMAS can search wide solution spaces and find the best solutions with large evolution iterations, and that is the reason why the iteration number of MMAS is set as 2500 times of city number in [3]. Once the maximal iteration number is limited in our research, the MMAS is inferior to PS-ACO in convergence performance. This can explain why the mean converging iterations of MMAS are larger than PS-ACO in Table 1 and the mean evolution curve of MMAS keeps on converging through the evolution in Fig. 2. Note that the MMAS may work better, in finding the global optimum if larger evolution iteration is allowed, while the PS-ACO performs better when the maximal evolution iteration is limited.

Conclusions can be drawn from the simulations and analysis that the PS-ACO algorithm performs better than the GA, ACO and MMAS algorithm in converging efficiency.

4 Parameters discussion

The influences of the key parameters, c_1, c_2, r_1, r_2 and ρ, on the PS-ACO are discussed in solving the eil51.TSP instance. All the experiments are performed with the assumption that in each experiment only one single factor is varied and, hence, performance differences can only be attributed to the variation of this single factor. The curves given in this section are averages of 20 independent executions of the algorithm with different values of parameters.

Figure 3 depicts the evolution curves of PS-ACO with c_1 varying from 0.5 to 3.0 and other parameters set to default. From Fig. 3, it can be observed that with the increment of c_1, the curves incline more sharply which means the algorithm converges faster, but the slopes of curves will decrease after $c_1 > 2.0$. Figure 4 shows the evolution curves of PS-ACO with c_2 varying in the same way as c_1. From Fig. 4, we can find that the slopes of curves are almost the same but the converging values are different, which mean that the parameter c_2 has little influence on convergence speed of the PS-ACO algorithm, but it affects the final converging values.

Table 2 gives the mean best values and mean iterations when c_1 and c_2 vary from 0.5 to 3.0 respectively. With c_1 increasing, the mean best values are almost the same while the mean converging iterations decrease with c_1 changing, which means that the mean best values have little relationship with c_1, but the PS-ACO algorithm converges faster with the increasing of c_1. When it comes to c_2, the mean best values decrease at first and increase after $c_2 > 2.0$, but the mean number of converging iterations are almost the same.

Therefore the parameter c_1 of the PS-ACO algorithm will mainly affect the converging speed; a suitable c_1 can speed

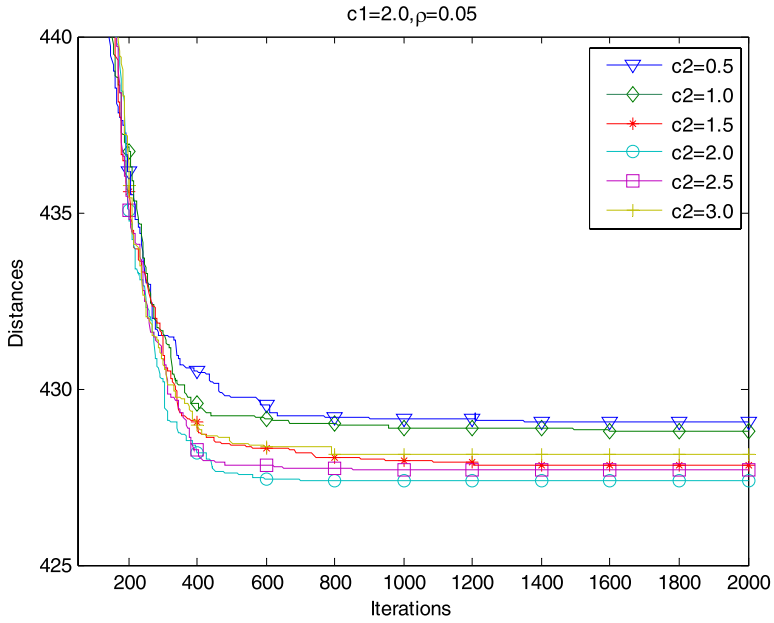

Fig. 4 Evolution curves of PS-ACO for different values of c_2

Table 2 Mean values and converging iterations for different values of c_1 and c_2

	Parameter value	0.5	1.0	1.5	2.0	2.5	3.0
c_1	Mean best values	428.15	428.00	427.95	427.40	427.75	428.20
	Mean iterations	1150.50	849.10	731.20	606.80	564.65	555.55
c_2	Mean best values	429.05	428.80	427.85	427.40	427.70	428.15
	Mean iterations	701.65	665.30	657.55	645.65	595.55	704.30

up the optimizing procedure. The parameter c_1 mainly controls the impact of the previous experience on the local ant, a large value of c_1 will facilitate exploration around the local best solutions, and the search space can be expanded from the view of the whole colony, which will increase the probabilities of finding the optimal solution and accelerate the search procedure. However, if the value of c_1 is too large, the local search procedure will be over emphasized and local experience will be reinforced excessively while the global experience will be ignored, so that the search will be limited in the local solution space and stagnate in local near-optimal values ultimately.

The parameter c_2 affects the converging values rather than the convergence performance. The parameter c_2 is the acceleration coefficient for global exploitation. With global exploitation, the pheromone trails of globally found best solutions are reinforced and ants in the following evolutions will have more probabilities to select the components of the best solutions, then the algorithm will search around the global best solutions. With the positive feedback of the global exploitation, the algorithm can push the search converging to the best solutions and keep on optimizing the solutions. However, if it is too large, the local exploration will be weakened relatively, which will lead to limited search spaces around the global best solution, then it will be difficult to improve the search and the search will converge prematurely. Hence the influences of c_1 and c_2 should be considered synthetically and the most suitable values of c_1 and c_2 are $c_1 = 2.0, c_2 = 2.0$ according to the simulations and analysis.

Figure 5 illustrates the evolution curves of PS-ACO to compare the influences of r_1 and r_2. The curve marked with squares is the evolution curve of PS-ACO when $r_1 = 1$, and r_2 is random within [0,1]. The curve marked with triangles corresponds to the PS-ACO when $r_2 = 1$ and r_1 is random within [0,1]. The curve marked with circles represents the normal PS-ACO. From Fig. 5, the evolution curve of $r_2 = 1$ decreases slower than the curve of $r_1 = 1$ before the 800^{th} iteration, and then the former curve converges faster and reaches a better value than the latter one. The normal evolution curve of PS-ACO is in the middle of the other two curves at the beginning, and then converges faster and better than the others.

The parameter r_1 is used to weaken the effect of local experience. $r_1 = 1$ means strengthening the local experience and relatively weakening the global exploitation, which will

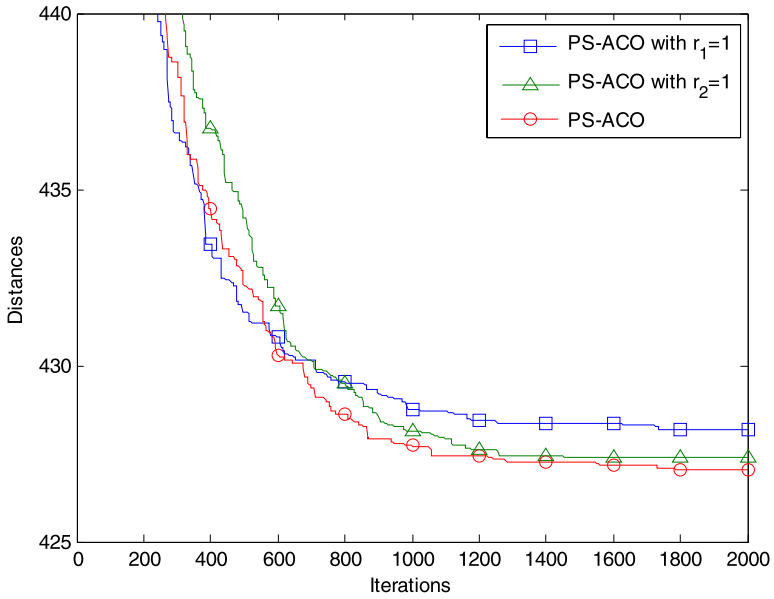

Fig. 5 Evolution curves of PS-ACO for using of r_1 and r_2

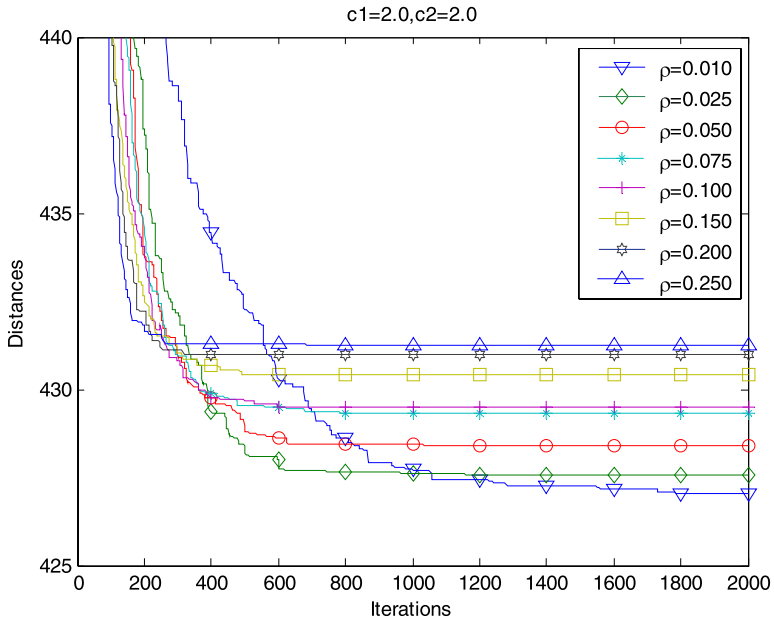

Fig. 6 Evolution curves of PS-ACO for different values of ρ

make the search focus on local exploration, so that the search will be accelerated while the algorithm may stagnate at the local near-optimal values. r_2 is used to weaken the influence of global best values in a random way. $r_2 = 1$ will reinforce the global experience and emphasize the global exploitation, so the search can be improved with the guide of global experience. However, the convergence speed is lowered at first for relative weakening of the local exploration. Then the search will be improved gradually with the evolution going on for reinforced global best solutions. On the other hand, randomizing the values of r_1 and r_2 can increase the diversity of the search and balance the effects of local exploration and global exploitation, which will consequently improve the convergence speed and converging values in PS-ACO. In another sense, $r_1 = 1$ or $r_2 = 1$ means relatively increasing the value of c_1 or c_2. A high value of c_1 can increase the convergence speed while c_2 with large value can improve the converging value, therefore, we can obtain the same conclusion from the point of c_1 and c_2.

Figure 6 depicts the evolution curves of PS-ACO with ρ varying from 0.010 to 0.250, the curves for different values of ρ are averages of 20 independent executions. It can be observed from Fig. 6 that better converging speed is obtained when larger value of ρ is used. However, with the increasing of ρ, the best values of different curves will increase and the search may stop within less iterations.

This is due to the fact that larger pheromone evaporation ratios on arcs will accelerate the pheromone evaporation and, hence, the search concentrates earlier around the best tours seen so far. If ρ is large, it is easy to reach marked relative differences between the pheromone trails on arcs contained in high quality tours and those which are not part of the best tours in few iterations, so the algorithm may stagnated and prematurely converging. Otherwise, for a lower ρ, the pheromone trails on arcs which are not belongs to the high quality tours will not decrease faster and the algorithm is able to explore wider search space, but longer evolution iterations are needed. Therefore, if larger total evolution iteration is used, a lower ρ can be selected for obtaining better converging value; otherwise, a higher ρ will be helpful for better convergence speed.

5 Conclusions

A new hybrid PS-ACO algorithm is presented in this paper which integrates the idea of PSO algorithm and ACO algorithm. The search mechanism of PSO algorithm is introduced into ACO algorithm, with combining the local exploration and the global exploitation into the pheromone update rules of ACO. Respectively, the local exploration is used to expand the search space of ACO, while the global experience is exploited to direct the search process and push the search converging to the global best values, so that the ant colony can keep on converging and exploring new search spaces synchronously.

The convergence performance of the PS-ACO algorithm is studied by comparison of GA, ACO, MMAS and PS-ACO algorithms in solving TSP problems. The specific reinforcement of local exploration of the PS-ACO can make use of local experience of ants and make the algorithm search in much more parallelized ways. The mechanism can expand the search space and make the algorithm find the better solutions more easily. The global reinforcement will exploit the best solutions and make the algorithm converge more quickly. With stochastic combination of the two effects, PS-ACO performs better than GA, ACO and MMAS in convergence.

Through the related simulations, we analyze the influences of the parameters and make recommendation on parameter selections. The simulation results manifest that the parameter c_1 mainly affects the convergence speed, a suitable c_1 can speedup the optimizing procedure. The parameter c_2 will influence the optimal values rather than the convergence performance. The random parameters r_1 and r_2 are used to combine the effects of local and global updates in a random way, balance the local exploration and global exploitation and keep the diversities of search. Additionally, a larger ρ will accelerate the search but the algorithm will converge prematurely with oversized value of ρ. Therefore, the influences of parameters should be considered synthetically and the most suitable parameters of the PS-ACO algorithm are $c_1 = 2.0$, $c_2 = 2.0$. The value of ρ should be selected with the maximal evolution iteration being considered. A larger maximal evolution iteration should correspond to a smaller value of ρ, and vice versa.

It should be pointed out that the PS-ACO algorithm performs well in solving TSP instances when the maximal evolution iteration of the algorithm is set to 2000 in the simulations. However, the value of maximal evolution iteration should not be constant. Here, 2000 is not appropriate for all instances; it may be too large for small-size problems or too small for large-scale ones. Therefore, we should pay attention to the relationship between the maximal evolution iteration and the problem scales. Furthermore,it is worth using the PS-ACO algorithm to solve other combination optimization problems, such as Quadratic Assignment Problem (QAP), scheduling, etc. All of these efforts will be carried out in the future.

Acknowledgements The authors gratefully acknowledge the grant of National Natural Science Foundation of China (No. 60475037 and No. 10477013), and Hi-Tech Research and Development Program of P. R. of China (No. 2007AA04Z340).

References

1. Maniezzo V, Carbonaro A (2001) Ant colony optimization: an overview, essays and surveys in metaheuristics. Kluwer, Dordrecht, 21–44
2. Dorigo M, Gianni DC, Gambardella LM (1999) Ant algorithms for discrete optimization. Artif Life 5:137–172
3. Stutzle T, Hoos HH (2000) MAX-MIN ant system. Future Gener Comput Syst 16(8):889–914
4. Dorigo M, Maniezzo V, Colorni A (1996) The ant system: Optimization by a colony of cooperating agents. IEEE Trans Syst Man Cybern B 26(2):29–41
5. Maniezzo V, Colorni A (1999) The Ant System applied to the quadratic assignment problem. IEEE Trans Data Knowl Eng 11(5):769–778
6. Bullnheimer B, Hartl RF, Strauss C (1999) An improved ant system algorithm for the vehicle routing problem. Ann Oper Res 89:319–328
7. Gambardella LM, Dorigo M (2000) Ant Colony System hybridized with a new local search for the sequential ordering problem. INFORMS J Comput 12(3):237–255
8. Zwann S, Marques C (1999) Ant colony optimization for Job Shop scheduling. In: Proceedings of the third workshop on genetic algorithms and artificial life (GAAL 99)
9. Dorigo M, Gambardella LM (1997) Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans Evol Comput 1(1):53–66
10. Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: Proceedings of IEEE conference neural networks, vol. IV, Piscataway, NJ, pp 1942–1948
11. Shi YH, Eberhart RC (1998) A modified particle swarm optimizer. In: Proceedings of 1998 IEEE international conference on evolutionary computation, Anchorage, AK, pp 69-73

12. Russell C, Eberhart, Shi YH (1998) In: Comparison between genetic algorithms and particle swarm optimization. Lecture notes in computer science, vol 1447. Springer, Berlin, pp 611–616
13. Parsopoulos KE, Vrahatis MN (2002) Recent approaches to global optimization problems through particle swarm optimization. Nat Comput 1:235–306
14. Larrañaga P, Kuijpers CMH, Murga RH et al (1999) Genetic algorithms for the travelling salesman problem: a review of representations and operators. Artif Intel Rev 13(2):129–170
15. http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/tsp/

Bing Shuang was born in Hubei, P. R. China, on September 17, 1979. He receives the MS degree in control theory and control engineering from East China University of Science and Technology in 2004. He is currently a PhD candidate in Research Institute of Micro/Nano Science and Technology, Shanghai Jiao Tong University, P. R. China. His research interests include microrobot communication, microrobot cooperation and swarm intelligence.

Jiapin Chen was born in Shanghai, P. R. China, on June 21, 1960. He received the BS degree in from Tong Ji University, China, in 1982, the MS degree in automatic control and the PhD degree in control theory and control engineering both from Shanghai Jiao Tong University, China, in 1986 and 2002, respectively. He joined the Research Institute of Micro/Nano Science and Technology, Shanghai Jiao Tong University, P. R. China, since 1995, where he worked until present. He has been a professor since 2003. His research interests are in MEMS, Intelligent Control and Microrobot.

Zhenbo Li was born in Jilin, P. R. China, on April 5, 1974. He received the BS degree in Mechanical Engineering and the MS degree in mechanical engineering both from Harbin Institute of Technology, China, in 1995 and 1997, respectively, and the PhD degree in control theory and control engineering from Shanghai Jiao Tong University, P. R. China, in 2000. He has been an associated professor in the Research Institute of Micro/Nano Science and Technology, Shanghai Jiao Tong University, China, since 2002. His current research interests are in the areas of Micro/Nanorobotics, Micro-actuator Technology and Control.