A New Pheromone Design in ACS for Solving JSP

Xiao-Lan Zhuo, Jun ZHANG, *MIEEE* and Wei-neng Cheng

Abstract—**Job Shop Scheduling Problem (JSP) is one of the most difficult NP-hard combinatorial optimization problems due to the "Combination Explosion" effect. This paper presents the implementation of Ant Colony System on JSP by proposing a novel combination of path-construction and pheromone-representation. Based on the simple traditional path-construction, a kind of more effective pheromone is employed to improve the optimization performance. Numerical experiment is executed on several benchmark JSP cases, and yields favorable results compared with results obtained by traditional implementation of ACS for JSP.**

Keywords: Ant colony system, job-shop scheduling problem

I. Introduction

JOB-SHOP Scheduling Problem (JSP) is one of the hardest COPs and has been demonstrated to be NP-hard by Garey, Johnson, and Sethi (1976) [1]. Solving a relatively small instance of 10×10 (Fisher and Thompson (1963) [2]) took researchers as long as about three decades. Even now researchers can only optimize JSPs up to a scale of 20×20 [3]. Since the JSP is vital in production and manufacturing industries, numbers of researchers have been devoted to finding an effective optimization algorithm for this problem[2], [4]-[7].

The traditional algorithm for JSP is bench-and-bound (B&B) method [8]. This method takes rather considerable computation time even when dealing with small JSP instance. During the past decades, researchers have been inclined to employing probabilistic, heuristic and approximate algorithms to solve NP-hard problems like JSP that cannot be exactly tackled in reasonable time. Algorithms applied to JSP include Genetic Algorithm (GA) [9]-[18], Simulated Annealing (SA) [10], [19]-[22], Tabu Search (TS) [10], [23]-[27], Neural Network (NN) [28], Particle Swarm Optimization (PSO) [29], and Ant System (AS) [30]-[32]. When GA is applied in JSP, modification is always needed for genotype representation and genetic operators [16]-[18], [33], since JSP is a discrete combinatorial optimization problem with constraints. Sometimes problem-specific information or supplementation of algorithms is also needed for interpreting and repairing genomes [3], [34]-[38]. SA is indicated to be no longer convergent due to the asymmetric standard neighborhood in the solution space of JSP [38], and the lack of memory may lead to oscillation in local optimum surrounding [3]. TS can produce excellent results for JSP, yet suffers from the dependent on the initial solution [3]. Moreover, convergence towards a global optimum is not promised in TS [3]. In [28], a neural network specially designed for JSP is also time-consuming, while AS in [30]-[32] cannot provide satisfactory JSP solutions. The newly proposed optimization algorithm PSO is applied for JSP in [29]. Due to the continuous nature of PSO, all the particle representation, movement and velocity should be specifically modified to suit PSO for the JSP. Unsuitable modification would significantly affect the optimization performance. Therefore, an algorithm naturally adapting to JSP with assurable and relatively fast convergent should be preferable, based on which efforts on improving the precision can be made.

This paper applied an ACO technique - Ant Colony System (ACS) [53] to solve JSP. ACS is a distributed algorithm simulating the foraging behavior of the ant colony. Ants are observed to be good at finding the shortest path between their nest and the food. They choose path based on memory and mutual-information in form of pheromone that they left along the way they walk. Properly designed pheromone can provide accurate and useful information for ants. The ACS algorithm deriving the ants' intelligence has been demonstrated with assurable and relatively quick convergence. Moreover, an ACS algorithm is inborn discrete which can simply adapt to COPs. The favorable characters of ACS make it an attractive alternative for JSP.

Designing an effective path-construction method and a proper pheromone representation concerning the specific problem can be of great effect to the performance. Since the path in JSP represents the scheduling order of operations, traditionally, the pheromone is accordingly designed to be the desirability of the relative or absolute position of an operation in the scheduling sequence [32]. However, some portions of the kind of pheromone are found redundant in instance study. On one hand, we don't take charge of all the relative positions among operations due to the problem constraints; on the other hand, some operations on different machines are irrelative, which would make no difference to the total processing time when switching their scheduling order. Our actual work is to schedule the operations on each machine, and the actual starting time of an operation just directly depends on its ancestor and its position on the requiring machine. Thus, considering the schedule in a view that separating the machines and relating the pheromone to the position of the operation on the machine can provide more effective guidance for ants by saving their labors on the connection of irrelative operations. In this order, some researchers divide

This work was supported by the National Natural Science Foundation of China No. 60573066 and Guangdong Natural Science Foundation No. 5003346

Xiao-lan Zhuo, Jun ZHANG and Wei-neng Chen are with Department of Computer Science, SUN Yat-sen University,Guangzhou, China.

Jun ZHANG is corresponding author,e-mail: junzhang@ieee.org.

the scheduling into sub-scheduling on different machines and construct sub-order sequence separately. However, additional efforts are needed for excluding infeasible solutions that obtained easily when being constructed in this way, which would lower the efficiency badly.

This paper presents a new combination of path-construction and pheromone-representation. The ants construct paths in the traditional way (without separating the machines), but their choices of operations depend on the pheromone that relates to the position of an operation on its requiring machine (in a view that separating the machines). This paper features on the combination of the simple construction with a more effective representation of pheromone.

Section II will present the description of the concerning JSP. Section III will describe the ACS flow and its implementation facing JSP with a novel pheromone design. Optimization results and further discussion are provided in Section IV. Finally, Section V gives the conclusion and intending expectation on ACS for JSP.

II. JOB-SHOP SCHEDULING PROBLEM

The Job-shop Scheduling Problem can be described as follows: Given an $n\times m$ static JSP, in which n jobs must be processed on m machines. Each job includes a set of m operations, each of which should be processed on one specified machine. Operations of a job should be processed in pre-fixed order. Issues about the problem constraints are as follow:

1) One job should use each machine once and only once, i.e., one machine can process exactly one operation of each job;
2) No machine can process more than one operation simultaneously;
3) No job can have more than one operation processed simultaneously, which means each operation cannot be processed until its anterior is finished;
4) Operation processing is uninterrupted;
5) There are no precedence between operations belonging to different jobs;
6) Processing time of each operation is pre-defined.

The problem is to schedule all the operations of jobs on machines without violating the above constraints, and the ultimate goal is to minimize the complete processing time C_{max}.

We define all the operations as a set $O = \{o_{ij} \mid i\in[1, n], j \in[1, m]\}$, in which o_{ij} indicates an operation of job i that have to be processed on machine j and taking an processing time p_{ij} ($p_{ij} >0$). The sequential relationship between operations o_{ip} and o_{iq} of job i can be defined as $o_{ip}\rightarrow o_{iq}$. No operation o_{ik} ($k \neq p$ and $k\neq q$) exists that $o_{ip}\rightarrow o_{ik}$ and $o_{ik}\rightarrow o_{iq}$ when $o_{ip}\rightarrow o_{iq}$. This means in a relationship $o_{ip}\rightarrow o_{iq}$, o_{iq} is the immediate successor of o_{ip}.

JSP can be translated into acyclic graph $G = \{N, E\}$ with a nodes set N and a directed edges set E. The nodes set N includes nodes corresponding to all the operations and two additional dummy nodes, that is, $N = N_0 \cup \{n_0\} \cup \{n_{n+1}\}$. Each node n_{ij} in node set N_0 corresponds to the operation o_{ij}, while n_0 and n_{n+1} identify the start and the end of the whole processing respectively. Thus the number of nodes in N is $n*m+2$. Edge (n_{ip}, n_{iq}) identifies the sequential relationship $o_{ip}\rightarrow o_{iq}$. Note that there are edges from the start node n_0 to nodes n_{i1} ($i\in [1, n]$) corresponding to the first operation of each job, and edges from nodes n_{im} ($i\in [1, n]$) corresponding to the last operation of each job to the end node n_{n+1}. The start node has n successors but no antecedence while the end node has n antecedence but no successor. Each of the other nodes has exactly one antecedence and one successor.

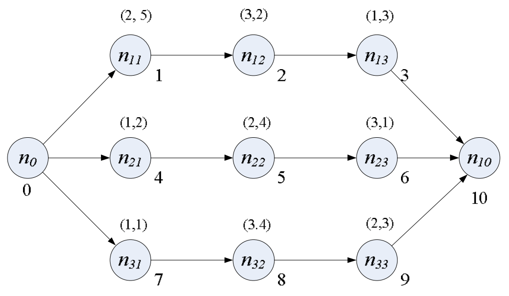


Fig. 1.Graphic representation of a 3×3 JSP (FT03).

Fig.1 is an example of graph based on the 3×3 JSP (FT03) [13]. Ignoring the start node in left and the end node in right, each row of the middle matrix includes all the operations in one job positioned in required processing order. Since there is no precedence between operations belonging to different jobs, no directed edge exist crossing different rows. Bracketed numbers (u, t) above each node inform that the operation should be processed on machine u with a processing time t.

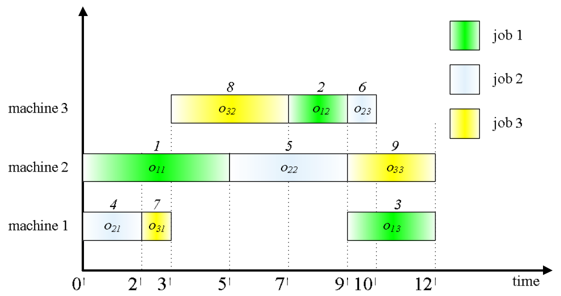


Fig.2.Processing of 3×3 JSP (FT03) according to the schedule (4, 1, 7, 8, 2, 3, 5, 9, 6).

A solution of JSP is represented by a permutation of operation indexes in their scheduled order. For example, (4, 1, 7, 8, 2, 3, 5, 9, 6) is a feasible solution to the above JSP. The total processing time according to this solution can be calculated hewing to constraints. On one hand, when a machine is busy with one operation, the next operation using this machine should wait until the machine finishes the current operation; on the other hand, when an operation is on processing, its successor (if it has any) should wait until the former one finishes even the required machine of the successor is idle. Fig.2 shows the processing.

As Fig.2 shows, the overall processing time C_{max} of this solution is 12. Though the machine is not at work in some period (machine 3 in time period 1~3 and machine 1 in time period 3~9), none of the waiting operators can be move ahead, restricted by their unfinished antecedences. It is also notable that the operation 5 positioning after operation 2 and 3 in the schedule solution, it actually gets to be processed earlier than operation 2 and 3, as the later two are suspended by the busy machine or their unfinished antecedence. This example shows us, the solution gives the order of scheduling priority rather than the actual starting time.

III. ANT COLONY SYSTEM FOR JSP

An ant colony is good at finding a shortest path from their nest to the food source. At the very start of the ants' foraging process, the anterior ants have no cues or visibility about the various paths leading to the food. They stochastically choose each path with equal possibility, say, the number of ants choosing each path is the same on average. While walking, ants deposit a trail of pheromone along the way. Pheromone is a communication tool for ants which is attractive to them. As the ants choosing a shorter path come back faster, pheromone on these paths consequentially accumulates faster. Thereby, when the latter ants arrive at a fork, the pheromone on the shorter path would be stronger, which allure the ants choose these paths with larger possibility. With more ants passing by, the shorter paths in turn receive more pheromone and become more attractive to the following ants. Over a period of time, the ants will all choose the shortest path.

When transformed to a manual system, the algorithm system does not simulate the back track of the ants. The convergence to the shortest path is brought on by ants' iterative path-building from the starting node to the end node. At each step of the ants, the decision of path-choosing is made according to a *State Transition Rule* based on the pheromone bias and an additional heuristic bias. Pheromone thickness is controlled by two update rule. A *Local Updating Rule* is executed at the same time of the ants' path-building to control the pheromone accumulation and evaporation, while a *Global Updating Rule* is applied only on the best path after all the ants complete their tour to make it preponderate in the next path-building iteration. The whole process of the ACS algorithm is showed as Fig.3.

A. Path-building

By adding two dummy nodes identifying the start and the completion of the overall job-shop, the ants can build a path from the start node to the end node passing (processing) each of the others corresponding to all the operations exactly once.

a. Eligible set

At the beginning of each iteration, all the ants are placed on the start node and then incrementally build their own paths step by step. Unlike the Traveling Salesman Problem, when standing on a specific node, ants cannot choose any of the unvisited nodes to move to. The set of nodes eligible for the ants to choose from is only a subset of the unvisited set due to the constraints of the required operation order. When applying the ants' path-building, we dynamically keep the eligible set S_k for ant k according to the constraints. At the beginning of the iteration with all the ants on the start node, S_k is initialed to be the first operations of all the jobs, that is $S_k = \{n_{i1} \mid i \in [1, n]\}$, which means the overall job-shop can begin with any one of the jobs. When ant k choose node n_{ij} from S_k to move to, n_{ij} is deleted from S_k and the immediate successor of n_{ij} (if it is not the last operation of a job) is added to S_k. Node that the end node is the successor of nodes n_{im} ($i \in [1,n]$ and m is the number of machine) corresponding to the last operation of all the jobs, but it can't be added into S_k only until all his antecedences are finished. The path-building process is completed when the eligible set is empty.

b. State Transition Rule

When standing on node r, ants choose a node s from the eligible set S_k to continue their tour by applying a stochastic greedy rule called *State Transition Rule* in form of formulation (1).

$$s = \begin{cases} \arg\ \max_{u \in S_k} \{[\tau(r,u)] \cdot [\eta(u)^{\beta}]\}, & \text{if } q \le q_0 \text{ (exploitation)} \\ S & \text{, otherwise (biased exploration)} \end{cases} \quad (1)$$

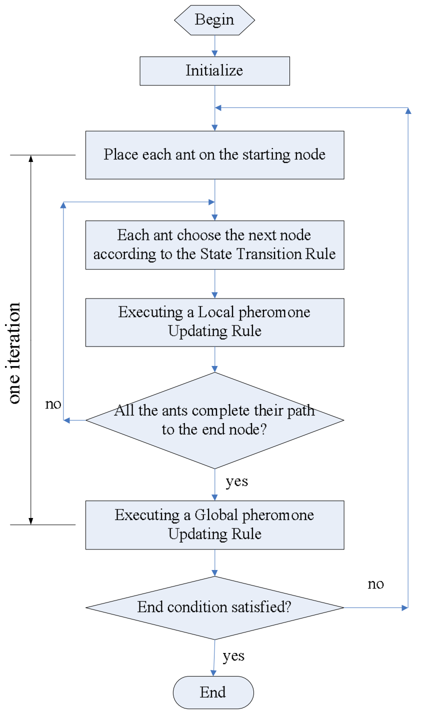
This rule consists of two parts: exploitation with probability q_0 and biased exploration with probability $(1\text{-}q_0)$. q is a random number uniformly distributed in [0,1] which is randomly gained to decide executing exploitation or biased exploration when an ant is about to choose a next node. If $q \leqslant q_0$, ants would applying exploitation and choose the next node deterministically according to the upper part of (1); Otherwise, a biased exploration is carried on and S in (1) is the decision made according to (2):

$$P_k(r,s) = \begin{cases} \dfrac{[\tau(r,s)] \cdot \eta(s)^{\beta}}{\sum_{u \in S_k} [\tau(r,u)] \cdot \eta(u)^{\beta}}, & \text{if } s \in S_k \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

$P_k(r,s)$ is the probability for ant k on node r to choose s to move to.

c. Pheromone

In (1) and (2), $\tau(r,u)$ and $\eta(u)$ are the two factors that the ants' decision depend on. Traditionally, $\tau(r,u)$ stands for the pheromone thickness on edge *(r, u)*. In other words, $\tau(r,u)$ is the desirability of scheduling operation u behind operation r. However, we find in our research that this type of pheromone is some kind of redundant that entails the ants sparing labors on unnecessary advisement. Compare another solution to the FT03 case (4, 7, 1, 8, 2, 3, 5, 9, 6) with the above mentioned solution (4, 1, 7, 8, 2, 3, 5, 9, 6). These two solutions actually lead to a same Gant graph as Fig 2 shows. Since operation 1 and 7 require different machines, the choice between 4->1 and 4->7 doesn't make much sense. In fact, the actual scheduling that matters and need consideration is the order of operations from different jobs on the same machine. What we actually do in the scheduling is to order operations on the same machine. Though operations from different jobs on different machines restrict each other to some degree, the start time of an operation directly depends on its antecedences and its position on the requiring machine.



Begin
Initialize
Place each ant on the starting node
Each ant choose the next node according to the State Transition Rule
Executing a Local pheromone Updating Rule
All the ants complete their path to the end node?
no
yes
Executing a Global pheromone Updating Rule
one iteration
End condition satisfied?
no
yes
End

Fig.3.The ACS algorithm flow.

This paper proposed a novel strategy of the scheduling mechanism and the pheromone design. To be more directly and effectively, solutions are incrementally built in the traditional way without separating the machines. But the pheromone is no longer the desirability of the connection between two operations. In our implementation, pheromone $\tau(u,p)$ indicates the desirability of placing operation u at the pth position on the requiring machine.

Hence, before calculating the possibility of each operation in the eligible set, the position on machine of each candidate (if chosen) should be figured out first. And the *State Transition Rule* is modified as (3).

$$P_k(r,s)=\begin{cases}\dfrac{\left[\tau(s,s_p)\right]\cdot\left[\eta(s)\right]^{\beta}}{\sum\limits_{u\in S_k}\left[\tau(u,u_p)\right]\cdot\left[\eta(u)\right]^{\beta}}, & \text{if } s\in S_k\\ 0, & \text{otherwise}\end{cases} \quad (3)$$

d. Heuristic Information

$\eta(u)$ stands for the heuristic information of the candidate node u. In the problems concerning distance between nodes like TSP, $\eta(u)$ is usually be defined as the inverse of the distance. In the time-concerned JSP, various types of time have been used as the heuristic information in literatures [40]. In the studies of heuristic information for JSP, TLJ (time-left for job) is demonstrated to be an effective heuristic guide, which are defined as the remaining required time for each job to be processed. In this point of view, an operation belonging to a job that remains more time for processing would be preferred. In order to balance the magnitude of $\tau(u,p)$ and $\eta(u)$, $\eta(u)$ is defined as follows:

$$\eta(u)=TLJ/(n*m) \quad (4)$$

By repeatedly applying the *State Transition Rule* on each step of the tour, ants finally complete a path to the end node representing a scheduling solution to the JSP. Fig.4 shows the path corresponding to the foregoing example solution (4, 1, 7, 8, 2, 3, 5, 9, 6).

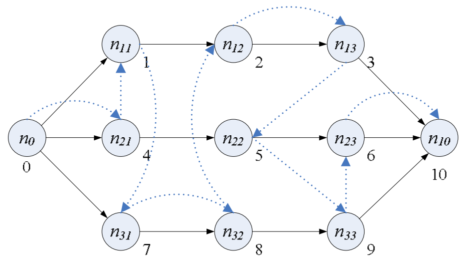


Fig. 4. Ant's path corresponding to solution (4, 1, 7, 8, 2, 3, 5, 9, 6) with doted edges indicating the path.

B. Pheromone Updating

As mentioned before, the pheromone thickness is controlled by two pheromone updating rules simulating pheromone reinforcement and evaporation.

a. Local Updating Rule

While constructing a tour, each ant will modify the amount of pheromone on the way by applying the local updating rule.

$$\tau(r,s)\leftarrow(1-\rho)\cdot\tau(r,s)+\rho\cdot\tau_0 \quad (3),$$

where ρ $(0<\rho<1)$ is the coefficient representing pheromone evaporation rate, and τ_0 is a predefined constant. This local updating takes place at each step of ants' path-building.

b. Global Updating Rule

Once all ants have arrived at their destination – the end node, pheromone is modified again by applying the global updating rule.

$$\tau(r,s)\leftarrow(1-\alpha)\cdot\tau(r,s)+\alpha\cdot\Delta\tau(r,s) \quad (4),$$

where

$$\Delta\tau(r,s)=\begin{cases}(L_{gb})^{-1}, & \text{if } (r,s)\in global-best-tour\\ 0, & \text{otherwise}\end{cases} \quad (5).$$

Here $0<\alpha<1$ is the pheromone reinforcing parameter, and L_{gb} is the length of the globally best tour from the beginning of the trial. $\Delta\tau(r,s)$ is the pheromone addition on position *(r, s)*. We can see that only the pheromone on the globally best tour can be increased.

C. *Solution Evaluation*

Since our ultimate goal of settling JSP is to minimize the overall processing time, then the makespan of a scheduling solution is the measurement of the solution quality. The calculation of the overall processing time of one solution is showed as Fig.5.

```
Set up two arrays: $M\_time[m]$ and $J\_time[n]$;
/* $m$ and $n$ are the number of machines and jobs. */
/* $M\_time[i]$: the time machine $i$ has used so far. */
/* $J\_time[j]$: the time job $j$ has used so far. */
Let $M\_tme[0...m]$=0 and $J\_time[0...n]$=0;
From the first operation of the tour to the last one do
/* The processing time of $n_{ij}$ is $p_{ij}$. The machine for $n_{ij}$ is $n_{ij}.machine$. */
    Let t = MAX($M\_time[n_{ij}.machine], J\_time[i]$);
    $M\_time[n_{ij}.machine] = J\_time[i] = t + p_{ij}$;
Return (max($J\_time[i]$), $i \in [1,n]$)
```

Fig.5.Pseudocode for computing the overall processing time of a solution.

IV. Experiment and Expectation

Experiments on the application of ACS algorithm in JSP are carried on six classic JSP cases [13] including FT06(6×6), ABZ6(10×10), LA06(15×5), LA07(15×5), LA11(20×5) and LA36(15×15). Number in bracket following the case name indicates the scale n (number of jobs) and m (number of machines) of the case. Each case is tested for 50 trials to ensure the reliability. In one trial, ants' path-building process is iterated for 10000 iterations.

Parameter setting for ACS is also pre-tested for obtaining better results. The best setting according to experimental observation is as follows: $\alpha = 0.1$, $\beta = 1.0$, $\rho = 0.1$ and $q_0 = 0.8$. What's more, the initial pheromone τ_0 on the edges is suggested as $\tau_0 = (n*L_{nn})^{-1}$, where L_{nn} is the length of the path produced by nearest neighbor heuristic algorithm and n the number of cities [39]. In our JSP case, τ_0 is set to be T_{nn}^{-1}, where T_{nn} is the overall processing time of the scheduling solution produced by nearest neighbor heuristic algorithm.

In the experiments, results obtained by employing different type of pheromone are compared. One type of pheromone is the traditional one laid on the edge connecting two operations, and the other is the proposed one laid on the position on the requiring machine of an operation, referred to as Phe_E and Phe_PM respectively. Fig.6 to Fig.11 show frequency counts of results mostly in normal school distribution, in which the results of Phe_PM are always distributed on the left of Phe_E indicating less processing time. Especially in case FT06(6×6) and LA06(15 × 5), Phe_PM can always obtain the best-known optimal while Phe_E failed to or rarely do. TABLE 1 lists the best and mean results. It is obvious that as the scale of the problem enlarges, the difficulty of optimally scheduling a JSP case increases rapidly.

Though the results obtained by the simple implementation of ACS are not very satisfied, we are primly intended to introduce the effectiveness of ACS for JSP and a new pheromone design which is better than the traditional one. There is still much room for improvement. Researchers have suggested that hybrid method involving different optimization algorithm should be more effective for JSP [3], [37]-[38]. Based on the simple adaptation of ACS for JSP, other algorithm can be employed as assistance, for example, by means of a local search, to improve the performance of ACS.

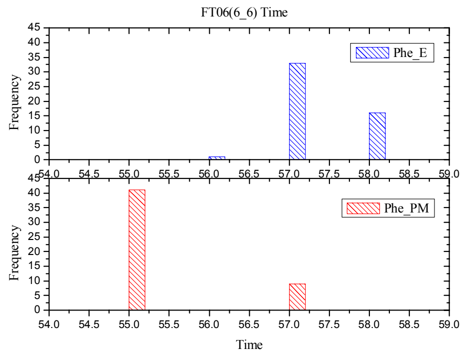


Fig. 6.Comparison of results employing Phe_E (pheromone on edge) and Phe_PM (pheromone on position of a machine) in case FT06(6_6).

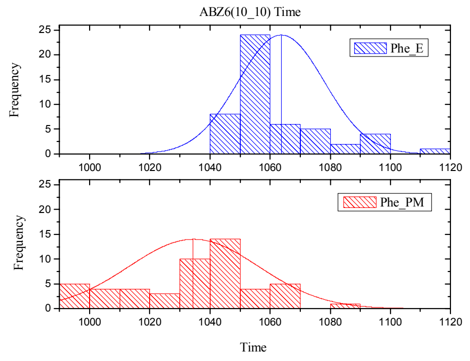


Fig. 7.Comparison of results employing Phe_E (pheromone on edge) and Phe_PM (pheromone on position of a machine) in case ABZ6(10_10).

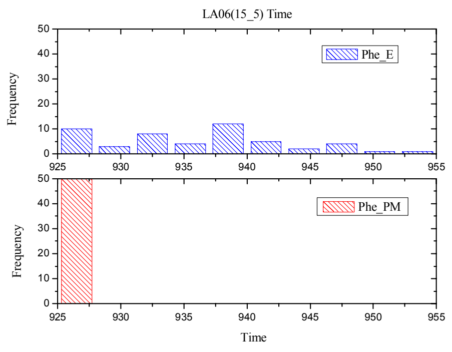


Fig. 8.Comparison of results employing Phe_E (pheromone on edge) and Phe_PM (pheromone on position of a machine) in case LA06(15_5).

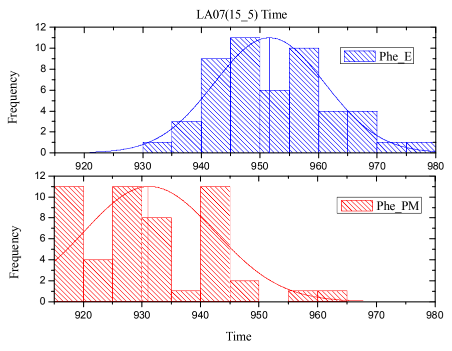


Fig. 9.Comparison of results employing Phe_E (pheromone on edge) and Phe_PM (pheromone on position of a machine) in case LA07(15_5).

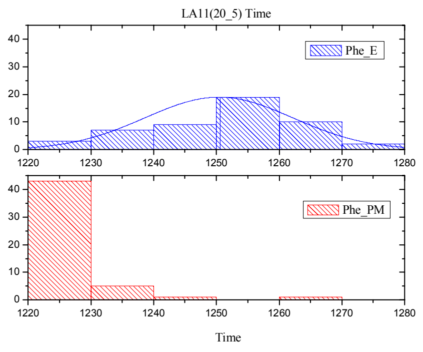


Fig. 10.Comparison of results employing Phe_E (pheromone on edge) and Phe_PM (pheromone on position of a machine) in case LA11(20_5).

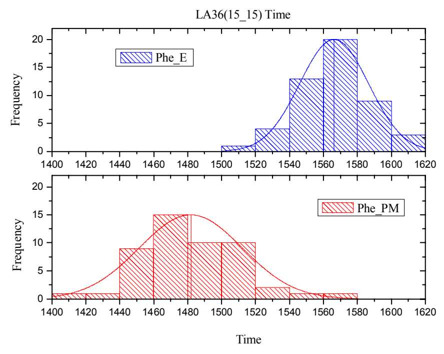


Fig. 11.Comparison of results employing Phe_E (pheromone on edge) and Phe_PM (pheromone on position of a machine) in case LA36(15_15).

Table I
The Best and Mean Results of six JSP cases

JSP case	Best known	Phe_E		Phe_PM	
		Best	Mean	Best	Mean
FT06(6_6)	55	56	57.3	55	55.36
ABZ6(10_10)	943	1047	1063.7	995	1034.62
LA06(15_5)	926	926	935.58	926	926
LA07(15_5)	890	930	951.62	917	931.28
LA11(20_5)	1222	1222	1250.4	1222	1224.58
LA36(15_15)	1268	1512	1566.28	1416	1481.82

V. CONCLUSION

This chapter gives a detail description of the application of ACS algorithm for JSP, a strategy of path-building with novel pheromone is proposed and experiment results are compared. It has been shown that ACS can simply adapt to the discrete combination optimization problem – JSP and obtain moderate result within reasonable computation time. The algorithm employing the proposed pheromone design can provide better result.

ACKNOWLEDGMENT

The authors would like to thank associate editor and reviewers for their valuable comments and suggestions.

REFERENCES.

[1] Garey EL, Johnson DS, Sethi R., "The complexity of flowshop and job-shop scheduling", Mathematics of Operations Research 1976; 1:117-29.
[2] Muth JF, Thompson GL., "Industrial scheduling", Englewood Cliffs, NJ: Prentice-Hall; 1963.
[3] Chao Yong Zhang, PeiGen Li, YunQing Rao and ZaiLin Guan, "A very fast TS/SA algorithm for the job shop scheduling problem", Computers & Operations Research, In Press, Corrected Proof, Available online 3 April 2006.
[4] Balas, E., "Machine sequencing via disjunctive graphs: an implicit enumeration algorithm", Oper. Res., 17, pp. 941-957, 1969.

[5] McMahon G. and M. Florian, “On scheduling with ready times and due dates to minimize maximum lateness”, Oper. Res., 23, 3, pp. 475-482, 1975.

[6] Barker J.R. and G.B. McMahon, “Scheduling the general job-shop”, Manage. Sci., 31, 5, pp. 594-598, 1985.

[7] Carlier J. and E. Pinson, “An algorithm for solving the job-shop problem”, Manage, Sci., 35, 2, pp- 164-176, 1989.

[8] Lageweg BJ, Lenstra JK, Rinnooy Kan AHG., “Job-shop scheduling by implicit enumeration”, Management Science, 1977; 24:441-50.

[9] Croce FD, Tadei R, Volta G., “A genetic algorithm for the job shop problem”, Computers & Operations Research, 1995; 22:15-24.

[10] J. Blazewicz, W. Domschke, E. Pesch, “The job shop scheduling problem: Conventional and new solution techniques”, European Journal of Operational Research, 93 (1996) 1-33.

[11] Croce FD, Tadei R, Volta G., “A genetic algorithm for the job shop problem”, Computers & Operations Research, 1995; 22:15-24.

[12] Dorndorf U, Pesch E., “Evolution based learning in a job shop scheduling environment”, Computers & Operations Research, 1995; 22:25-40.

[13] Wang Ling., Shop Scheduling with Genetic Algorithm. TstingHua University Press, 2003, pp 59 – 67.

[14] Candido M A B, Khator S K, Barcia R M., “A Genetic algorithm based procedure for more realistic job shop scheduling problem”, International Journal of Production Research, 1998, 36 (12):3437-3457.

[15] Mati, Y., Rezg, N., and Xiaolan Xie., “An integrated greedy heuristic for a flexible job shop scheduling problem”, Systems, Man, and Cybernetics, 2001 IEEE International Conference on, Volume: 4, 7-10 Oct. 2001, Pages:2534-2539 vol.4.

[16] Masato Watanabe, Kenichi Ida and Mitsuo Gen, “A genetic algorithm with modified crossover operator and search area adaptation for the job-shop scheduling problem”, Computers & Industrial Engineering, Vol. 48, Issue 4, June 2005, Pages 743-752.

[17] Morikawa, K., Furuhashi, T., Uchikawa, Y., “Single populated genetic algorithm and its application to job-shop scheduling”, Industrial Electronics, Control, Instrumentation, and Aotomation, 1992. ‘Power Electronics and Motion control’, Proceedings of the 1992 International Conference on 9-13 Nov. 1992 Page(s): 1014-1019 vol.2.

[18] Gen, M., Tsujimura, Y., Kubota, E., “Solving job-shop scheduling problems by genetic algorithm”, System, Man, and Cybernetics, 1994. ‘Humans, Information and Technology’, 1994 IEEE International Conference on, Vol.2, 2-5 Oct. 1994 Page(s): 1577-1582 vol.2.

[19] Van Laarhoven PJM, Aarts EHL, Lenstra JK., “Job shop scheduling by simulated annealing”, Operations Research, 1992; 40:113-25.

[20] Steinhofel K, Albrecht A, Wong CK., “Two simulated annealing-based heuristics for the job shop scheduling problem”, European Journal of Operational Research, 1999; 118(3):524-48.

[21] Peter J. M. van Laarhoven, Emile H. L. Aarts, and Jan Karel Lenstra, “Job shop scheduling by simulated annealing”, Operations Research, Volume 40, Issue 1, Jan.-Feb. 1992, Pages 113-125.

[22] H. Matsuo, C. J. Suth, and R. S. Sullivan, “A Controlled search Simulated Annealing Method for the General Job-Shop Scheduling Problem”, Austin, TX: Grad. School of Bus., Univ. of Texas, 1988.

[23] Dell'Amico M, Trubian M., “Applying tabu search to the job shop scheduling problem”, Annual Operations Research, 1993; 40:231-52.

[24] Taillard ED., “Parallel taboo search techniques for the job-shop scheduling problem”, ORSA Journal on Computing, 1994; 6:108-7.

[25] Nowicki E, Smutnicki C., “A fast taboo search algorithm for the job shop scheduling problem”, Management Science, 1996; 42(6):797-813.

[26] Guohua Wan, Feng Wan, “Job shop scheduling by taboo search with fuzzy reasoning”, Systems, Man and Cybernetics, 2003. IEEE International Conference on, Volume: 2, 5-8 Oct. 2003, Pages 1566-1570 vol.2.

[27] J. W. Bames and J. B. Chambers., “Solving the job-shop scheduling problem using tabu search”, IIE Transaction, vol. 27, pp. 257-263, 1995.

[28] Chuan Yu Chang, Mu Der Jeng, “Experimental study of a neural model for scheduling job shops”, Systems, Man and Cybernetics, 1995. ‘Intelligent Systems for the 21st Century’, IEEE International Conference on, Vol. 1, 22-25 Oct. 1995 Page(s): 536-540 vol.1.

[29] D.Y. Sha and Cheng-Yu Hsu, “A hybrid particle swarm optimization for job shop scheduling problem”, Computers & Industrial Engineering, Vol. 51, Issue 4, December 2006, Pages 791-808.

[30] A. Colorni, M. Dorigo, V. Maniezzo, and M. Trubian, “Ant System for Job-Shop Scheduling”, Belgian Journal of Operations Research, Statistics and Computer Science, 34(1): 39-54, 1993.

[31] Zhou Pin, Li Xiao-ping, and Zhang Hong-fang, “An Ant Colony Algorithm for Job Shop Scheduling Problem”, Proceedings of the 5th World Congress on Intelligent Control and Automation, June 15-19, 2004.

[32] Dorigo, Marco, Ant Colony Optimization, ISBN 0-262-04219-3.

[33] Takeshi Yamada, Ryohei Nakano, “Scheduling by Genetic Local Search with Multi-Step Crossover”, The fourth International Conference on Parallel Problem Solving from Nature, (PPSN IV) BERLIN, GERMANY, Sep. 22-26, 1996 pp. 960-969.

[34] Bagchi S., S. Uckun, Y. Miyabe, K. Kawamura, “Exploring problem-specific recombination operators for job shop”, Proc. of 4th ICGA, San Mateo, pp.10-17, 1991.

[35] Nakano R., “Conventional Genetic Algorithms for Job-Shop Problems”, Proc. of 4th ICGA, San Mateo, pp.477-479, 1991.

[36] Yamada, T. and R. Nakano, “A Genetic Algorithm Applicable to Large-scale Job-Shop Problems”, Parallel Problem Solving from Nature 2, Manner and Manderick(ed), pp.281-290, 1992.

[37] Ling Wang, Da-Zhong Zheng, “An effective hybrid optimization strategy for job-shop scheduling problems”, Computers & Operations Research, Vol. 28, Issue 6, May 2001, Pages 585-596.

[38] M. Kolonko, “Some new results on simulated annealing applied to the job shop scheduling problem”, European Journal of Operational Research, Vol. 113, Issue 1, 16 February 1999, Pages 123-136.

[39] Dorigo, M.; Gambardella, L.M., “Ant colony system: a cooperative learning approach to the traveling salesman problem”, Evolutionary Computation, IEEE Transactions on, Vol. 1, Issue 4, April 1997 Page(s): 53-66.

[40] Kuo-Ling Huang, Ching-Jong Liao, “Ant colony optimization combined with tabu search for the JSP”, Computers & Operations Research, In Press, Corrected Proof, Available online 22 August 2006

[41] J. Heinonen, F. Pettersson, “Hybrid ant colony optimization and visibility studies applied to a job-shop scheduling problem”, Applied Mathematics and Computation, In Press, Corrected Proof, Available online 18 October 2006