An Effective Ant Colony Optimization-Based Algorithm for Flow Shop Scheduling

Ruey-Maw Chen, Shih-Tang Lo, Chung-Lun Wu, and Tsung-Hung Lin

Abstract—**This article presents a modified scheme named local search ant colony optimization algorithm on the basis of alternative ant colony optimization algorithm for solving flow shop scheduling problems. The flow shop problem (FSP) is confirmed to be an NP-hard sequencing scheduling problem, which has been studied by many researchers and applied to plenty of applications. Restated, the flow shop problem is hard to be solved in a reasonable time, therefore many meta-heuristics schemes proposed to obtain the optima or near optima solution efficiently. The ant colony optimization (ACO) is one of the well-applied meta-heuristics algorithms, nature inspired by the foraging behavior of real ants. Different implementations of state transition rules applied in ACO are studied in this work. Meanwhile, a local search mechanism was introduced to increase the probability of escaping from local optimal. Hence, this work integrates the local search mechanism into ant colony optimization algorithm for solving flow shop scheduling problem to improve the quality of solutions. Simulation results demonstrate that the applied "random order" state transition rule used in ACO with local search integrated is an effective scheme for the flow shop scheduling problems .**

Index Terms—**ant colony optimization (ACO) , flow shop problem (FSP) , local search, meta-heuristics, Scheduling.**

I. Introduction

There are many different types of combinational optimization problem applications in real world, such as job shop problem, task assignment problem, project scheduling problem, and the flow shop problem [1, 2, 3, 4]. However, solving these problems for an optimal solution is usually quite time consuming. For example, the FSP is known to be *NP-hard* [5, 6], the search space is composed of *n!* possible sequences, hence it is impracticable to solve the FSP by exact algorithms. Instead, many approximation algorithms and heuristic methods have been studied for searching near optima solution with faster calculation. However, these ways yielded limited quality of solutions.

To efficiently obtain the high-quality solutions, many meta-heuristics based methods have been proposed for solving these problems, such as genetic algorithm (GA) [7], simulated annealing algorithm (SA) [8], tabu search (TS) [9], particle swarm optimization (PSO) [10] and the ant colony optimization (ACO) [11]. Meanwhile, the ACO has demonstrated as a good alternative to existing algorithms for hard combinatorial optimization problems. Moreover, the flow shop problem has been applied in plenty of applications fields. Accordingly, this study focuses on applying the ACO to solve the flow shop problem. The ACO is originated by Dorigo et al. [11], which is a powerful nature-inspired algorithm for optimization problems. In ACO, the ants exchange information by leaving pheromone on the path, and they would intend to follow the path with higher amount of pheromone for searching shortest path, and hence for the optimum solution of the interested problem. The ACO has been verified to be applicable for variable combinatorial problems and scheduling problems. However, the solution obtained from ACO is dependent on the exploitation strategy of state transition rule. Therefore, three different ways of state transition rules, i.e., "in order", "random order", and "Pheromone-related", are investigated in this study. Additionally, a similar interchange-moves local search mechanism [12] is also applied in this work. The use of the local search mechanism enables the ACO scheme to find global optimal solution.

This article is organized as following: section 2 describes definition of the problem studied in this paper. Section 3 introduces and describes the ACO with local search scheme to solve flow shop problem. The simulated cases and results of experiments are displayed in section 4. Finally, section 5 presents the conclusions and discussions.

II. Problem definition --

Flow shop scheduling problem is one of the most well-known production scheduling problems with strong industrial background [6]. The flow shop problem is hard to solve by giving the proper order for job sequence. The scheduling flow shop scheduling problem studied in this investigation is defined as following:

Manuscript received Jan 31, 2008.

Ruey-Maw Chen. Author is with the Department of Computer Science and Information Engineering, National Chinyi University of Technology, Taiping, Taichung 411, Taiwan, R. O. C. (corresponding author phone: 886-4-2392-4505ext.8726,Fax:886-4-23920892;e-mail:raymond@mail.ncut.edu.tw).

Shih-Tang Lo. Author is with the department of information management, Kun-Shan University, Yung-Kang, Tainan 701, Taiwan, R. O. C. (e-mail: edwardlo@mail.ksu.edu.tw).

Chung-Lun Wu, Author is a graduated student with National Chin-yi Univ. of Technology, Taiping, Taichung 411, Taiwan, R. O. C. (e-mail: arashilen@yahoo.com.tw).

Tsung-Hung Lin, Author is with the Department of Computer Science and Information Engineering, National Chinyi University of Technology, Taiping, Taichung 411, Taiwan, R. O. C. (e-mail: duke@mail.ncut.edu.tw).

978-1-4244-3782-5/08/$25.00 ©2008 IEEE

1. A set $N = (1,...,n)$ consists of n independent jobs, and the set $M = (1,...,m)$ consists of m independent machines in the scheduling system. These n jobs are to be processed on m machines.
2. Every job j ($j \in N$) which comprises m operations $o_{j,k}$ ($k=1,...,m$) must be processed with the duration $d_{i,j}$ on every machine i ($i \in M$), all of the jobs are non-segmented and non-preemptive in the scheduling. Operation $o_{j,k}$ ($k = 2, ..., m$) can not start until $o_{j,k-1}$ is completed.
3. All of the jobs should be processed with the same permutation (order) $S=\{s_1,\ldots,s_n\}$ in all machines from the fist machine to the final machine, where $s_r \in S$ indicating the r^{th} order executed job. Meanwhile, S presents the solution of the FSP.
4. The objective is widely defined as finding the minimization of *makespan* (total completion time). The solution with proper permutation of jobs is critical for the minimization of *makespan*.

To facilitate understanding of the studied flow shop scheduling problem, an example which has 3 jobs and 5 machines is demonstrated. And the processing time duration is shown in TABLE.1.

TABLE 1.
DURATION $D_{i,j}$ FOR REPRESENTATIVE EXAMPLE.

Machine \ Job	1	2	3
1	2	3	4
2	3	1	1
3	1	2	4
4	2	4	2
5	4	2	1

Two solutions of this example: $S_1=\{2,1,3\}$ and $S_2=\{3,1,2\}$ are schematized in Fig. 1 and Fig. 2 respectively, where the solution S_1 is with shorter *makespan*. Restated, different jobs order would yields different *makespan*. However, finding the shortest *makespan* is a typical NP-hard combinatorial optimization problem.

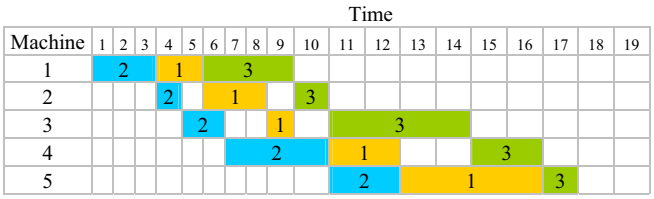

Fig. 1. The gantt chart of S_1.

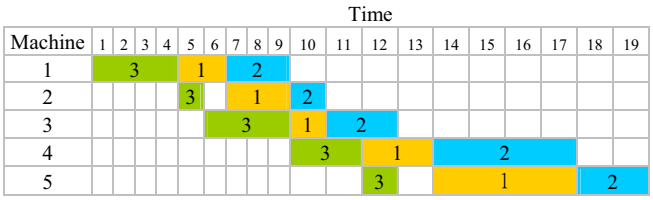

Fig. 2. The gantt chart of S_2.

III. SOLVING FLOW SHOP PROBLEM BY ANT COLONY OPTIMIZATION WITH LOCAL SEARCH

A. The background of ACO

The ACO emulates the exchanging information mechanism of ants for searching shortest path [13], where the information is the pheromone laid on the traveled path, and the amount of pheromone left by ants is inverse proportional to the length of path. Therefore, the shorter path the higher pheromone concentrated, and attracts more ants to follow the path.

1. Initialize
2. Loop
3. Each ant k is positioned on a starting node
4. Loop
5. Initialize J_k
6. while $J_k \neq \varphi$ do
7. Select one job $j_k \in J_k$ to work at order r by *state transition rule*
8. $J_k = J_k - \{j_k\}$
9. apply *local update rule*
10. end while
11. Until all ants have built a complete solution
12. A *global updating rule* is applied with the best solution
13. Using local search to generate new solutions
14. Until End condition is reached

Fig. 3. The pseudo-code of ACO for problem

B. Pseudo code of ACO with local search for problem

Figure 3 shows the pseudo-code of scheduling algorithm for the FSP studied in this study.

In the ACO, each ant parallel generates one solution. In an iteration, there are K solutions generated, where K is the number of ants. Each ant uses the local update rule to update pheromone value (step 9). Among the K generated solutions, the ACO algorithm applies the best solution to increment the pheromone by using the global update rule (step 12). Furthermore, a local search in step 13 is utilized to improve the solution quality.

C. State transition rule

A solution is the permutation of jobs and expressed by a queue. The state transition rule is used to select the job j_k ($j_k \in J_k$) for running at the r^{th} order by ant k, where the J_k is the set of all the candidate jobs for ant k. If the job j_k has been selected as the r^{th} order job, then $J_k=J_k-\{j_k\}$, $s_r=j_k$ ($s_r \in S_k$; the solution constructed by ant k). The common way of applying state transition rule is that selecting job repeatedly for running in the increasing order. Restated, the jobs selected for s_r, r is increasing progressively. Finally, the construction of solution S_k is completed when $J_k=\varphi$. The other state transition rule related operations will be discussed in section 3.5.

The state transition rule includes two strategies: "exploitation" and "exploration". When $q<q_0$, the exploitation

strategy is taken, otherwise, the exploration strategy is applied. Where the q is a random number uniformly distributed in [0, 1], and $0 \le q_0 \le 1$ is a predetermined parameter that determines the relative importance of that "exploitation" versus "exploration".

The exploitation strategy leads ant k to select the job which is with maximum pheromone–greediness product. Restated, the desired r^{th} order job is j_k. The fundamental exploitation is based on the Eq. (1).

$$j_k = \max_{j \in J_k} \{[\tau(j,r)]^{\alpha} \times [\eta(j,r)]^{\beta}\} \quad (1)$$

The product of pheromone value $\tau(j, r)$ and heuristic value $\eta(j, r)$ in Eq. (1) is corresponds to the probability for selecting job j work at the r^{th} order. Moreover, the parameter α and β control the relative importance of the pheromone versus the greediness (heuristic) information.

The exploration strategy makes the job for order r (j_k)is randomly selected from J_k according to the probability distribution given by Eq. (2):

$$p_k(j,r) = \begin{cases} \dfrac{[\tau(j,r)]^{\alpha} \times [\eta(j,r)]^{\beta}}{\sum_{l \in J_k} [\tau(l,r)]^{\alpha} \times [\eta(l,r)]^{\beta}}, & \text{if } j \in J_k, \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

However, the heuristic information is omitted in the study, i.e., the $\eta(j, r)$=1. Furthermore, the control parameter α and β can then be omitted. Therefore, Eq. (1) and Eq. (2) are simplified as follows.

$$j_k = \max_{j \in J_k} \{[\tau(j,r)]\} \quad (3)$$

$$p_k(j,r) = \begin{cases} \dfrac{[\tau(j,r)]}{\sum_{l \in J_k} [\tau(l,r)]}, & \text{if } j \in J_k, \\ 0, & \text{otherwise.} \end{cases} \quad (4)$$

D. Updating rules

Once an ant selects one job j to work at the r^{th} order, the corresponding pheromone $\tau(j, r)$ is updated using the local updating rule as following:

$$\tau(j,r) \leftarrow (1-\rho) \times \tau(j,r) + \rho \cdot \tau_0 \quad (5)$$

The $0<\rho<1$ is the evaporation rate, and τ_0 is the initial pheromone value. The pheromone $\tau(j, r)$ would be depressed to a lower value after applying the local updating rule. This mechanism is designed to prevent the remained ants from been hell-bent selecting jobs selected by previous ant converging quickly.

After all ants have built their solutions, the global update rule is used to increase the pheromone $\tau(j, r)$ which corresponds to the best solution in current iteration. The global update rule is defined as following:

$$\tau(j,r) \leftarrow (1-\delta) \times \tau(j,r) + \delta \cdot \Delta\tau(j,r) \quad (6)$$

The $0<\delta<1$ is the decay rate. The $\Delta\tau(j, r)$ is computed as below:

$$\Delta\tau(j,r) = \begin{cases} \dfrac{C}{ms_{gb}}, & \text{if } (j,r) \in \text{the global best solution} \\ 0, & \text{otherwise.} \end{cases} \quad (7)$$

The $\Delta\tau(j, r)$ is inverse proportion to ms_{gb} ; the best (smallest) *makespan* in current iteration, and the C is a proper chosen constant to ensure $\Delta\tau(j, r)$ is increased enough.

E. Implementation of state transition rule

To facilitate explanation, a pheromone map with case n=5 is used as illustrated in TABLE 3, and that q_0=1 is assumed. Restated, the state transition rule would select the job with maximum pheromone directly. There are three ways of state transition rule applied in this study. They are

1. In order:

Repeat selecting job for running at the increasing or decreasing order. Suppose that increasing order is adopted. At first , s_1=2 because of $\tau(2, 1)=\max\{\tau(j, 1)\}, j \in J_k$, then $2 \notin J_k$, and s_2=3 . Eventually, the solution will be S_k ={2,3,4,5,1}.

TABLE 2.
4 EXAMPLES FOR RANDOM ORDER

Sequence of r	S_k
5,1,2,4,3	2,3,1,5,4
1,3,4,5,2	2,1,4,5,3
3,5,1,2,4	2,3,4,1,5
2,5,1,4,3	3,2,1,5,4

2. Random order:

Repeat selecting job for execution at the random order. If the random sequence of r is (5,1,2,4,3) indicating the s_5=4 is the first, then s_1=2 is the second, and so on .The solution should be S_k ={2,3,1,5,4}. There are 4 examples as follows.

Accordingly, the solution depends on the determined random sequence of r.

3. Pheromone-related:

Repeat selecting job for running at the order decided by pheromone value. Restated, the higher $\tau(j, r)$ value in pheromone map has the higher selection priority. For instance, the highest $\tau(2, 2)$=0.08 is the first chosen, then select the second highest $\tau(4, 5)$=0.075, the else are $\tau(5, 3)$, $\tau(3, 1)$, $\tau(1, 4)$. Hence, the sequence of r is (2,5,3,1,4) and S_k ={3,2,5,1,4}.

TABLE 3.
THE PHEROMONE MAP

Job / Order	1	2	3	4	5
1	0.031	0.032	0.03	0.033	0.03
2	0.05	0.08	0.04	0.04	0.037
3	0.035	0.04	0.034	0.032	0.05
4	0.04	0.03	0.06	0.07	0.075
5	0.032	0.031	0.055	0.05	0.06

The comparison of these three operations is displayed in TABLE 4 as below.

The state transition operation is the decisive step that affects the complexity of the method studied in this paper. The operation of "In order" and "Random order" is just visits the n^2 size pheromone map once. Comparatively, the "Pheromone-related" needs to sort the pheromone map. If the pheromone map sorted by quick sort scheme, the computation complexity would be $O(n^2 \log n^2)$.

F. Local search

Some local search structures for FSP have been studied and used such as simulated annealing (SA), and an effective local search method is integrated into the ant colony optimization in this work. This local search method is named interchange-moves in other papers. Restated, a generated source solution has $s_i=a$, and $s_j=b$, then a new solution could be constructed by interchange the order of jobs a and b when the local search process applied, i.e., the $s_i=b$, and $s_j= a$. For example, a source solution of a $n=5$ FSP is presented in

TABLE 5.
THE SOURCE SOLUTION

Order	1	2	3	4	5
Job#	3	4	1	5	2

TABLE 5.

Assume job 4 and job 2 are selected for local search, and then a new solution is shown in TABLE 6.

TABLE 6.
THE NEW SOLUTION

Order	1	2	3	4	5
Job#	3	2	1	5	4

In this study, the source solution is the global best solution. If a new solution constructed by applying local search is better than the source (global best) solution, then the source solution would be substituted by the new solution. The selection of target jobs to be interchanged for local search is determined by random decision. This local search operation enables this algorithm to escape from local optima solution.

IV. SIMULATIONS AND RESULTS

To evaluate the scheme studied in this investigation which is capable of solving the FSP, there are 14 classic cases were simulated by different schemes as displayed in TABLE 7. These cases are available downloaded from the OR-Library web site (http://people.brunel.ac.uk/~mastjjb/jeb/orlib/flowshopinfo.html).

In TABLE 7, the "Opt." is the optimum *makespan* of the corresponding cases, and the other values for comparison are the relative percentage deviation calculated by the best solution of 20 trails, each trail tested for 300 iterations. The relative percentage deviation is computed as bellow:

$$\frac{(best\ solution\ of\ 20\ trails - Opt.)}{Opt.} \times 100 \quad (8)$$

TABLE 7.
THE COMPARISON OF 5 SCHEMES

Case	Opt.	Random-order		Pheromone-related		Hybrid PSO
		LS	non-LS	LS	non-LS	
car1	7038	0	0	0	0	0
car2	7166	0	0	0	0	0
car3	7312	0	0	0	0	0
car4	8003	0	0	0	0	0
car5	7720	0	0	0	0	0
car6	8505	0	0	0	0	0
car7	6590	0	0	0	0	0
car8	8366	0	0	0	0	0
rec01	1247	0.1604	0.1604	0.1604	0.1604	0.16
rec07	1566	0	1.1494	0	0.3831	0.13
rec13	1930	1.6062	1.8135	1.5026	1.3472	1.4
rec19	2093	1.29	3.44	1.3856	2.5323	2.1
rec25	2513	2.3478	3.263	2.4672	3.1039	3.14
rec31	3045	2.9228	5.3859	2.4631	4.2693	5.25
average		0.5948	1.0866	0.5699	0.8426	0.87

The "average" is the average of relative percentage deviations of 14 cases.

There are 4 schemes including the combination of two ways of state transition operation and with/without local search "LS/non-LS". The 5th scheme, Hybrid PSO, is proposed in [14], its performance also included for comparison. The number of ants is set to 20 if the scheme adopts the local search hence the number of solutions constructed by local search is 20 at each iteration. Otherwise, the number of ants is 40.

The parameter settings of this simulation are: τ_0=0.01, q_0 =0.95, ρ=0.1, δ=0.15.

TABLE 7 shows that the ACO with local search yields the better simulation result among these different schemes.

The following simulation problems were from Taillard [19]. The performances of different algorithms on exactly the same test problems were compared. Taillard has produced a set of unsolved n/m/P/Cmax problems with 20 to 500 jobs and 5, 10, and 20 machines as shown in Table 8. There are 10 instances for each problem size. The test problem files are available via Taillard's web site (URL: http://www.idsia.ch/~eric/) or can be downloaded from the OR-Library web site (URL: http://ina2.eivd.ch/Collaborateurs/etd/problemes.dir/ordonnancement.dir/ordonnancement.html). The simulation terminates when either the total number of iterations reaches 5000 or the lower bound result is obtained. The ACS method using 20 ants is proposed as in [4]. The other meta-heuristics used for comparison were introduced in the work of [15], such as GA, SA, and NS. To evaluate the proposed advanced ACS

TABLE 4.
THE COMPARISON OF THREE STATE TRANSITION OPERATIONS

Way \ Property	Variety of solution	Faith in the pheromone	Computation complexity
In order	Low	Low	$O(n^2)$
Random order	High	Low	$O(n^2)$
Pheromone-related	Low	High	$O(n^2 \log n^2)$

algorithm (denoted by ACS+) on the same stated benchmark, the number of ant used is 10, and using local search to find the 10 neighborhood solutions of the current best solution.

TABLE 8.
COMPARISON OF DIFFERENT ALGORITHMS

	ACS	Palmer	SA	ACS+
20/5	1.19	10.81	1.27	0.247
20/10	1.70	15.27	1.71	1.225
20/20	1.60	16.34	0.86	1.400
50/5	0.43	5.23	0.78	0.107
50/10	1.89	13.48	1.98	1.432
50/20	**2.71**	15.46	2.86	**2.732**
100/5	0.22	2.38	0.56	0.118
100/10	1.22	9.08	1.33	0.874
100/20	2.22	13.24	2.32	2.307
200/10	0.64	4.75	0.83	0.709
200/20	1.30	12.15	1.74	2.011
500/20	1.68	6.72	0.85	1.832
Average	**1.40**	10.41	1.42	**1.250**

Ying showed that ACS outperformed GA, SA, and NS as in [4]. For the mean percentage deviation, an average of 1.40 with a maximum of 2.71 was achieved by ACS. The proposed method ACS+ using random order approach only get an average of 1.250 with a maximum of 2.732 mean percentage deviation. Table 8 demonstrates that ACS+ can effectively improve the solutions yield from Ying's and Palmer's methods.

V. CONCLUSIONS

In the investigation, the ACO scheme with the local search mechanism is applied to solve the FSP. According to simulation results, the performance of modified ACO with the designed local search outperforms other meta-heuristics schemes.

The effect of implementation of different state transition rules is thoroughly discussed in this study. The "Pheromone-related" state transition rule is tightly coupled with the pheromone, and retaining sound performance of applying local search mechanism. Comparatively, the use of "Random order" state transition rule with local search is better than that without local search applied as shown in Table 7. The "Pheromone-related" scheme seems suitable for solving the FSP. However, the performances of these schemes are different slightly when the local search mechanism adopted. Moreover, the "Random order" state transition rule has lower computation complexity. Therefore, the implementing of using "Random order" state transition rule is preferable. Restated, the "Pheromone-related" state transition rule is with high stability for solving problems. However, the "Random order" state transition rule integrated with local search is recommended for its relative low complexity and variety for other cases.

The more researches in the future would focus on the setup time or the cases with larger scale in FSP. Moreover, more schemes should be further studied for solving the FSP.

REFERENCES

[1] P.K. Jain and P.K. Sharma, "An ant colony algorithm for job shop scheduling problem;Systems," Man and Cybernetics, 2005 IEEE International Conference on Volume 1, Oct. 2005, pp.288 - 292

[2] A. Zhu and S.X. Yang, "A Neural Network Approach to Dynamic Task Assignment of Multirobots," Neural Networks, IEEE Transactions on Volume 17, Issue 5, Sept. 2006 Page(s):1278 – 1287

[3] L. Ozdamar, "A genetic algorithm approach to a general category project scheduling problem," Systems, Man and Cybernetics, Part C, IEEE Transactions on Volume 29, Issue 1, Feb. 1999 Page(s):44 – 59

[4] K. C. Ying and C. J. Liao, "An ant colony system for permutation flow-shop sequencing," Computers and Operations Research,Volume 31 , Issue 5,l 2004, pp. 791 – 801

[5] R. Kahg, Machine scheduling problems. The Hague: Martinus Nijhoff, 1967.

[6] M. R. Garey and D.S. Johnson. Computers and Intractability: a Guide to the Theory of NP-Completeness. Freeman, San Francisco, 1979.

[7] J. H. Holland, "Genetic algorithms and classifier systems: foundations and future directions," Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application, Cambridge, Massachusetts, United States Pages: 82 - 89 Year of Publication: 1987

[8] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, "Optimization by Simulated Annealing," Science 13 May 1983: Vol. 220. no. 4598, pp. 671 – 680.

[9] F. W. Glover and M. Laguna, Tabu Search ;1997 Springer.

[10] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," Proceedings IEEE Int'l. Conf. on Neural Networks, IV, (1995), pp.1942-1948.

[11] M. Dorigo, M. Birattari and T. Stützle, "Ant colony optimization," Computational Intelligence Magazine, IEEE Volume 1, Issue 4, Nov. 2006 Page(s):28 – 39.

[12] I. H. Osman and C. N. Potts, Simulated annealing for permutation flow-shop scheduling. OMEGA, 17, 1989, pp..551–557.

[13] M. Dorigo and L.M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," IEEE Transactions on Evolutionary Computation 1997; 1(1), pp. 53-66.

[14] Z. Liu and S. Wang, "Hybrid Particle Swarm Optimization for Permutation Flow Shop Scheduling," Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on Volume 1, 2006 Page(s):3245 – 3249

[15] R. R. Colin, " A genetic algorithm for &owshop sequencing," Computers and Operations Research 1995;22, pp.5 – 13.

Ruey-Maw Chen, he was born at Tainan, Taiwan, R.O.C. in 1960. He received the B. S., the M. S. and the PhD degrees in engineering science from National Cheng Kung University of Taiwan R.O.C. in 1983, 1985 and 2000, respectively.

From 1985 to 1994 he was a senior engineer on avionics system design at Chung Shan Institute of Science and Technology (CSIST). Since 1994, he is a technical staff at Chinyi Institute of Technology. Since 2002, he has been with the Department of Computer Science and Information Engineering, National Chinyi University of Technology (NCUT), where he is an assistant professor. His research interests include scheduling, neural networks, and computer networks.

Shih-Tang Lo, he was born at Hsinchu, Taiwan, R.O.C. in 1965. He received the B. S in computer science from Soochow University in 1987, the M. S. in Information Engineering from Tamkang University in 1989 and the PhD degree in engineering science from National Cheng Kung University of Taiwan R.O.C. in 2007.

Lo is an associate professor of department of information management, Kun-Shan University. His research interests include project scheduling, neural networks, genetic algorithm and ant colony optimization.

Chung-Lun Wu, he was born at Taoyuan, Taiwan, R.O.C. in 1984. He received the B.S. degree in electronic engineering from National Chinyi University of Technology (NCUT) of Taiwan R.O.C. in 2007. Since 2007, he is a graduate student at the Institute of electronic engineering in NCUT. His research interests include scheduling, grid computing, and meta-heuristic algorithm.

Tsung-Hung Lin, he received the B.S. degree in computer science from Tamkang University, Taiwan, Republic of China, in June 1988 and the M.S. and Ph.D. degrees in Computer Science and Information Engineering from the National Chung Cheng University, Taiwan, Republic of China. He joined the faculty of Department of Information Management, Hsing Wu College, Taiwan, Republic of China, as an associate professor in August 2005. Since 2007, he has been an associate professor at the Department of Computer Science and Information Engineering at National Chin-Yi University of Technology, Taiwan, Republic of China. His research interests include wireless communication and mobile computing, the next-generation mobile network and system, wireless personal area network, wireless sensor network, 4G system, scheduling, and optical computing.