MultiCraft

International Journal of Engineering, Science and Technology
Vol. 2, No. 1, 2010, pp. 144-151

INTERNATIONAL JOURNAL OF ENGINEERING, SCIENCE AND TECHNOLOGY

www.ijest-ng.com

An enhanced genetic algorithm with simulated annealing for job-shop scheduling

A. Tamilarasi and T. Anantha kumar*

Department of Computer Applications, Kongu Engineering College, Perundurai, Erode – 638052, Tamil Nadu, INDIA
**Corresponding Author (e-mail: anandmecse@gmail.com, T. Anantha kumar)*

Abstract

The Job-Shop Scheduling Problem (JSSP) is one of the most difficult problems, as it is classified as NP-Hard problem. The main objective of the JSSP is to find a schedule of operations that can minimize the maximum completion time (called makespan) that is the completed time of carrying total operations out in the schedule for *n* jobs and *m* machines. In many cases, the combination of goals and resources exponentially increases the search space, and thus the generation of consistently good scheduling is particularly difficult, because we have a very large combinatorial search space and precedence constraints between operations. Exact methods such as the branch and bound method and dynamic programming take considerable computing time to obtain the optimum solution. In order to overcome this difficulty, it is more sensible to obtain a good solution near the optimal one. Stochastic search techniques such as evolutionary algorithms can be used to find a good solution. In this paper we proposed a new method for solving job-shop scheduling problem using hybrid Genetic Algorithm (GA) with Simulated Annealing (SA). This method introduces a reasonable combination of local search and global search for solving JSSP.

Keywords: Job-shop scheduling, genetic algorithm, simulated annealing, local search, global search

1. Introduction

Several problems in various industrial environments are combinatorial. This is the case for numerous scheduling and planning problems. Generally, it is extremely difficult to solve this type of problems in their general form. Scheduling can be defined as a problem of finding an optimal sequence to execute a finite set of operations satisfying most of the constraints. The problem formulated is extremely difficult to solve, as it comprises of several concurrent goals and several resources which must be allocated to lead to our goals, which are to maximize the utilization of machines and to minimize the time required to complete the entire process being scheduled (Mesghouni *et al*., 2004).

Therefore, the exact methods such as the branch and bound method, dynamic programming and constraint logic programming need a lot of time to find an optimal solution. So, we expect to find an optimal solution using the heuristic methods. Realistically, we are satisfied by obtaining a good solution near the optimal one. New search techniques such as genetic algorithms, simulated annealing or Tabu search are able to solve the job-shop scheduling problem.

Scheduling is broadly defined as the process of assigning a set of tasks to resources over a period of time (Pinedo *et al*., 1999). Effective scheduling plays a very important role in today's competitive manufacturing world. Performance criteria such as machine utilization, manufacturing lead times, inventory costs, meeting due dates, customer satisfaction, and quality of products are all dependent on how efficiently the jobs are scheduled in the system. Hence, it becomes increasingly important to develop effective scheduling approaches that help in achieving the desired objectives.

Several types of manufacturing shop configurations exist in real world. Based on the method of meeting customer's requirements they are classified as either open or closed shops. In an open shop the products are built to order where as in a closed shop the demand is met with existing inventory. Based on the complexity of the process, the shops are classified as single-stage, single-machine, parallel machine, multi-stage flow shop and multi-stage job shop (Lawler *et al.* 1993). The single-stage shop configurations require only one operation to be performed on the machines. In multi-stage flow shops, several tasks are performed

for each job and there exists a common route for every job. In multi-stage job shops, an option of selecting alternative resource sets and routes for the given jobs is provided. Hence the job shop allows flexibility in producing a variety of parts.

The processing complexity increases as we move from single stage shops to job shops. Various methods have been developed to solve the different types of scheduling problems in these different shop configurations for the different objectives. These range from conventional methods such as mathematical programming and priority rules to meta-heuristic and artificial intelligence-based methods (Holland, 1992).

Job shop scheduling is one of the widely studied and most complex combinatorial optimization problems. A vast amount of research has been performed in this particular area to effectively schedule jobs for various objectives. A large number of small to medium companies still operate as job shops (Lin *et al*. 2009). Despite the extensive research carried out it appeared that many such companies continue to experience difficulties with their specific job shop scheduling problems. Therefore developing effective scheduling methods that can provide good schedules with less computational time is still a requirement. Most of the real world manufacturing companies aim at successfully meeting the customer needs while improving the performance efficiency.

Sivanandam *et al*. (2008) gave an idea to use the genetic algorithm (GA) to solve JSSP in an efficient manner. This book presents a basic knowledge and use of various genetic operators in our real world problems. Lin *et al*. (2009) proposed a new hybrid swarm intelligence algorithm consists of particle swarm optimization, simulated annealing technique and multi-type individual enhancement scheme is presented to solve the job-shop scheduling problem. And also it described about the basics of job-shop scheduling problem.

Huiyuan *et al.* (2009) established a dual resource (machines and moulds) constrained job shop scheduling problem model, according to the actual factors of mass injection molding processing job shop scheduling. A heuristic active algorithm combined with priority rules is employed to give the solution. Bozejko *et al*. (2009) developed a Job Shop Scheduling Problem using Parallel Simulated Annealing Technique.

Juang (2004) proposed a hybrid of genetic algorithm and particle swarm optimization for recurrent network design. In this paper the author described how the hybridization of GA with PSO overcomes each other's disadvantages. This paper gives an idea to hybrid the genetic algorithm with simulated annealing to solve the JSSP.

Ge *et al.* (2007) presented a new hybrid algorithm based on particle swarm optimization and simulated annealing for job shop scheduling problem. Beasley (1990) was developed an OR-Library and it is a collection of test data sets for a variety of Operations Research (OR) problems. The benchmark problems are taken from this OR-Library. Several researchers have addressed the importance of solving Job-shop Scheduling Problems, which will help in solve real world problems in industries and to schedule their jobs perfectly.

In the presented work we hybrid the genetic algorithm with simulated annealing to solve the job-shop scheduling problems. GA has a strong ability finding the most optimistic result. SA has a strong ability of finding the local optimistic result. And it can avoid the problem of local minimum. Combining GA and SA, learning from other's strong points to offset one's weaknesses each other, this is the basic idea of our proposed algorithm.

GA is a computational abstraction of biological evolution that can be used to solve optimization problem. In its simplest form, a GA is an iterative process applying a series of genetic operators such as selection, crossover and mutation to a population of elements. These elements, called chromosomes or individuals represent possible solutions to the problem (Watanabe *et al*. 2005).

Simulated annealing (SA) is a kind of global optimization technique based on annealing of metal. It can find the global minimum using stochastic searching technology from the means of probability. Simulated annealing algorithm has a strong ability to find the local optimistic result and it can avoid the problem of local minimum (Triki *et al*., 2005).

2. Job-shop scheduling problem

Job-Shop Scheduling Problem (JSSP) is among the hardest combinational problems and it has been the subject of a significant amount of literature in the Operations Research areas. The JSSP consists of n jobs and m machines. Each job must go through m machines to complete its work. We consider one job consists of m operations. Each operation uses one of m machines to complete one job's work for a fixed time interval. Once one operation is processed on a given machine, it cannot be interrupted before it finishes the job's work. The sequence of operations of one job should be predefined and may be different for any job. In general, one job being processed on one machine is considered as one operation noted as O_{ji} (means jth job being processed on ith machine, $1 \le j \le n, 1 \le i \le m$) (Garey *et al*. 1976 and Lawler *et al*. 1993). Each machine can process only one operation during the time interval. The objective of JSSP is to find an appropriate operation permutation for all jobs that can minimize the makespan C_{max}, i.e., the maximum completion time of the final operation in the schedule of $n \times m$ operations.

For an $n \times m$ JSSP, the problem can be modeled by a set of m machines, denoted by $M=\{1,2,\ldots,m\}$, to process a set of $n \times m$ operations, denoted by $O = \{0,1,2,\ldots,(n \times m)-1\}$ (Lin *et al*. 2009). The notations are as follows:

n : number of jobs

m : number of operations for one job

O_i : completed time of operation i (i=0,1,2,……($n \times m$)-1)

t_i : processing time of operation i on a given machine

C_{max} : makespan

The JSSP has n jobs to be processed on m machines and we assume the following:

1. No machine may process more than one job at a time;
2. No job may be processed by more than one machine at a time;
3. The sequence of machines which a job visit is completely specified and has a linear precedence structure;
4. Processing times are known;
5. All jobs must be processed in each machine only one time.

3. Genetic algorithm

The genetic algorithm (GA) is a search technique based on the mechanics of natural genetics and survival of the fittest (Goldberg, 1989). GA simulates the biological processes that allow the consecutive generations in a population to adapt to their environment. The adaptation process is mainly applied through genetic inheritance from parents to children and through survival of the fittest. The genetic algorithm object determines which individuals should survive, which should reproduce, and which should die. It also records statistics and decides how long the evolution should continue. Figure 1 illustrates the simple genetic algorithm (Premalatha *et al*., 2009).

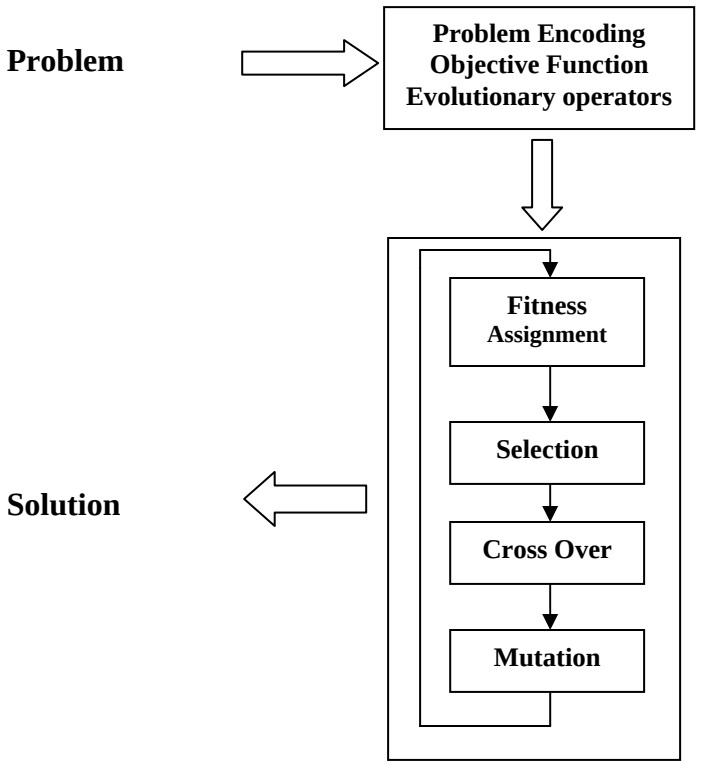

Figure1: Simple genetic algorithm (Premalatha *et al*., 2009)

To successfully apply a GA to solve a problem one needs to determine the following (Premalatha *et al*., 2009):

1. The representation of possible solutions, or the chromosomal encoding.
2. The fitness function which accurately represents the value of the solution.
3. Genetic operators (selection, crossover and Mutation) have to employ and the parameter values (population size, probability of applying operators, etc.), that are suitable.

4. Simulated annealing

Annealing is an operation in metal processing (Kirkpatrick *et.al.*, 1983). Metal is heated up very strongly and then cooled slowly to get a very pure crystal structure with a minimum of energy so that the number of fractures and irregularities becomes minimal. first the high temperature accelerates the movement of the particles. During the cooling time they can find an optimal place within the crystal structure. While the temperature is lowered the particles subsequently lose the energy they were supplied with in the first stage of the process. Because of a thermodynamic, temperature-dependent random component some of them can reach a higher energy level regarding the level they were on before. These local energy fluctuations allow particles to leave local minima and reach a state of lower energy.

Simulated annealing is a relatively straight forward algorithm through which includes metropolis Monte Carlo method .the metropolis Monte Carlo algorithm is well suited for simulated annealing, since only energetically feasible states will be sampled at any given temperature. The simulated annealing algorithm is therefore a metropolis Monte Carlo simulation that starts at a high temperature. The temperature is slowly reduced so that the search space becomes smaller for the metropolis simulation, and when the temperature is low enough the system will hopefully have settled into the most favorable state.

Simulated Annealing can also be used to search for the optimum solution of the problems by properly determining the initial (high) and final (low) effective temperatures which are used in place of kT (where k is a Boltzmann's constant) in the acceptance checking, and deciding what constitutes a Monte Carlo step (Wang *et al*., 2004).

The initial and final effective temperatures for a given problem can be determined from the acceptance probability. In general, if the initial Monte Carlo simulation allows an energy (E) increase of dE_i with a probability of P_i, the initial effective temperature is $kT_i = -dE_i/ln(P_i)$. If at the final temperature an increase in the cost of 10 should only be accepted with a probability of 0.05 (5%), the final effective temperature is $kT_f = -10/\ln(0.05) = 3.338$.

Algorithm

```
Start with the system in a known configuration, at known energy E
T=temperature =hot; frozen=false;
While (! frozen) {
    repeat {
                Perturb system slightly (e.g., moves a particle)
                Compute E, change in energy due to perturbation
                If(ΔE < 0 )
                   Then accept this perturbation, this is the new system config
                Else accept maybe, with probability =   exp(-ΔE/KT)
            } until (the system is in thermal equilibrium at this T)
            If(ΔE still decreasing over the last few temperatures)
          Then T=0.9T//cool the temperature; do more perturbations
           Else frozen=true
}
return (final configuration as low-energy solution)
```

5. Proposed system

5.1 Problem representation

Consider the following 3x3 Job-Shop Scheduling Problem. The Operation times and machine ordering is shown in the following tables.

Table1. 3x3 JSSP

Jobs	**Operations**		
J1	*O1*	*O2*	*O3*
J2	*O4*	*O5*	*O6*
J3	*O7*	*O8*	*O9*

Table2. Machine Allocation

Machine Allocation		
M1	**M2**	**M3**
O1	*O3*	*O2*
O5	*O4*	*O6*
O8	*O9*	*O7*

Table 3. Operation Times

Jobs	**Operation Times**		
J1 (*O1,O2,O3*)	2	3	4
J2 (*O4,O5,O6*)	1	5	2
J3 (*O7,O8,O9*)	4	6	4

The feasible Schedule for the above JSSP is 1,4,7,5,2,8,3,6,9. That is the operations are performed in the machines as of in the above schedule gives the makespan as 17, which is feasible result for the give JSSP. The Gantt chart of the schedule 1, 4, 7, 5, 2, 8, 3, 6, 9 is given in Fig 2. In the proposed method we have taken the schedules as the chromosomes to solve the JSSP.

Time Slot

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	**17**
M1	*O1/2*		*O5/5*					*O8/6*									
M2	*O4/1*							*O3/4*						*O9/4*			
M3	*O7/4*				*O2/3*			*O6/2*									

Figure.2. Gantt chart for Feasible Schedule

The searching space is created in $n \times m$ dimensions space for n jobs on m machines Job-Shop Scheduling Problem. The problem solution is represented as a chromosome in Genetic Algorithm. The positions of a chromosome consists of $n \times m$ dimensions and is represented with $n \times m$ real numbers. In order to simulate an operation permutation sequence of JSSP, the $n \times m$ real numbers are transformed into an integer series from 1 to $n \times m$. Each integer number represents one operation index ($O_{ji}, 1 \leq j \leq n, 1 \leq i \leq m$) according to its ordering in all $n \times m$ real numbers. Fig. 3 illustrates an example of a representation of Chromosome for a 3×2 JSSP.

Chromosome	10.2	2.5	7.5	6.7	1.3	5.3
Integer Series (π_k)	6	2	5	4	1	3
(π_k mod 3)+1	1	3	3	2	2	1
Operation Sequence	O_{11}	O_{31}	O_{32}	O_{21}	O_{22}	O_{12}

Figure 3. Problem representation

5.2 Objective function

The main objective of the JSSP is to find a schedule of operations that can minimize the maximum completion time (called makespan), that is the completed time of carrying total operations out in the schedule for n jobs and m machines. The Objective or fitness function takes the input as the number of jobs, number of Operations, Chromosome, Operation time Sequence and Machine Sequence of the corresponding operation. Each chromosome genes are assigned by an integer number (π_k) by ranking the genes (real numbers) in ascending order. And then perform (π_k mod No. of Jobs) +1 operation to each π_k to get the corresponding operation sequence of a chromosome. The fitness function produces the output as a makespan value for the corresponding operation sequence.

5.3 Algorithm

The proposed algorithm for solving JSSP is as follows

1. Initialize temperature T to a particular value.
2. Initialize the N number of chromosomes by generating $n \times m$ real numbers for each chromosome.
3. Find the Operation time sequence and Machine sequence for N chromosomes.
4. Find the makespan value for each and every chromosome using the objective function and also find the minimum makespan value (*best*) among N makespan values.
5. Select $N/2$ chromosomes using the Roulette - Wheel selection from N chromosomes
6. Crossover the selected chromosomes with the probability as 0.9 and Mutate the new chromosomes with the probability as 0.3 to get new chromosomes.
7. Find the makespan values for newly generated chromosomes using the objective function.
8. Choose the N best chromosomes which have the minimum makespan values from the newly generated and also from old chromosomes.
9. Find the minimum makespan value (*best*) among the N best chromosomes.
10. If *best* chromosome is not changed over a period of time
    a. Find a new chromosome using temperature.
11. Accept the new chromosome as *best* with probability as exp-($\Delta E/T$), even though current position is worse. Here ΔE is the difference between current best chromosome's makespan and new chromosome's makespan value.
12. Reduce T.
13. Terminate if the maximum number of iterations is reached or optimal value is obtained.

14. Go to step 3.

6. Experimental results

In this paper, we have used 21 instances that are taken from the OR-Library (Beasley, 1990) as benchmarks to test our new proposed algorithm. In this algorithm we have taken 200 chromosomes as the initial population and temperature *T* as 5000.

We have used the Intel Pentium Core 2 Duo 3GHz Processor and 2 GB RAM configuration system with Windows XP as the platform to run this algorithm and achieved the following results. The CPU time is the time to find the optimal schedule for the benchmark JSSPs from using the above configuration machine. The proposed work gives the optimal solution for most of the benchmark problems in considerable amount of CPU time.

Goncalves *et al*. (2005) gives optimal Solution for most of the benchmark problems. But our proposed algorithm gives the optimal solution within a minimum considerable amount of time. We can't compare the computation times of Goncalves et al. (2005) with our proposed work, because the system configuration is not unique. Goncalves *et al*. (2005) gives 907 as a makespan for the Instance LA 20, but it is not an optimal solution. But our proposed algorithm gives the optimal solution (902).

Table.4. Experimental results

S No	Instance Name	Instance Size ($n \times m$)	Best Known Value	Obtained Values from the JSSP using Simple Genetic Algorithm	Goncalves *et al*. (2005)	Obtained Values from the Proposed Algorithm	CPU Time (in Milliseconds) for the proposed Algorithm	Relative Error
1.	FT06	6 × 6	55	**55**	55	**55**	72789	0.00
2.	LA01	10 × 5	666	686	666	**666**	64887	0.00
3.	LA02	10 × 5	655	739	655	**655**	1689748	0.00
4.	LA03	10 × 5	597	670	597	**597**	12564254	0.00
5.	LA04	10 × 5	590	634	590	**590**	10458265	0.00
6.	LA05	10 × 5	593	**593**	593	**593**	17344	0.00
7.	LA06	15 × 5	926	956	926	**926**	92072	0.00
8.	LA07	15 × 5	890	994	890	**890**	3939717	0.00
9.	LA08	15 × 5	863	906	863	**863**	102010	0.00
10.	LA09	15 × 5	951	956	951	**951**	39050	0.00
11.	LA10	15 × 5	958	**958**	958	**958**	30947	0.00
12.	LA11	20 × 5	1222	1267	1222	**1222**	50394	0.00
13.	LA12	20 × 5	1039	1071	1039	**1039**	155259	0.00
14.	LA13	20 × 5	1150	1188	1150	**1150**	173347	0.00
15.	LA14	20 × 5	1292	**1292**	1292	**1292**	17087	0.00
16.	LA15	20 × 5	1207	1383	1207	**1207**	12435098	0.00
17.	LA16	10 × 10	945	1080	945	**945**	13008994	0.00
18.	LA17	10 × 10	784	906	784	**784**	13072238	0.00
19.	LA18	10 × 10	848	976	848	**848**	12407916	0.00
20.	LA19	10 × 10	842	1021	842	**842**	13038003	0.00
21.	LA20	10 × 10	902	1072	*907*	**902**	18945246	0.00

7. Conclusion and future work

In this paper, the hybridization of genetic algorithm with simulated annealing is used to solve the NP-hard JSSP. This algorithm adopts the real space as the search space and the chromosome represents the permutation of all operations of all jobs. Combining

genetic algorithm and simulated annealing can narrow the field of search and speed up the rate of convergence continually in the optimizing process. It has higher searching efficiency. It can also escape from the local minimums.

In the proposed algorithm, the problem representation is simple and easy when compared with other heuristic methods. Also this algorithm reduces the computational complexity. The experiment is conducted for population of size 200 chromosomes, and the optimal solution (Best Known Solution) in the search space is obtained for the benchmark problems with the above mentioned system configuration. The proposed algorithm gives optimal result for most of the benchmark problems when compared with JSSP using simple genetic algorithm which is shown in Table 4.

In the proposed algorithm, it is assumed that each operation of a job is to be completed on a single machine. The future works may be

1. To find the methods to solve multi machine environment and the dynamic Job-Shop Scheduling Problem.
2. Very high configuration machines are used to obtain the optimal solutions in the minimum time period.
3. Grid based algorithms will be found to find the makespan of the Job-Shop Scheduling Problem. It will reduce the computational complexity and increase the performance and quality of the scheduling.
4. Other hybridization heuristic methods are to be used to solve the Job-Shop Scheduling Problem.

Acknowledgment

This project work is supported by University Grants Commission; New Delhi under the Major Research Project Grant (F.No. 33-65/2007(SR)) entitled “Job Shop Scheduling using Heuristic Methods”.

References

Beasley J.E. 1990. OR-library: Distributing test problems by electronic mail, *Journal of the Operational Research Society,* Vol.41, No.11, pp.1069–1072.

Bozejko W., Pempera J. and Smuntnicki C. 2009. Parallel simulated annealing for the job shop scheduling problem, *Lecture notes in computer science*, *Proceedings of the 9th International Conference on Computational Science*, Vol.5544, pp. 631-640.

Chan Felix T.S., Wong T.C. and Chan L.Y. 2009. The application of genetic algorithms to lot streaming in a job-shop scheduling problem, *International Journal of Production Research*, Vol.47, No.12, pp.3387-3412.

Eberhart R.C. and Shi Y. 1998. Comparison between genetic algorithms and particle swarm optimization, *Proceedings of Congress on Evolutionary Computation*, San Diego, Berlin, pp.105-110.

Garey J.E., Johnson D.S. and Sethi R. 1976. The complexity of flowshop and jobshop scheduling, *Mathematics of Operations Research*, Vol.1, No. 2, pp. 117–129.

Ge H.W., Du W. and Qian F. 2007. A hybrid algorithm based on particle swarm optimization and simulated annealing for job shop scheduling, *Proceedings of the Third International Conference on Natural Computation*, Vol. 3, pp. 715–719.

Goldberg D.E., 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Publishing Corporation, Boston, MA, USA.

Goncalves J.F., Mendes J.J.D.M. and Resende M.G.C. 2005. A hybrid genetic algorithm for the jobshop scheduling problem, *European Journal of Operational Research* , Vol.167, pp. 77–95

Gray P., Hart W.E., Painton L., Phillips C., Trahan M. and Wagner J. 1997. A survey of global optimization methods, *Technical Report*, *Sandia National Laboratory,* Albuquerque, NM 87185.

Holland J.H. 1992. A*daptation in Natural and Artificial Systems*, MIT Press, Cambridge.

Huiyuan R., Lili J., Xiaoying X. and Muzhi L. 2009. Heuristic optimization for dual-resource constrained job shop scheduling, *International Asia Conference on Informatics in Control, Automation and Robotics*, pp.485-488.

Juang C.F. 2004. A hybrid of genetic algorithm and particle swarm optimization for recurrent network design, *IEEE Transactions on Evolutionary Computation*, Vol.34, No. 2, pp.997-1006.

Kennedy J. and Eberhart R.C. 2001. *Swarm intelligence,* Morgan Kaufmann Publishers, Inc., San Francisco, CA

Kirkpatrick S., Gelatt C.D. and Vecci M.P. 1983. Optimization by simulated annealing, *Science, New Series*, Vol. 220, No. 4598, pp. 671-680.

Lawler E.L., Lenstra J.K., Rinnooy Kan A.H.G. and Shmoys D.B. 1993. Sequencing and scheduling: Algorithms and complexity, *Handbooks in operations research and management science, Logistics of Production and Inventory,* Vol. 4, pp. 445-552.

Lian Z., Gu X. and Jiao B. 2006. A dual similar particle swarm optimization algorithm for job-shop scheduling with penalty, *Proceedings of the Sixth World Congress on Intelligent Control and Automation,* Vol. 2, pp. 7312–7316.

Lin T.L., Horng S.J., Kao T.W., Chen Y.H., Run R.S., Chen R.J., Lai J.L. and Kuo I.H. 2009. An efficient job-shop scheduling algorithm based on particle swarm optimization, *Expert Systems with Applications*, Vol.37, No. 3, pp. 2629-2636.

Mesghouni K., Hammadi S. and Borne P. 2004. Evolutionary algorithms for Job-Shop Scheduling, *International Journal of Applied Mathematics and Computer Science,* Vol. 14, No. 1, pp. 91-103.

Pinedo M. and Singer M. 1999. A shifting bottleneck heuristic for minimizing the total weighted tardiness in a job shop, *Naval Research Logistics*, Vol. 46, No. 1, pp. 1–17.

Premalatha K. and Natarajan A.M. 2009. Hybrid PSO and GA for global maximization. *International Journal of Open Problems in Computer Science and Mathematics*, Vol. 2, No. 4. pp. 597-608.

Sivanandam S.N. and Deepa S.N. 2008. *Introduction to Genetic Algorithms,* Springer, Berlin, New York.

Triki E., Collette Y. and Siarry P. 2005. A theoretical study on the behavior of simulated annealing leading to a new cooling schedule, *European Journal of Operational Research*,Vol.166, No.1, pp.77-92.

Udomsakdigool A. and Kachitvichyanukul V. 2008. Multiple colony ant algorithm for job-shop scheduling problem, *International Journal of Production Research* , Vol. 46, No. 15, pp. 4155–4175.

Wang X. and Li J. 2004. Hybrid particle swarm optimization with simulated annealing, *Proceedings of Third International Conference on Machine Learning and Cybernetics*, Vol.4, pp. 2402-2405.

Watanabe M., Ida M. and Gen M. 2005. A genetic algorithm with modified crossover operator and search area adaptation for the job-shop scheduling problem, *Computers and Industrial Engineering*, Vol.48, No. 4, pp. 743–752.

Zhou Q., Cui X., Wang Z. and Yang B. 2009. A hybrid optimization algorithm for job-shop scheduling problem, *Proceedings of first ACM/SIGEVO Summit on Genetic and Evolutionary Computation*, pp.757-764.

Biographical notes

Dr.A.Tamilarasi is a Professor and Head in the Department of Computer Applications, Kongu Engineering College, Perundurai, Erode, Tamil Nadu, India. She obtained Ph.D. in 1994 from the University of Madras, Chennai. She was awarded JRF by University Grants Commission, New Delhi in the year 1986. She has the total teaching experience of 15 years. She has published about 35 papers and 7 books. Her areas of interest are semi group theory, fuzzy logic and its applications in engineering fields. She is an approved guide of Anna University, Chennai and Anna University, Coimbatore and presently she is guiding 10 Ph.D. scholars.

Mr.T. Anantha kumar is a Project Fellow in the Department of Computer Applications, Kongu Engineering College, Perundurai, Erode, Tamil Nadu, India for the project entitled "Job Shop Scheduling using Heuristic Methods" supported by University Grants Commission, New Delhi. He completed M.E., Computer Science and Engineering in 2009 at Kongu Engineering College, Perundurai and B.E., Computer Science and Engineering in 2006 at Erode Sengunthar Engineering College, Thudupathi. His areas of interest are data mining, optimization techniques and scheduling.

Received January 2010
Accepted January 2010
Final acceptance in revised form February 2010