ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology
Volume 1, Issue 5, July 2012

Comparison of Memetic Algorithm and PSO in Optimizing Multi Job Shop Scheduling

M. Nandhini[#1], S.Kanmani[*2]

[#]*Research and Development Centre,*
Bharathiar University,Coimbatore-46
Tamil Nadu, India -14.
[1]mnandhini.pu@gmail.com

[*] *Professor, Department of Information Technology,*
Pondicherry Engineering College,
Puducherry, India -14.
[2]kanmani@pec.edu

Abstract—**In this paper, proposes a memetic algorithm to optimize multi objective multi job shop scheduling problems. It consists of customized genetic algorithm and local search of steepest ascent hill climbing algorithm. In genetic algorithm, the customization is done on genetic operators so that new selection, crossover and mutation operators have been proposed. The experimental study has done on benchmark multi job shop scheduling problem and its versatility is proved by comparing its performance with the results of simple genetic algorithm and particle swarm optimization .**

Index Terms— ***Combinatorial optimization, genetic algorithm, particle swarm optimization, local search, multi job shop scheduling, optimal solution.***

I. INTRODUCTION

Scheduling deals with the timing and coordination of activities which are competing for common resources. MJSSP is one of the most eminent machine scheduling problems in manufacturing systems, operation management, and optimization technology.

The goal of MJSSP is to allocate machines to complete jobs over time, subject to the constraint that each machine can handle at most one job at a time. The complexity of MJSSP increases with its number of constraints and size of search space.

Scheduling problems like timetabling, job scheduling etc., are the process of generating the schedule with multiple objectives, follows *multi objective combinatorial optimization* or *MOCO* problem

Since exact approaches are inadequate and requires too much computation time on large real world scheduling problems, heuristics and meta heuristics are commonly used in practice. Meta-heuristics usually combines some heuristic approaches and direct them towards solutions of better quality than those found by local search heuristics.

Meta-heuristics, using mainly two principles: local search and population search. *Local search* is the name for an approach in which a solution is repeatedly replaced by another solution that belongs to a well-defined *neighborhood* of the first solution. In *Population search methods*, a diversified exploitation of the population of solution is performed in each generation, resulting new generation with better optimal solutions.

Genetic algorithms(GA), Simulated annealing(SA), Tabu search(TS), Artificial neural networks(ANN), Greedy randomized adaptive search procedures(GRASP), Threshold algorithms, Scatter, Variable neighborhood search(VNS), Cooperative search systems are some of the meta heuristics applied for optimization of combinatorial problem over 20 to 30 years (Schaerf S A, 1999). GA helps in diversifying the search in order to get global optima and local search techniques help in intensifying the search. This motivated us to form a hybrid model called as memetic algorithm (Olivia and Ben(2004) by combining GA with local search to get the exertion of both.

In the past years, evolution of the population in GA took place with the vast introduction of GA operators particularly on selection, crossover and mutation [3-7] and variety of representation of chromosomes [8-11] on various applications like traveling salesman problem, multi job shop scheduling, timetabling etc.

Although the existing algorithms in the literature can increase the convergence rate and search capability of the simple genetic algorithm to some extent, the crossover and mutation operators used in these algorithms have not sufficiently made to use the characteristics of the problem structure.

Most of genetic operators only change the form of encoding and are difficult to integrate the merit of the parent individuals. Sufficient use of information in the problem structure and the inspiration of soft constraints satisfaction which decide the optimality factor, motivated us to propose domain specific reproduction operators; Combinatorial Partially Matched crossover(CPMX), mutation with adaptive strategy and grade selection operator to generate the optimal solution with minimum execution time.

To balance the exploration and exploitation abilities, local search (SAHC) is applied simultaneously with customized GA architecture. This hybrid combination has been tried over multi job shop scheduling (MJSSP) and its versatility was proved by comparing its performance with simple GA and PSO.

The organization of the rest of the paper is as follows. The discussion on related works is given in Section 2. In Section

3, the problem definition of MJSSP with its constraints and chromosome representation are mentioned. In Section 4, proposed methodology with design of GA operators followed by description of particle swarm optimization (PSO) in Section 5. The data set and discussion on experimental results and comparison with other algorithms are done in Section 6 and 7. Finally, Section 8 concludes this work.

II. Literature Review

Over two decades, a number of investigations are carried out in the evolution strategies. The striking point of using GA refers to the way how to select a combination of appropriate GA operators in selection, crossover , mutation probability and so forth. Some of the related works carried out using GA with some local search for solving MJSSP are given below.

Ping-Teng Chang and Yu-Ting Lo (2001) modelled the multiple objective functions containing both multiple quantitative (time and production) and multiple qualitative objectives in their integrated approach to model the JSSP, along with a GA/TS mixture solution approach.

Kamrul Hasan S. M. et. al., (2007) proposed a hybrid genetic algorithm (HGA) that includes a heuristic job ordering with a GA.

Christian Beirwirth (1995) proposed a new crossover operator preserving the initial scheme structure as Generalization of OX (GOX) to solve MJSSP. The new representation of permutation with repetition and GOX support the cooperative aspect of genetic search for scheduling problems.

Adibi M A et al.(2010) developed dynamic job shop scheduling that consists of variable neighborhood search (VNS), a trained artificial neural network (ANN). ANN updates parameters of VNS at any rescheduling.

Takeshi Yamada and Ryohei Nakano (1996) presented multi-step crossover operators fusion (MSXF) with a neighborhood search algorithm for JSS. MSXF searches for a good solution in the problem space by concentrating its attention on the area between the parents. GA/MSXF could find near-optimal solutions.

Rui Zhang (2011) presented a PSO algorithm based on Local Perturbations for the JSSP. In his work, a local search procedure based on processing time perturbations is designed and embedded into the framework of PSO for MSSP model.

In this paper, we attempt to introduce a new customized GA algorithm which has the strength of improving optimality obtained through satisfaction of multi soft constraints where crossover and mutation operators are designed according to the problem structure. To improve the outcome of GA operators, proposed to apply SAHC local search.

Also, the optimization mechanism of the traditional PSO is analyzed and a general optimization model based on swarm intelligence is proposed to solve MJSSP.

It is showed that this customized GA with local search could give better optimal solution than simple GA and PSO. This proposed algorithm is described in the following section.

III. Multi Job Shop Scheduling

The job shop scheduling problem (JSSP) can be described as follows: given n jobs, each composed of m operations that must be processed on m machines. Each operation uses one of the m machines for a fixed duration. Each machine can process at most one operation at a time and once an operation initiates processing on a given machine it must complete processing on that machine without interruption.

The JSSP consists of n jobs and m machines. Each job must go through m machines to complete its work. It is considered that 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. In general, one job being processed on one machine is considered as one operation noted as O_{ji} (means j^{th} job being processed on i^{th} machine, $1 \le j \le n$, $1 \le i \le m$) (Garey *et.al*., 1976 , Lawler *et.al*.,1993). Each machine can process only one operation during the time interval.

The objective of MJSSP is to find an appropriate operation permutation for all jobs that can minimize the makespan i.e., the maximum completion time of the final operation in the schedule of n×m operations with minimum waiting time of jobs and machines.

The problem can be made to understand with its known constraints (mandatory (C_i) & optional(S_i)) / assumptions(A_i) as listed below.

- C_1: No machine should process more than one job at a time
- C_2: No job should be processed by more than one machine at a time
- C_3: The order in which a job visits different machines is predetermined by technological constraints
- C_4**:** Different jobs can run on different machines simultaneously
- C_5: At the moment T, any two operations of the same job cannot be processed at the same time
- A_1: Processing time on each machine is known
- S_1: Idle time of machines may be reduced
- S_2: Waiting time of jobs may be reduced

Mathematical Modeling of Constraints***:***

Entities used in mathematical representation of constraints are given below.

J (j1, j2, j3) :Jobs
M (m1, m2, m3) :Machines
O (01, 02, 03) :Sequence of operations for a job
TS :Processing time of an operation
Cmax :Makespan
M[j][ts] ->m : machine name with job j at time ts
J[m][ts] -> j : job name on machine m at time ts
Mac[j][o][ts] ->m: machine name with operation o of job j at time ts

Capacity Constraints

C1: No machine may process more than one job at a time.

$$\exists m1 \in M, \exists j1, j2 \in J, \exists ts \in TS$$
$$if M[j1][ts] = m1, M[j2][ts] \neq m1$$

C2: No job may be processed by more than one machine at a time.

$$\exists m1, m2 \in M, \exists j \in J, \exists ts \in TS$$
$$if J[m1][ts] = j, J[m2][ts] \neq j$$

C3: The order in which a job visits different machines is predetermined by technological constraints.

$$\forall j \in J,$$
$$\exists m,1, m2..mm \in M, \exists o1, o2,..om \in O,$$
$$j[o1][ts] = m1,\ j[o2][ts] = m2,....$$
$$j[on][ts] = mm \mid o1 < o2 < ..om;$$

C4: Different jobs can run on different machines simultaneously.

$$\exists j1,\ j2 \in J, \exists m1, m2 \in M, ts \in TS,$$
$$if j1 \neq j2 then M[j1][ts] = m1, M[j2][ts] = m2$$

Precedence Constraints

C5: At the moment *T*, any two operations of the same job cannot be processed at the same time.

$$\exists ts \in TS, \exists j \in J, \exists o1, o2 \in j,$$
$$\exists m1, m2 \in M,$$
$$if \text{Mac[j][o1][ts]} = \text{m1 then Mac[j][o2][ts]} \neq \text{m2}$$

Soft Constraints

S1: Idle time of machines may be reduced.

$$\forall m \in M, idleTime(m) = Min\{t\}$$

S2: Waiting time of jobs may be reduced.

$$\forall j \in J, waitTime(j) = Min\{t\}$$

Chromosome Representation

In our work, each state is represented as an array of structures. Each structure consists of job name and its operation as members. In this representation, the chromosome consists of n*m genes. Each job will appear m times exactly. By scanning the chromosome from left to right, the *k*-th occurrence of a job number refers to the *k*-th operation in the technological sequence of this job. This method followed with heuristics which can always gain the feasible scheduling solution.

Job No.	*Op. No.*	*Job No.*	*Op. No.*	...	...	*Job No.*	*Op. No.*

Where,

Job No. i: 1..n; Operation No. j: 1..m;

IV. PROPOSED MEMETIC ALGORITHM

To design a robust steady state GA, domain specific operators for selection, crossover and mutation are introduced. The GA operators probability also decided experimentally.

A. Design of proposed operators:

The right choices of GA operators help in getting the optimal solution. This motivates us to propose domain specific GA operators like grade selection , crossover (Combinatorial Partially Matched (CPMX)) and mutation with adaptive strategy. To analyze the performance of these customized operators, they have been combined to form customized GA architecture. To have faster convergence and to improve the quality of individuals, Steepest Ascent Hill Climbing(SAHC) local search algorithm is combined with this combination and resulting to a new memetic algorithm and the proposed methodology is given in Fig.3. The descriptions about the proposed operators are as follows.

1) Objective Function:

The fitness value (objective function) decides the strength of the chromosome. In MOCO problem, the satisfaction of soft constraints decides the optimality factor. This aspires us to include the satisfaction level of soft constraints S_1 and S_2 in the proposed objective functions to get wealthy chromosomes and is proposed as follows.

Minimize Z = Min (Makespan+ Jobs Waiting Time+ Machines Waiting Time)

Where, *Job No. i : 1..m ; Machine No. j: 1..n*

2) Selection: Selection is the process of choosing parents from the generated population to undergo genetic operations like mutation or crossover.

a) ***Grade:*** The problem of local search algorithms in finding optimal solution is terminating in local optimal solution. The global optimality is made possible by introducing variety in the population. In GA, the feature of individuals is exploited by the recombination operators. This trails the individuals to undergo problem specific recombination operation. It is the striking point of proposing this selection operator. This takes chromosomes randomly from combination of groups decided based on fitness value such as worst, worst; worst, better; worst, best; better, better; better, best; and best, best and mating those chromosomes result in salient features. Hence, random with guided selection is done.

b) Procedure of *Grade* Selection: Each group is formed with the chromosomes of similar nature. To form the groups, the standard deviation for the individuals in the mating pool is found using the formula, $$s_n = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2}.$$

Where

x_i - Cost of the i^{th} individual

N - Number of individuals in the mating pool

To select the individuals for mating, the first parent is randomly selected from one group and the other one is selected randomly from any other group other than the group containing first parent. After selecting both the parents, they are removed from the mating pool.

3) Crossover:

To encourage the exploration of search space crossover is applied on individuals. The proposed domain specific operators are,

CPMX: It revises partially matched crossover (PMX) (Sivaanandham S N and Deepa S N , 2008) to exchange more genes by finding the permutation of partially matched set. To avoid the uncertainty in the resulted individuals due to the context sensitivity of these problems, this guided crossing is proposed. The procedure of this crossover operator is given in Fig.1.

Procedure for CPMX.

Find the partial matching set(PMS) in two chromosomes / Length of the PMS to be decided earlier

Find the possible combinations of PMS

Repeat

Do the exchange in genes between a combination of PMS of two chromosome

Count the number of exchanges done on genes

Until All combination of PMS of two chromosomes are over

Take the resultant chromosomes formed with maximum exchanges of genes in PMS
Check the feasibility of resultant offspring
If offspring solution is infeasible
Repair the solutions
else
Input the chromosomes to SAHC process
End if

Fig. 1. Procedure for Proposed CPMX.

4) Mutation:

In general the change in genes is done on random basis. From the past researches, it is inferred that there is no guarantee of improvement in the solution by these random changes. To achieve this, removing the violation of soft constraints is attempted through mutation. This is achieved by tuning the genes and hence proposed and named as gene tuning. To identify the soft constraint to be satisfied, the following strategy on mutation has been proposed and its procedure is given in Fig.2.

- Mutation with adaptive strategy : Selecting the soft constraint resulting to minimum fitness score by gene tuning

```
Repeat
    Repeat
        On each chromosome [ Rank Selection]
            Repeat
                Apply mutation of first soft constraints
                If mutated chromosome is feasible, find fitness
                Else
                Repair and find fitness.
            Until ( all soft constraints )
        Select and store mutated chromosome with minimum fitness
    Until (mutation rate is reached)
    Select an offspring with minimum fitness from mutated chromosomes
    If (offspring has better fitness than its parent)
        Replace parent with the offspring
    Else
        Apply SAHC local search to improve its fitness and replace with parent.
Until(Termination criteria is reached)
```

Fig.2 Procedure for Adaptive strategy Mutation

B. Local Search (Steepest Ascent Hill Climbing)

We use a very effective local search consisting of a stochastic process in three phases:

- The first phase to improve an infeasible solution (timetable/jobs schedule) so that it becomes feasible (Repair Function)
- The second phase to increase the quality of a feasible solution by reducing the number of soft constraint violated (Mutation) and
- The third phase is to improve the quality of the feasible solution by interchanging the genes(SAHC)

The above discussed GA operators and SAHC are combined to form a memetic approach with the probabilities specified in Table.1. The parameters probabilities which gave better performance from various experiments are taken.

Table.1 GA parameters

Parameter	Probability
Population	1000
Elitism	.01
Crossover	.08
Mutation	.03

V. PARTICLE SWARM OPTIMIZATION

Particle swarm optimization (PSO) is one of the metaheuristics algorithms. The concept of particle swarm has become very popular these days as an efficient search and optimization technique. PSO does not require any gradient information of the function to be optimized, but uses only primitive mathematical operators and is conceptually very simple.

In PSO[16], there is a population called a swarm, and every individual in the swarm is called a particle which represents a solution to the problem. All of particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct the flying of the particles. The particle flies in the D-dimensional problem space with a velocity which is dynamically adjusted according to the flying experiences of its own and its colleagues. The basic Principle of Particle swarm move towards the best position in search space, remembering each particle's best known position (pbest) and global (swarm's) best known position (gbest).

1) General Particle Swarm Optimization Algorithm:

Algorithm begins with a set of solutions as a Initial Population called swarm of particles and searches for optima by updating generations. For each particle in the swarm, velocity and position is calculated with their constraints to obtain feasible solutions. In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. This value is called pbest. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called gbest. The new generation of solutions thus formed is most likely to be better than the previous ones. This process of forming new generation of solutions is continued until a best fitness value is obtained.

2) Particle Swarm Optimization Algorithm for MJSSP:

The Multi Job Shop Scheduling Problem is implemented using Particle Swarm Optimization Algorithm by the following steps:

i) Generate random population of N particles using Random Constructive heuristic process
ii) Evaluate makespan for each particle in the swarm
iii) For each particle in the swarm
a. Calculate pbest and gbest
b. Update particle velocity and position
iv) Use the new generated swarm for the further run of the algorithm
v) Repeat the steps (ii) to (iv) until stop criterion is met

a) pbest Evaluation

The particle's best known position achieved so far in their respective iteration is said to be pbest. It is obtained by comparing the previous pbest value with the fitness of the current particle. If it is better the pbest is changed with the new better fitness.

b) gbest Evaluation

The global best known position in swarm achieved so far in each iteration is said to be gbest. It is obtained by comparing the previous gbest value with the minimum fitness obtained in current iteration. If it is better the gbest is changed with the new better fitness.

c) Velocity Updation

In each iteration, after finding the two best values, the particle updates the velocity with the following equation (a).

*v[] = v[] + c1 * rand() * (pbest[] - present[]) + c2 * rand() * (gbest[] - present[]) --- (a)*

The equation (a) helps the particle move in same direction to achieve optimum. The velocity equation has three parts. The first part represents the inertia of previous velocity, forces the particle to move in same direction. The second part is the "cognition" part, which represents the private thinking by itself and forces the particle to go back to the previous best position. The third part is the "social" part, which represents the cooperation among the particles, forces the particle to move to the best previous position of its neighbors. The maximum velocity v_{max} is said to number of jobs. If the velocity exceeds $v_{max,}$ then it is assigned to $v_{max.}$

d) Position Updation

The position vector represents jobs' arrangement on all machines. The maximum position p_{max} is said to nxm (no. of jobs * no. of operations). If the position exceeds $p_{max,}$ then it is assigned to $p_{max.}$ The results of a particle's position may have a meaningless job number as a real value such as 3.125. So, the real optimum values are round off to its nearest integer number. The computation results of the following equation (b),

present[] = persent[] + v[] ---------- (b)

will generate repetitive code (job number), i.e. one job is processed on the same machine repeatedly. It violates the constraint conditions in MJSSP and produce banned solutions. Banned solutions can be converted to legal solutions by modification. The process of modifying solutions is proposed as follows:

- Check a particle and record repetitive job numbers on every machine.
- Check absent job numbers on every machine of a particle.
- Sort absent job numbers on every machine (of a particle) according to increment order of their processes.
- Substitute absent job numbers for repetitive codes on every machine of a particle from low dimension to high dimension accordingly.

For example, the following is the positions obtained after applying to the formula.

Position	1	4	4	5	5	6	8	6	9

This position should be scheduled using constructive heuristics process. In the above figure, the position 4 corresponds to job 2 first operation and the same position 4 is repeated, so their next operation is scheduled. Position 5 is repeated twice and job 2 operations are completed. So, the Modifying solution procedure is applied and unscheduled least job operation is scheduled. Likewise, the schedule is obtained for the above positions.

Solution	O_{11}	O_{21}	O_{22}	O_{23}	O_{12}	O_{13}	O_{31}	O_{32}	O_{33}

These steps are repeated for further generations, to find the optimum value for MJSSP.

VI. DATA SET

In this work, 15 instances (LA1-LA10, LA16-LA20) with various sizes for MJSSP from bench mark problems of OR-Library contributed by Lawrence(1984) (Beasley J E ,1990) . have been taken as data set to test our new proposed algorithms. The works taken for comparison to prove the versatility of this proposed algorithm are Simple GA and PSO.

VII. RESULTS AND DISCUSSION

In order to evaluate the reliability of proposed operators in defining MA algorithms, the quality of solution is measured with respect to the high optimal value found. To analyze the results, the best optimal values found for various instances of MJSSP taken from [Lawrence(1984)] OR library for population size 1000 with different sizes(Small, Medium and Large)are compared and is shown in Table.2.

As the evolution progresses, more and more good candidates exist in the next generation and therefore, it can narrow the search space so that fast convergence can be achieved.

A. Performance of Simple GA:

The operators used in GA are roulette wheel selection, partially matched crossover and random mutation. For no instances, the optimal values found by simple GA are better than the best known values found.

B. Performance of Proposed Memetic Algorithm and PSO

From the Table.2, it is observed that,

Simple GA does not beat the best known values for any of the instances

PSO beats the best known values for 2 instances , LA03,LA10.

Memetic Algorithm beats the best known values for 6 instances , LA03,LA06,LA07,LA10, LA17 and LA20

From the results, it is obvious that proposed memetic algorithm is giving optimal values higher than best known values found for 6 instances(LA03,LA06,LA07,LA10, LA17 and LA20) and for other instances giving optimal values same as the best known values. In the case of PSO, only for 2 instances, it is giving results better than the best known values but not better than memetic algorithm. That is , PSO is not giving better results than memetic algorithm for no one instances.

With these discussions, it is concluded that proposed and customized memetic is doing better than existing bio inspired GA and nature inspired algorithm.

Time Complexity:

From the Table.2, it is observed that , memetic algorithm is taking lesser CPU time than simple GA and PSO. In some cases, PSO takes more execution time than simple GA. Hence, the converging speed of population is increased in the proposed algorithm. This is the effort of compensating computational complexity of the proposed operators with its domain specific nature.

Hence, a customized algorithm has been designed to solve multi constrained combinatorial problem with multi objectives.

Instance Name	Instance Size	Best Known Value	High Optimal Values			CPU Time (In m.s)		
			Simple GA	MA	PSO	GA	MA	PSO
LA01	10 x5	666	777	666	666	65552	63552	68754
LA02	10 x 5	655	790	655	655	99935	99835	110222
LA03	10 x 5	597	747	590	590	97749	72749	87522
LA04	10 x 5	590	776	590	590	38452	30452	42512
LA05	10 x 5	593	735	593	593	556821	456821	536256
LA06	15 x 5	926	1015	914	926	59452	39452	65898
LA07	15 x 5	890	978	882	890	30851	29851	32569
LA08	15 x 5	863	956	863	863	32251	34251	36525
LA09	15 x 5	951	1123	951	951	1605871	1405871	1542563
LA10	15 x 5	958	992	951	951	1340359	1240359	1352467
LA16	10 x 10	945	1087	945	945	1140581	994581	1010002
LA17	10 x 10	784	958	779	784	1785467	1985467	1998653
LA18	10 x 10	848	1035	848	848	966542	986542	1000356
LA19	10 x 10	842	1112	842	842	273686	253686	259333
LA20	10 x 10	902	1054	894	902	689684	589684	598675

Table.2.High Optimal Values of MJSSP for different data set

VIII. CONCLUSION

The proposed memetic algorithms with customized GA and SAHC producing more promising results Also, its robustness is proved by obtaining better performance for all problem instances than algorithms taken from the literature. Since, the operators have been designed in order to reduce the complexity of adjusting resources according to the constraints; these models are suitable for optimizing soft constrained combinatorial problems with multi objectives.

REFERENCES

[1] S. A. Schaerf ,"A Survey of automated timetabling", Artificial intelligence review, Vol.13,pp. 87- 127,1999.
[2] Olivia Rossi-Doria and Ben Paechter, " A memetic algorithm for University Course Timetabling", in Proceedings of Combinatorial Optimization(CO'2004), School of Computing, Napier University, Scotland, 2004.
[3] D. Datta, K.Deb, and C.M. Fonseca, " Multi-objective evolutionary algorithm for university class timetabling problem", Series of Studies in Computational Intelligence, Berlin: Springer, Vol. 49, pp.197-236, 2007.
[4] K.Royachka, and M.Karova, "High-performance optimization of genetic algorithms", in IEEE International Spring Seminar on Electronics Technology, 2006, pp.395-400.
[5] Shigenobu Kobayashi, Isao Ono, and sayuki Yamamura,"An efficient genetic algorithm for job shop scheduling problems", ICGA,1995, pp.506-511.
[6] Takeshi Yamada, and Ryohei Nakano,"A fusion of crossover and local search", in IEEE International Conference on Industrial Technology,1996,pp.426-430.
[7] T.Yamada and R.Nakano, Chapter 7: Job Shop Scheduling, In Genetic algorithms in engineering systems, The Institution of Electrical Engineers, London, UK,1997.
[8] D.E.Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. USA: Addison-Wesley, Boston, MS,1989.
[9] K.M.Lee,T. Yamakawa and K.M.Lee," A genetic algorithm for general machine scheduling problems", in International Conference on Knowledge- Based Electronic System, 1998,pp.60-66.
[10] J.G. Qi,G.R.Burns, and D.K.Harrison," The application of parallel multi population genetic algorithms to dynamic job-shop scheduling",International Journal of Advanced Manufacturing Technology,Vol.16, pp.609-615, 2001.
[11] T.Yamada, Studies on metaheuristics for job shop and flow shop scheduling Problems, Japan: Kyoto University, 2003.
[12] Ping-Teng Chang, and Yu-Ting Lo ,"Modeling of Jobshop Scheduling with Multiple Quantitative and Qualitative Objectives and a GA/TS Mixture approach", Int. J. Computer Integrated Manufacturing, Vol.14,No.4,pp.367-384, 2001.
[13] S.M.Kamrul Hasan,M.Rauhul Sarker, and David Cornforth," Hybrid genetic algorithm for solving job-shop scheduling problem", in IEEE International Conference on Computer and Information Science,2007, pp. 519-524.
[14] Christian Bierwirth," A generalized permutation approach to job shop scheduling with genetic algorithm", OR Spectrum, Vol .17, No 2-3, pp.87-92,1995.
[15] M.A.Adibi,M. Zandieh,and M.Amiri," Multiobjective Scheduling of Dynamic Job Shop Using Variable Neighborhood Search", Expert Systems with Applications,Vol.37,No.1, pp.282-287, 2010.
[16] Rui Zhang, "A Particle Swarm Optimization Algorithm based on Local Perturbations for the Job Shop Scheduling Problem", International Journal of Advancements in Computing Technology, Vol.3, No.4, May 2011.
[17] J.E.Garey,D.S.Johnson, and R.Sethi, "The complexity of flowshop and jobshop scheduling",Mathematics of Operations Research, Vol.1,2, pp.117–129, 1976.
[18] E.L.Lawler, J.K.Lenstra, A.H.G.Rinnooy Kan, and D.B.Shmoys,"Sequencing and scheduling: Algorithms and complexity", Handbooks in Operations Research and Management Science,Vol.4, pp.445-552,1993.
[19] S.N.Sivanandham, and S.N. Deepa, Introduction to Genetic Algorithm. New York: Springer Berlin Heidelberg,2008.
[20] J.E.Beasley, "OR-library: Distributing test problems by electronic mail", Journal of the Operational Research Society, Vol.41,No.11, pp.1069–1072,1990.

S. Kanmani received her B.E and M.E in Computer Science and Engineering from Bharathiar University, Coimbatore and Ph.D from Anna University, Chennai.
She has been the faculty of department of Computer Science and Engineering, Pondicherry Engineering College from 1992. Her research interests are in Software Engineering, Software Testing and Object Oriented Systems. She is a member of Computer Society of India, ISTE and Institute of Engineers India.

M. Nandhini received her B.Sc (Mathematics) and M.C.A from Bharathidasan University, Trichirappalli. M.Phil in Computer Science form Alagappa University, Karaikudi.
She is the faculty of department of Computer Science, Pondicherry University, Puducherry. She is doing her research in Artificial Intelligence domain for finding the efficient scheduling for combinatorial problems using hybrid approach. She is a member of ACEEE, ICASIT .