Multi-objective Particle Swarm Optimization Algorithm for Scheduling in Flowshops to Minimize Makespan, Total Flowtime and Completion Time Variance

S. Chandrasekaran, S. G. Ponnambalam, R. K. Suresh and N. Vijayakumar

Abstract: **The present work deals with the development of particle swarm optimization algorithm to solve the multi-objective flowshop scheduling problem. In this paper, minimization of makespan, total flowtime and completion time variance are considered simultaneously. Performance of the proposed methodology has been tested by solving benchmark scheduling problems available in the literature. The proposed methodology is guided to search a set of non-dominated solutions close to the Pareto front. The search capability of the proposed PSO algorithm is enhanced using a local search mechanism. This work is a preliminary step in our research to identify the reference or Pareto solution sets for the benchmark FSPs proposed in the literature, when $(C_{\max})$, (tft) and (ctv) are to be simultaneously optimized.**

I. Introduction

A flowshop is a conventional manufacturing system where machines are arranged in the order of performing operations on jobs. In flowshop, n jobs are considered for processing using m machines. The following assumptions are made in solving the flowshop scheduling problem. (1) Each job has to be processed on all the machines in the order $1,2,.........m$, and the flow is unidirectional. (2) Jobs and machines are available for processing at time zero. (3) Each machine can process only one job at a time and individual operations are not pre-emptable. (4) The operating sequences of the jobs are the same. Various notations and conventions used are presented in Table 1.

Makespan time $(C_{\max})$ is one of the performance measure considered by many researchers [2-5]. It is defined as the maximum time required for completing a given set of jobs. Minimization of makespan time ensures better utilization of the machines and leads to a high throughput. Makespan is computed using

$$C_{max} = max\{C_i, i = 1,2,.......n\} \tag{1}$$

Manuscript received March 31, 2007.

S. Chandrasekaran is with the Sri Ramakrishna Mission Vidyalaya Polytechnic College, Coimbatore, India; (e-mail: chandrussb@yahoo.co.in)

S. G. Ponnambalam is with the School of Engineering, Monash University, 46150 Petaling Jaya, Selangor, MALAYSIA; Phone: 00 603 56360600, Extn: 3240; (e-mail: sgponnambalam@eng.monash.edu.my)

R. K. Suresh is with the School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore, India; Phone: +91 422 2656422, Extn: 222; (e-mail: rks_amrita@yahoo.com)

N. Vijayakumar is with the Acetech Computer Education, Coimbatore – 641020, India; (e-mail: vijay_acetech@yahoo.com)

TABLE I

NOTATIONS AND CONVENTIONS

n	-	Number of jobs to be scheduled
m	-	Number of machines in the flowshop
q	-	Number of objectives
N	-	Swarm size, i.e. Number of particles
t_{ij}	-	Processing time of job i on machine j
C_{ij}	-	Completion time of job i on machine j
C_{max}	-	Makespan of the schedule
tft	-	Total flowtime of the schedule
ctv	-	Completion time variance
P_{kt}	-	Position of k^{th} particle at time step $t\left(k = 1,N\right)$
${}^{e}P_{kt}$	-	Position of the best previous k^{th} particle at time step t, i.e, L_{best}
$Z\left({}^{e}P_{kt}\right)$	-	Makespan value of the schedule represented by the position of the particle ${}^{e}P_{kt}$
G_t	-	The non-dominated solution set obtained at time step t, i.e, G_{best}
t_{max}	-	Maximum number of iterations
N_{nd}	-	Number of non-dominated solutions
V_{xt}	-	Velocity of the particle x at time step t
$\lVert V_{xt}\rVert$	-	Length of list of transportations of the particle at time step t
C_1,C_2,C_3	-	Accelerating constants

Total time taken by all the jobs is called total flowtime (tft) of the schedule. Minimizing total flowtime results in minimum work-in-process inventory [6, 7]. This is given by

$$tft = \sum_{i=1}^{n} C_i \tag{2}$$

Minimizing completion time variance (ctv) serves to minimize variations in resource consumption and utilization [8, 9]. *ctv* is defined as the variance about the mean flowtime and is computed by

$$ctv = \frac{1}{n} \sum_{i=1}^{n} \left(C_i - \bar{F}\right)^2 \quad (3)$$

where $\bar{F} = \frac{F_T}{n}$, is the mean flowtime

Makespan and total flowtime are regular performance measures and completion time variance is a non-regular performance measure. In this paper, minimization of makespan, total flowtime and completion time variance are considered simultaneously. This is because scheduling optimization problems are multi-objective in nature. Simultaneous consideration of the several objectives during scheduling totally modifies the scheduling approach and the goal of such approach is redefined to be [10]. The present work deals with the development of particle swarm optimization algorithm (PSO) to solve the multi-objective flowshop scheduling problem. A position based local search improvement algorithm is applied at the end of all iterations to improve the performance of proposed method. Performance of the proposed PSO algorithm is evaluated by solving benchmark problems proposed by [1].

II. Literature Survey

Flowshop scheduling problem (FSP) is a widely researched combinatorial optimization problem. In this case computational complexity increases with increase in problem size due to increase in number of jobs and/or number of machines. The flowtime minimization problem is NP-complete [5].

Even for single machine scheduling problems, with completion time variance minimization as a performance measure is proved to be NP-hard [11]. Exact solution methods are impractical for solving FSP with large number of jobs and /or machines. Literature shows that heuristics can obtain very good results for such problems. Therefore, researchers have focused more attention on developing heuristics for solving FSPs

Heuristics can be constructive heuristics or improvement heuristics. Constructive heuristics build a feasible schedule from scratch and the improvement heuristics try to improve a previously generated schedule by applying some form of specific improvement methods. Improvement method includes genetic algorithm [12, 13], simulated annealing algorithm [14], tabu search algorithm [15] and particle swarm optimization algorithm [16-18].

Metaheuristic algorithms such as simulated annealing and tabu search based methods are single point local search procedures were a single solution is improved continuously by a solutions improvement procedure.

Algorithms such as genetic algorithm, ant colony algorithm and particle swarm optimization algorithm belong to population based search algorithms. These are designed to maintain a set of solution transiting from a generation to the next. This is more suitable for solving multi-objective optimization problems.

PSO algorithm has been introduced as a solution methodology for continuous non-linear optimization functions. A discrete PSO algorithm for solving flowshop scheduling with the objective of minimizing makespan has been proposed by Rameshkumar et al [16]. PSO algorithm has been proposed for solving flowshop scheduling problem with the objective of minimizing total weighted tardiness in a single machine flowshop [17].

Real life scheduling problems are multi-objective in nature. There exist no single solution which can meet all the criteria under consideration, especially when the objectives under consideration are conflicting in nature. Therefore, solution to such multi-objective problems is a set of non-dominated or Pareto solutions [19]. Some researchers have developed multi-objective metaheuristics for solving flowshop scheduling problems [10, 20-25].

A multi-objective PSO algorithm is proposed for minimizing weighted sum of makespan and maximum earliness [18]. Two simulated annealing heuristic algorithms using the concepts of forward and backward scheduling approach for minimizing makespan and *ctv* in jobshops has been proposed [9]. A Pareto ranking based multi-objective genetic algorithm has been proposed with the objective of generating a set of non-dominated solutions [20].

A heuristic algorithm has been proposed by Chandrasekharan Rajendran [21] with three objectives namely makespan, flowtime and machine idle time which are considered separately. Suresh and Mohanasundaram [10] have proposed a Pareto archived simulated annealing algorithm for multi-objective scheduling. A modified multi-objective genetic algorithm for solving flowshop scheduling is presented in [25]. In this paper, PSO algorithm has been suitably modified to generate non-dominated solution set considering the three performance measures simultaneously.

III. Single-Objective Particle Swarm Optimization Algorithm

PSO Algorithm is a population based evolutionary computation technique [26, 27]. It mimics the behavior of flying birds and their means of information exchange to solve optimization problems. It works with a population of several particles called *'swarm'*. A set of moving particles (the swarm) is initially thrown inside the multi-dimensional search space. Each particle is a potential solution, which has the ability to remember its previous best position and current position, and it survives from generation to generation [28].

Initially, a set of solutions, i.e. *N* number of particles are generated randomly. Particle consists of a string of integers with number of integers equal to number of jobs to be scheduled. After the swarm is initialized, each potential solution is assigned a velocity randomly. The length of the velocity of each particle $\|V\|$ is generated randomly between 0 and n [16, 29], and the corresponding lists of transpositions $(i_q, j_q) : q = 1, \|V_k\|$ are generated randomly for each particle. The above formulation permits exchange of jobs $(i_1, j_1), (i_2, j_2) \ldots (i_{\|v\|}, j_{\|v\|})$ in the given order. Each particle keeps track of the its improvement and the best objective function value achieved by the individual particles so far is stored as local best solution ${}^eP_k$, and the overall best objective function achieved by all the particles together so far is stored as the global best solution (G_t). The iterative improvement process, as explained in Fig. 1 is continued afterwards.

```
    $t = 0;$
do{
    $for\ (k = 1, N)$
        Update Position $P_k^{t+1}$;
        Update velocity $v_k^{t+1}$;
    Apply local search on all particle positions;

    Evaluate all particles;
    Update ${}^eP_k^{t+1}$ and $G^{t+1}$, $(k = 1, N)$;
    $t \rightarrow t+1$;
}while $(t < t_{\max})$
```

Fig.1: Iterative Search Loop of the PSO Algorithm

It is to be noted that particle velocity and position are continuously updated using (4) and (5).

$$v_k^{t+1} = C_1 U_1 v_k^t + C_2 U_2\, rand()({}^eP_k^{t+1} - P_k^{t+1}) + C_3 U_3\, rand()(G - P_k^{t+1}) \qquad (4)$$

$$P_k^{t+1} = P_k^t + v_k^{t+1} \qquad (5)$$

The acceleration constants C_1, C_2 and C_3 in (4) guide every particle toward the local best and the global best solution during the search process. Low acceleration value results in walking far from the target, namely local best and the global best. High value results in premature convergence of the search process [29].

IV. Multi-Objective Particle Swarm Optimization Algorithm

Multi-objective optimization usually results in a set of non-dominated solutions instead of one single solution. The goal is to find as many different non-dominated solutions as possible, so that the decision maker can choose an appropriate schedule meeting the prevailing shop floor requirements at the time of decision making. Before presenting the proposed algorithm, the non-dominated sorting procedure, Pareto search procedure and the local search method adopted are discussed below.

A. Non- domination Sorting

Among a set of solutions *X*, a solution $x^1 \in X$ is said to dominate the other solution $x^2 \in X$ denoted as $(x^1 \succ x^2)$ when, both the following conditions are true [19].

i) The solution $x^1 \in X$ is no worse than $x^2 \in X$ in all objectives.
ii) The solution $x^1 \in X$ is strictly better than $x^2 \in X$ in at least one objective.

When both the conditions are satisfied, x^2 is called as a dominated solution and x^1 as a non-dominated solution. If any one of the above conditions is violated, the solution x^1 does not dominate the solution x^2. Among a set of solutions X, the non-dominated set X' are those that are not dominated by any member of the set.

The above conditions are used to find non-dominated set of solutions. Consider a set of N solutions (particles). The following procedure is used to generate non-dominated solution set [19].

Step 0: Begin with $i = 1$; $j = i + 1$, and repeat steps

Step 1: Compare solutions x^i and x^j for domination using the two conditions mentioned above considering all the objectives.

Step 2: If x^j is dominated by x^i, mark x^j as 'dominated,' increment *j*, and go to Step 1. Otherwise mark x^i as dominated, increment *i*, set $j=i+1$ and go to step 1.

All solutions that are not marked 'dominated' forms a non-dominated solution set.

B. Pareto Search

In case of a single objective scheduling optimization, a single solution forms the Global best (G). With more than one objectives, G will consist of a set of non-dominated solutions.

Once the swarm is initialized, $G_t\ (t = o)$ is obtained after non-dominated sorting of the particles as given in section IV-A. During the next iteration, position and velocity update of the particles are carried out using local best and global best. It is to be noted that one solution is randomly chosen from the Archive as Global best. During the next iteration, again a new set of non-dominated solution is identified.

This non-dominated solution set is added with the Archive and the combined set is sorted for non-dominance. Dominated solutions within the combined set are removed and the remaining non-dominated solutions forms $G_t\ (t = 1)$. This procedure is repeated to guide the non-dominated search process towards the Pareto region.

C. Local Search Algorithm

A position based local search improvement algorithm is used for the quick search. Fig. 2 shows the process of local search algorithm.

```
Step 1)  Select a Particle P;
Step 2)  Initialize i =1, j = i+1;
                 // i and j are job positions in a
                         particle;
Step 3)  Select i
Step 4)  Swap i with j to generate a new sequence
Step 5)  Evaluate the new sequence;

Step 6)  if (new sequence is better)
            replace the Particle P with the new
              sequence;
            terminate the search process;
         else
              swap j with i;
              j=j+1;
         if (j = n)
              i = i+1; j = i+1;

              go to step 4
                   //to continue the search process;
```

Fig. 2: Position based local search improvement algorithm

D. Parameter Setting

The proposed PSO algorithm is coded in C language and run on an Intel Pentium IV 900MHz PC. The swarm size is taken as 80. The values of acceleration constants are fixed by trial and error as $C_1 = 1; C_2 = 2$ and $C_3 = 2$. The acceleration constants C_1, C_2 and C_3 guides every particle towards the local best and global best solutions during the search process. The values of U_1, U_2 and U_3 are generated randomly between 0 and 1 during the iterative improvement process. Termination criterion is taken as 100 iterations.

E. Proposed PSO Algorithm

Initially, a set of particles and corresponding velocities are generated randomly. All the particles are evaluated and the non-dominated sorting is done. Within the swarm, the non-dominated solution i.e. G_t is identified through non-dominated sorting and they are stored in an archive. Then the positions and velocities of the particles are updated iteratively. Velocity update is done to improve every solution into a non-dominated solution. After every iteration, non-dominated sorting is done to identify the current set of non-dominated solutions. These current sets of non-dominated solutions are combined with the archive solutions. Non-dominated sorting of archive is done to identify the archive survival members. During this, all dominated members of the combined set are removed. These archive members form the global best solution set which is used to update velocity during every iteration. After the termination criterion is met, the solution set stored in the archive forms the result. The iterative improvement process of multi-objective PSO algorithm is presented in Fig.3.

```
Initialize the parameters
        Generate the swarm
         t = 0
        Evaluate all the particles
        Perform non-dominated sorting to identify $G_t$
        Open Archive to store $G_t$
        Initialize velocity for each particle

do      {
        for k=(1.N)
        Update position
        $t = t + 1$
        Evaluate
        Do non-dominated sorting to identify $G_t$
        Archive update
        Do local search
        Update velocity

        }     until $(t < t_{\max})$
        Output $G_t$
```

Termination criterion is set arbitrarily as 100 iterations.

Fig.3: Iterative search loop of the multi-objective PSO algorithm

V. Results And Discussions

The performance of the multi-objective PSO has been evaluated by solving selected FSPs found in the literature Problems with jobs varying from 20 to 500 and machines varying from 5 to 20 are solved. Non-dominated solution set obtained by the multi-objective PSO after 100 iteration are presented in the Appendix. Fig. 4 to Fig. 15 shows the non-dominated front obtained during 1st, 50th and 100th iteration for a selected flowshop scheduling problems. Normalized values of the performance measures are plotted for better visualization.

It is clear from the results that the non-dominated solution set moves toward the Pareto front during the search.

VI. Conclusion

A multi-objective PSO algorithm with a position based local search is proposed. The result shows that the proposed multi-objective PSO algorithm performs better in terms of yielding more number of non-dominated solutions. With more iterations, results can be further improved. This work is a preliminary step in our research to identify the reference solution sets for the FSPs discussed in the literature, when $(C_{\max})$, (*tft*) and (*ctv*) are to be simultaneously optimized. With more iterations, it is possible to generate a reference solution set for every problem which can be used as a benchmark by future researchers.

VII. Appendix

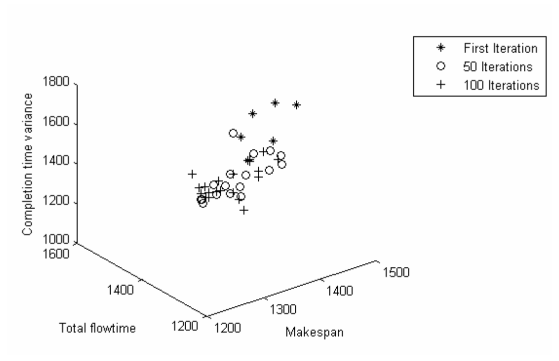

Fig 4: Results for the problem ta001

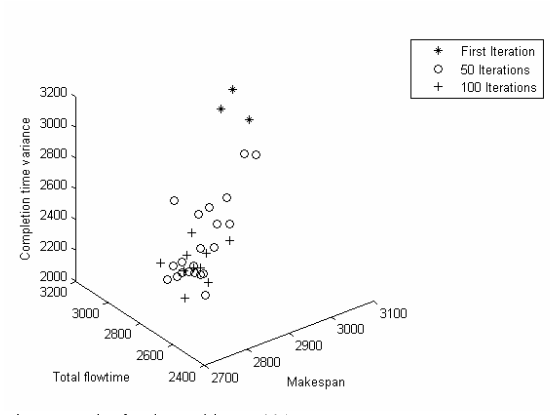

Fig 7: Results for the problem ta031

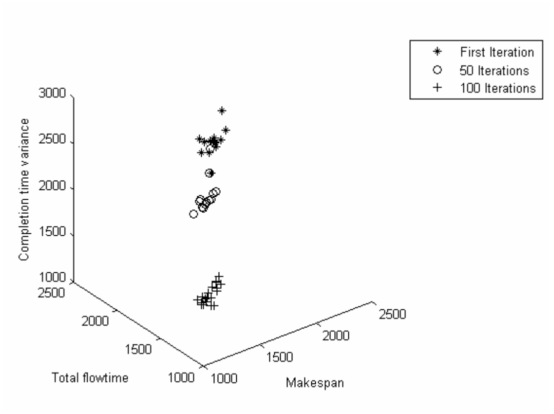

Fig 5: Results for the problem ta011

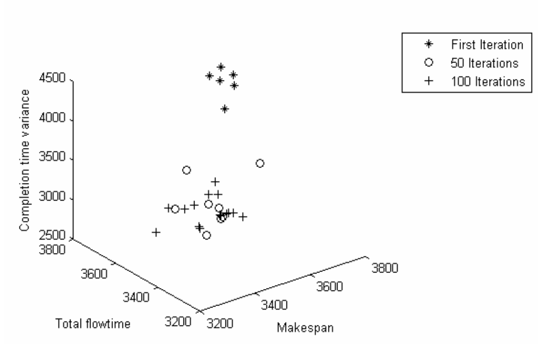

Fig 8: Results for the problem ta041

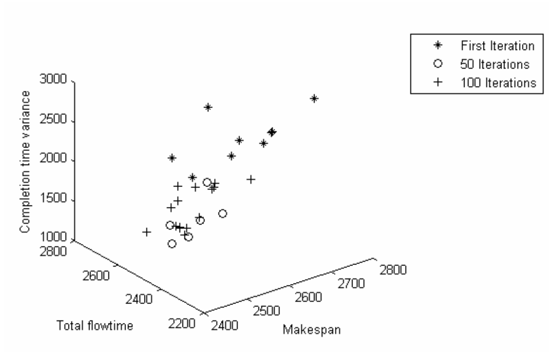

Fig 6: Results for the problem ta021

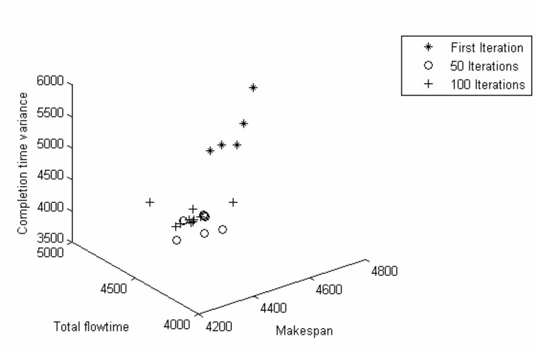

Fig 9: Results for the problem ta051

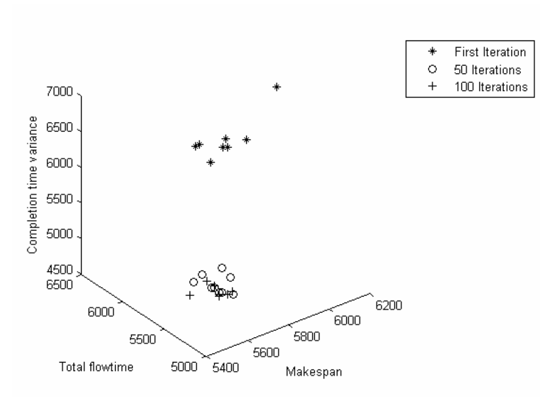

Fig 10: Results for the problem ta061

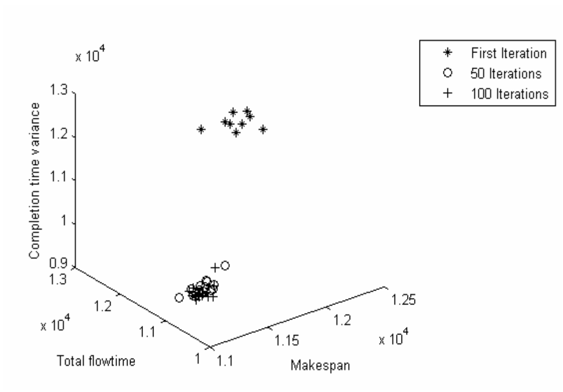

Fig 13 Results for the problem ta091

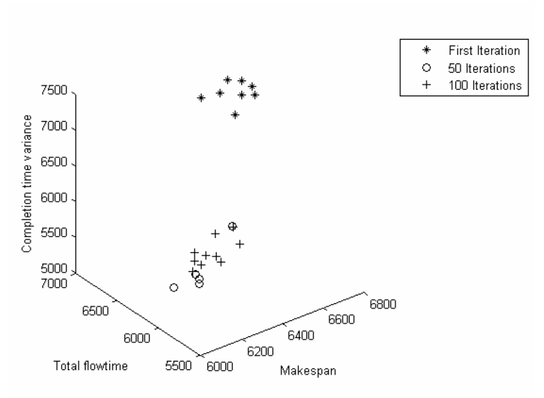

Fig 11 Results for the problem ta071

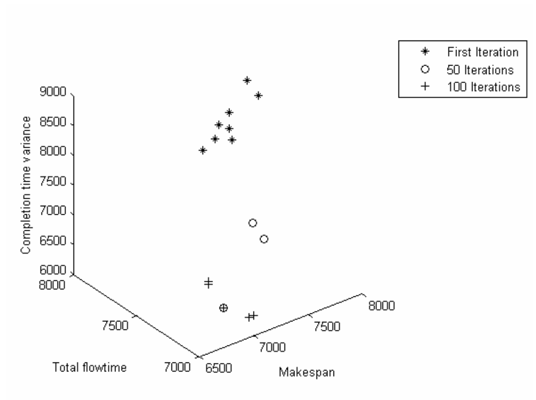

Fig 14 Results for the problem ta101

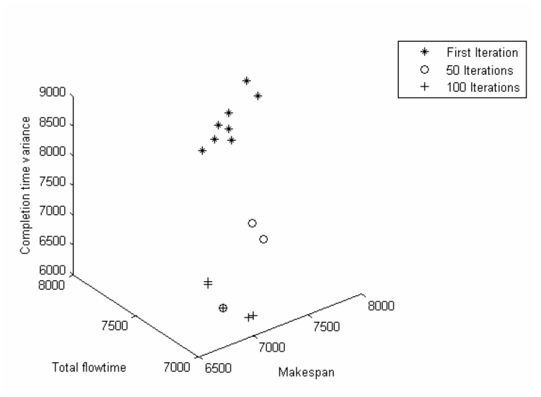

Fig 12 Results for the problem ta081

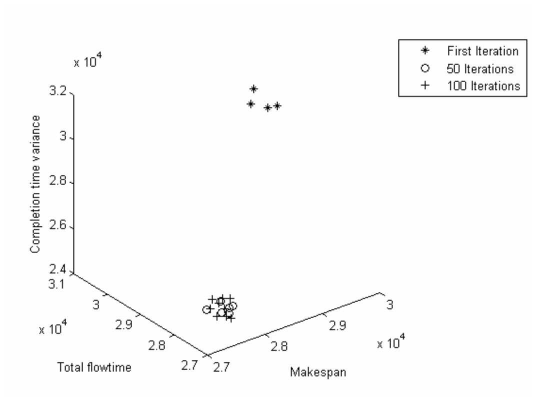

Fig 15 Results for the problem ta111

VIII. References

[1] E. Taillard, “Benchmarks for the basic scheduling problem,” *European Journal of Operational Research*, vol. 64, pp. 278 – 285, 1993.

[2] E.Ignall, and L.Scharge, “Application of the branch and bound technique to some flow shop-scheduling problems,” *Operations Research*, vol. 13, pp. 400-412, 1965.

[3] D.G. Dannenbring, “An evaluation of flow shop sequencing heuristics,” *Management Science*, vol. 23, pp. 1174-1182, 1977.

[4] M. Nawaz, E.E. Enscore, and I. Ham, “A heuristic algorithm for the m-machine n-job flow shop-sequencing problem”. *Omega*, vol. 11, pp. 91-98, 1983.

[5] M.R. Garey, D.S. Johnson, and R. Sethi, “The complexity of flow shop and shop job scheduling,” *Mathematics of Operation Research*, vol. 1, pp. 117-129, 1976.

[6] C. Rajendran and H. Ziegler, “Ant-colony algorithms for minimizing total flowtime in permutation flow shops”, *Computers & Industrial engineering*, vol. 48, pp 789-797, 2005.

[7] C. Rajendran and H. Ziegler, “Ant-colony algorithms for permutation flowshop scheduling to minimize makespan / total flowtime of jobs,” *European Journal of Opn. Research*, vol. 155, pp. 426-438, 2004.

[8] Yuvraj Gajpal and Chandrasekharan Rajendran, “An ant-colony algorithms for minimizing the completion time variance of jobs in flowshop,” *International Journal of Production Economics*, 2005.

[9] Viswanath Kumar Ganesan, Appa Iyer Sivakumar and G. Srinivasan, “Hierarchical minimization of completion time variance and makespan in jobshops”, *Computers &Operations Research*, vol. 33, pp 1345-1367, 2006.

[10] R.K. Suresh and K.M. Mohanasundaram, “Pareto archived simulated annealing for permutation flow shop scheduling with multiple objectives”, *Proceedings of the IEEE conference on Cybermatics and Intelligent Systems*, 1-3 Dec 2004, Singapore

[11] W. Kusiak, “Completion time variance minimization on a single machine is difficult,” *Operations Research Letters*, vol.14, pp. 49-59, 1993.

[12] J. Sridhar and C. Rajendran, “Scheduling in flow shop and cellular manufacturing system with multiple objectives- A genetic algorithmic approach,” *Production Planning and Control*, vol.74, pp. 374-382, 1996.

[13] F.T.S. Chan, T.C. Wong, T.C, and L.Y. Chan, “A genetic algorithm-based approach to machine assignment problem.” *International Journal of Production Research*, vol.43, No.12, pp. 2451-2472, 2005.

[14] F.A. Ogbu and D.K. Smith, “The application of the simulated annealing algorithm to the solution of the n /m /C_{max} flow shop problem,” *Computers and Operations Research*, vol. 17, pp 243 – 253, 1990.

[15] J.V. Moccellin. M.S. Nagamo, “Evaluating the performance of tabu search procedures for flow shop sequencing,” *Journal of the Operational Research Society*, vol. 49, pp. 1296-1302, 1998.

[16] K. Rameshkumar, R.K. Suresh and K.M. Mohana sundaram, “Discrete particle swarm optimization (DPSO) algorithm for permutation flow shop scheduling to minimize makespan,” *Lecture Notes in Computer Science, Springer Verlag- GMBH*. 0302-9743. vol. 3612, pp. 2005.

[17] Faith Tasgetiren, Mehmet Sevkli, Yen-Chia Liang and Gunes Gencyilmaz, “Particle swarm optimization algorithm for single machine total weighted tardiness problem,” *IEEE Transaction on Power and Energy Systems*, pp.1412 – 1419, 2004.

[18] G. Prabhaharan, B. Shahul Hamid Khan, P. Asokan and M. Thiyagu, “A Particle swarm optimization algorithm for permutation flow shop scheduling with regular and non-regular measures,” *International Journal of Applied Management and Technology*, vol. 3, No.1, pp.171-182, 2005.

[19] Kalyanmoy Deb, “Multi-Objective Optimization Using Evolutionary Algorithms*”,* First Edition*, John Wiley & Sons*, 2003.

[20] T. Pasupathy, Chandrasekharan Rajendran and R.K. Suresh, “A multi-objective genetic algorithm for scheduling in flow shops to minimize makespan and total flow time”, International Journal of Advanced Manufacturing Technology, Publisher: Springer-Verlag London Ltd, vol. 27, pp 804-815, 2006.

[21] Chandrasekharan Rajendran, “Heuristics for scheduling in flowshop with multiple objectives”, *European Journal of Operational Research*, vol. 82, pp 540-555, 1995.

[22] Ali Allahverdi, “The two- and m- machine flowshop scheduling problems with bi-criteria of makespan and mean flowtime”, *European Journal of Operational Research,* vol. 147, pp 373-396, 2003

[23] Ali Allahverdi, “A new heuristic for m-machine flowshop scheduling problem with bicriteria of makespan and maximum tardiness”, *Computers & Operations Research*, vol. 31, pp 157-180, 2004.

[24] Vincent T’kindt, Nicolas Monmarche, Fabrice Tercinet and Daniel Laugt, “An ant colony optimization algorithm to solve a 2-machine bicriteria flowshop scheduling problem”, *European Journal of Operational Research*, vol. 142, pp 250-257, 2002.

[25] Hisao Ishibuchi, Tadashi Yoshida and Tadahiko Murata, “Balance between genetic search and local search in memtic algorithms for multiobjective permutation flowshop scheduling”, *IEEE Transactions on evolutionary computation*, vol. 7, No. 2, pp 204-223, 2003.

[26] R.C. Eberhart, J. Kennedy, “A new optimizer-using particle swarm theory,” *In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, Japan, pp.39-43, 1995.

[27] J. Kennedy, R.C. Eberhart and Y. Shi, “Swarm Intelligence,” *Morgan Kaufmann*, San Mateo, CA, USA, 2001

[28] Yuhui Shi, “Particle Swarm Optimization,” *IEEE Neural Networks Society*, pp. 8-13, Feb 2004.

[29] S. Chandrasekaran, S.G. Ponnambalam, R.K. Suresh, N. Vijayakumar. "An Application of Particle Swarm Optimization Algorithm to Permutation Flowshop Scheduling Problems to Minimize Makespan, Total Flowtime and Completion Time Variance," In Proceedings of the IEEE International Conference on Automation Science and Engineering, 2006 (CASE '06.), Shanghai, China, pp. 513-518, ISBN: 1-4244-0311-1.