American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)

ISSN (Print) 2313-4410, ISSN (Online) 2313-4402

http://asrjetsjournal.org/

Application of an Efficient Genetic Algorithm for Solving n× *m* Flow Shop Scheduling Problem Comparing it with Branch and Bound Algorithm and Tabu Search Algorithm

Md. Humayan Ahmed[a]*, Romana Rahman Ema[b], Tajul Islam[c], Rana M Pir[d]

[a,d]*Department of Computer Science and Engineering , Sheikh Fazilatunnesa Mujib University, Jamalpur-2000, Bangladesh*

[b,c]*Department of Computer Science and Engineering, North Western University, Khulna-9100, Bangladesh*

[a]*Email: humayan036@gmail.com*

[b]*Email: romanacsejstu@gmail.com*

[c]*Email: tajulkuet09@gmail.com*

[d]*Email: ranapir@yahoo.com*

Abstract

Emergence of advance manufacturing systems such as CAD/CAM, FMS and CIM etc. has increased the importance of the flow shop scheduling. Flow shop scheduling problem is considered NP-hard for m machines and n jobs. In this paper, we develop an efficient genetic algorithm (EGA) for solving n× *m* flow shop scheduling problem with makespan as the criterion. The objective of this proposed EGA is to obtain a sequence of jobs and the minimization of the total completion time and waiting time. For finding optimal solution, this EGA is very effective. In large scale problems, the result of the proposed algorithm shows that the efficient genetic algorithm gives high performance comparing with Branch and Bound algorithm and Tabu search algorithm.

Keywords: Flow shop scheduling; Efficient genetic algorithm (EGA); Makespan; Branch and Bound algorithm; C++ code; Tabu search algorithm.

---

* Corresponding author.

1. Introduction

Flow shop scheduling refers to gain a sequence of jobs in which all jobs are to be visited on various machines. All jobs must process on machines in the same order and this process is called permutation flow shop scheduling [1-4]. On the other hand, jobs that can have different sequences on each machine are called non-permutation flow shop scheduling. Computationally, the research in flow shop scheduling is a great challenge to increase the effectiveness of industrial production. A flow shop consists of m machines in series and n different jobs where each job must be processed by the m machines in the same order [5-9]. This characteristic differentiates it from a job shop [10-11]. In this paper, the machines are numbered 1,2,….,m; the jobs are numbered 1,2,….,n and the processing time for the operation of job is P_{i1}, p_{i2},….,P_{ij}. In medium size problems, optimal solution can be obtained by Branch and Bound technique and Tabu search method but these are inadequate in large size problem. As the flow shop scheduling problem is considered NP-hard for m>=3 [12], GA could be used to deal with this large size problem. The aim of this paper is to develop EGA for flow shop scheduling to minimize the makespan of the tasks. The rest of the paper is organized as follows: section 2 describes the related work, assumptions are presented in section 3. Section 4 discusses the efficient GA for flow shop scheduling problem, implemented Tabu Search and Branch and Bound algorithm are illustrated in section 5 and section 6 respectively. Performance evaluation with implementation is examined in section 7. Finally, Conclusion and Recommendations are discussed in section 8 and section 9 respectively.

2. Related Work

In recent years, the scheduling problem with the makespan minimization criterion has attracted the attention of researchers because of its applications in practice. For example: sequencing of programs to be run in computer center, sequencing of jobs for processing in a manufacturing plant etc. The main issue deals with sequencing n jobs that has to be processed on m machines so that the manufacturing system performs better in terms of performance measures such as minimum makespan, minimum tardiness and minimum work-in-process. Johnson has presented an algorithm for two machines, n-jobs flow shop scheduling problem by considering total completion time.

Lomnicki [13] has proposed a Branch and Bound method for flow shop scheduling problem. The work has further improved by Deepak Gupta, Singla and Bala [14] with the branch and bound technique for n×3 flow shop scheduling with breakdown interval.

Osman and Potts [15] have introduced simulated annealing heuristics. Widmer and Hertz [16] and Taillard [17] have provided Tabu search heuristics for flow shop scheduling. The main objective of the paper is to minimize the maximum computation time for flow shop problems. Glass and his colleagues [18], Leisten and Kolbe [19], Sadjadi and his colleagues [20] proposed some model for flow shop scheduling. Those above methods used by several researchers for flow shop scheduling problem is not applicable for large size problems. Reeves [21] proposed a genetic algorithm for flow shop sequencing. As flow shop scheduling problem is NP-hard for m>=3, researchers focused on genetic algorithm [22-23] for obtaining high efficiency.

In this work, an efficient genetic algorithm is presented for the objective of minimizing the makespan and waiting time comparing with Tabu search and Branch and Bound method.

3. Assumptions

In this study, the following assumptions are made:

- Each job can be processed at most one on each machine.
- All the jobs are processed on all the machines in the same order.
- Machines are not allowed to process more than one job at a time.
- A single operation of the same job is performed at the same time.
- The number of jobs waiting is allowed for processing.
- If a job is not processed on a machine, the processing time will be zero.

Parameters:

n: number of jobs.

i: job index, (i =1,2,..,n)

m: number of machines.

J: machine index, (j =1,2,…..,m)

$C_{ij:}$ completion time for job i on machine j.

$C(M_i(k))$: total completion time (makespan) of i^{th} chromosome in k^{th} generation for proposed EGA.

S_o: optimal sequence.

4. Proposed Efficient Genetic Algorithm (EGA) for Flow Shop Scheduling Problem

Genetic algorithms come from the concept of a great source of natural selection and genetics inspiration. It is a search based optimization technique. To get optimal solution for large size problem, GA gives better solution for flow shop scheduling.

GA was first developed by Holland [24]. He used GA where each gene represents a feature of the problem and chromosomes are used to encode the factor of a problem. By using reproduction, crossover, mutation operator, the compilation of genes is evolved and gives the optimal solution of the chromosomes. Our proposed efficient genetic algorithm (EGA) is presented as follows.

4.1. Chromosome representation

In the present work, we assumed that all jobs are selected randomly and they have an order on any machine. The chromosome is a job sequence vector on machine where each gene is a job in the chromosome. Consider n jobs where: i=1,2,3, ….,n may be processed on m machines where: j=1,2,3,……,m with processing time P_{ij} given below:

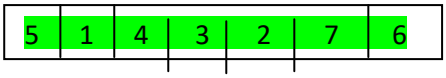
Table 1: Processing time table

Job\Machine	M1	M2	M3	M4	M5
Processing Time, P_{ij}(Min)	Pi1	Pi2	Pi3	Pi4	Pi5
J1	3	4	6	8	4
J2	8	3	7	7	10
J3	7	2	5	9	23
J4	4	5	11	2	25
J5	9	1	5	3	14
J6	8	4	6	4	5
J7	7	3	12	6	8
Total Time	46	22	52	39	89

The job sequence vector on machines is illustrated in figure 1.

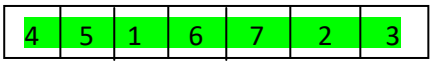
Machine M1: J5-J1-J4-J3-J2-J7-J6

Job sequence on machine 1:



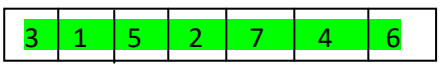
Machine M2: J4-J5-J1-J6-J7-J2-J3

Job sequence on machine 2:



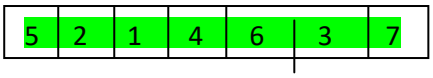
Machine M3: J3-J1-J5-J2-J7-J4-J6

Job sequence on machine 3:



Machine M4: J5-J2-J1-J4-J6-J3-J7

Job sequence on machine 4:



Machine M5: J1-J7-J4-J6-J2-J5-J3

Job sequence on machine 5:

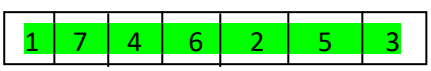


Figure 1: job sequence vector

In this paper, we use maximum population size, Ps=30, k= 0 and number of generation, NG=0 as initialization.

4.2. Fitness function calculation

As our objective is to minimize the makespan, fitness value calculation is the main criterion of a substructure. Fitness function of the job sequence or chromosome can be calculated as follows:$f(M_i(k)) = max\{C(M_i(k))\} - C(M_i(k))$Where: $f(M_i(k))$= fitness function of i^{th} chromosome in k^{th} generation which gives the fitness value and $C(M_i(k))$=Total completion time (makespan) of i^{th} chromosome in k^{th} generation. Another criterion is needed for the selection of job sequences and this is selection probability $P_h(M_i(k))$ for a schedule with lower makespan is computed as follows: $P_h(M_i(k)) = \frac{f(Mi(K))}{total\ fitness\ of\ the\ current\ chromosomes}$ Our implementation of efficient genetic algorithm (EGA) is illustrated in figure 2.

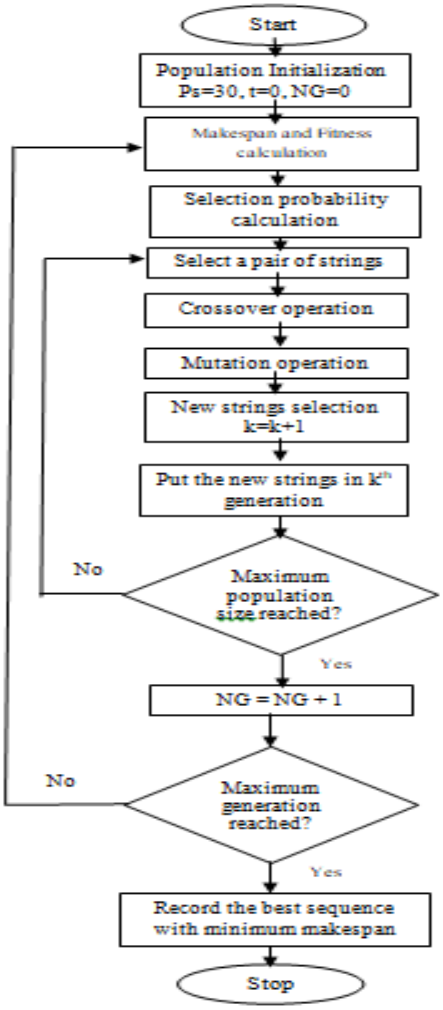


Figure 2: flow chart of proposed efficient genetic algorithm

4.3. Crossover

Crossover is a genetic operator which is needed to vary the programming of chromosomes from generation to

next. Our crossover mechanism is considered as follows:

Step 1: First of all, choose two chromosomes (parent 1 and parent 2) randomly, select a pair of chromosomes according to selection probability and obtain the chromosome with high selection probability is Parent 1. However, parent 2 is obtained by the high selection probability among the selected pair of chromosomes. follow the step 2 for the same machine from parent 1 and parent 2.

Step 2: from parent 1, randomly select two points of job sequence for i^{th} machine and generate an offspring by copying the two point substrings of job sequence into the corresponding positions. Then starting with first position from i^{th} machine of parent 2 and delete the jobs from i^{th} machine of the parent 2 which are in the two points substrings. The remaining sequence of jobs in i^{th} machine of parent 2 is needed for offspring. Finally, put the remaining jobs into the empty position of the offspring from left to right according to the order of the sequence in the i^{th} machine of parent 2.

The process is described in figure 3.

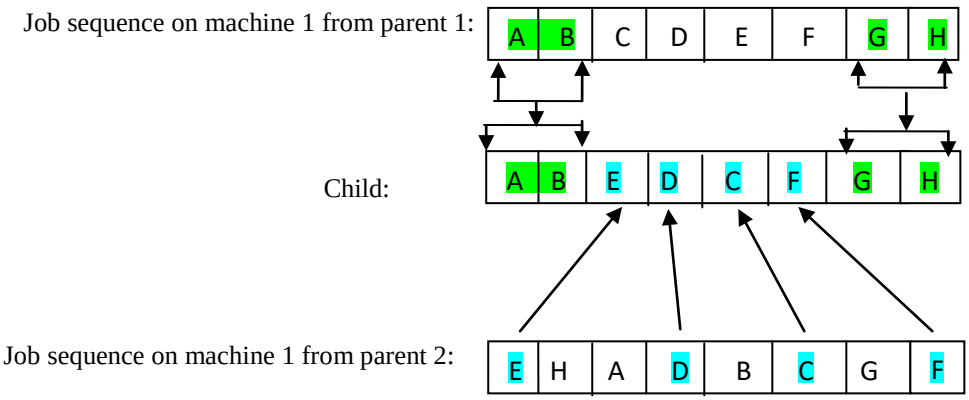


Figure 3: crossover operation

4.4. Mutation

Mutation operator is the best way to maintain genetic diversity from one generation to the next. Our mutation mechanism is as follows: First of all, randomly choose one chromosome then a single job is selected randomly from i^{th} machine of selected chromosome. Finally, insert the selected job in a random position. The process is described in figure 4.

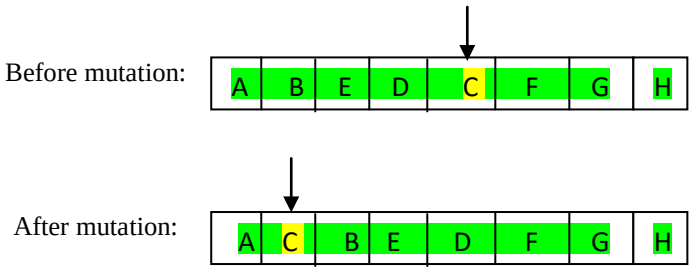


Figure 4: mutation operation

4.5. Termination

We use the number of generations (NG) as the termination criterion. If the algorithm reaches maximum number of generations, it will terminate in that time.

5. Tabu Search (TS)

Tabu search is a neighborhood based search technique [25-27]. We use neighborhood scheme and tabu list in this section. Our implemented Tabu search algorithm steps are given below:

a) Initial solution:

First of all, we use Branch and Bound method as if we get the best starting solution.

b) Neighborhood scheme:

In our implemented TS algorithm, we use neighborhood scheme where a sequence of job is given and we obtain all the sequences of jobs using swap operator or insertion operator.

c) Choose the best neighboring solution:

From neighborhood scheme, we calculate the makespan of all the sequences. As our goal is to obtain a sequence with smallest makespan, select the first sequence for the improvement of total completion time is considered. Otherwise, examine the whole neighborhood to get better sequence with smallest makespan is considered.

d) Tabu list:

In our experiment, we use the tabu list with the fixed length where job sequence with makespan is stored. A tabu move is possible when it leads to the better solution than the best obtained. In tabu list, a newly one replaces the oldest one.

e) Stopping criteria:

To check the maximum number of iteration is the stopping criteria in our TS algorithm.

6. Branch and Bound algorithm

At present the Branch and Bound gives optimal solutions. Furthermore, as heuristics do not usually produce optimal solutions, the goal of the Branch and Bound is to solve a constrained optimization problem [28]. Branch and Bound is another method for solving combinatorial optimization problems. It is based on the idea of intelligently enumerating all feasible solutions. To explain the details, we assume again that the discrete optimization problem P to be solved is a minimization problem. We also consider sub problems of P which are defined by a subsets S of the set S of feasible solutions of P. It is convenient to identify P and its sub problems

with the corresponding subset S ⊆S.

Band B is characterized by the three following basic components:

1) Branching rule:

S is replaced by smaller problems S_i(i =1,...,r) such that r_i=1, S_i=S. This process is called branching. Branching is a recursive process, i.e. each Si is the basis of another branching. The whole branching process is represented by a branching tree. S is the root of the branching tree, S_i (i =1,...,r) are the branch node of S, etc. The discrete optimization problems created by the branching process are called sub problems.

2) A lower bounding:

An algorithm is available for calculating a lower bound for the objective values of all feasible solutions of a sub problem.

3) Upper bounding:

We calculate an upper bound U of the objective value of P. The objective value of any feasible solution will provide such an upper bound. If the lower bound of a sub problem is greater than or equal to U, then this sub problem cannot yield a better solution for P. Thus, we need not continue to branch from the corresponding node in the branching tree. To stop the branching process in many nodes of the branching tree, the bound U should be as small as possible. Therefore, at the beginning of the branch and bound algorithm we apply some heuristic to find a good feasible solution with small value U. After branching many times we may reach a situation in which the sub problem has only one feasible solution. Then the lower bound LB of the sub problem is set equal to the objective value of this solution and we replace U by LB if LB < U.[29]. The lower bound calculation and our improved Branch and Bound algorithm are given below:

6.1. Lower Bound (LB) calculation

The lower bound calculation equation is needed to find out the minimum makespan of our five machines is as follows:

$$\begin{cases} \text{(i) LB1: } \sum \text{Pi1+min (Pi2+pi3+Pi4+Pi5)} \\ \text{(ii) LB2: MinPi1+}\sum \text{ Pi2+Min (Pi3+Pi4+Pi5)} \\ \text{(iii) LB3: Min (Pi1+Pi2) +}\sum \text{ Pi3 +min (Pi4+Pi5)} \\ \text{(iv) LB4: Min (Pi1+Pi2+ Pi3) +}\sum\text{Pi4 +minPi5} \\ \text{(v) LB5: Min (Pi1+Pi2+ Pi3+Pi4) +}\sum\text{Pi5} \end{cases} \quad (1)$$

6.2. Improved Branch and Bound Algorithm (IBBA) for flow shop scheduling problem

Algorithm 1. IBBA

Ensure; let m be the number of machine, n be the number of jobs and P_{ij} be the processing time.

1: Calculate the addition of P_{ij} for each machine.

2: Calculate the lower bound value for each machine from lower bound equation (1).

3: Choose the largest lower bound from step2 and create branch for each job.

4: Calculate lower bound values of each job.

5: Put the largest lower bound value of each job from their calculated lower bound values.

6: Choose the smallest lower bound and the related job from step5.

7: If there is more than one smallest lower bound, then search to choose the smallest lower bound and put the value into memory else, select that only one job with smallest lower bound and put it into memory.

8: Create branch for the remaining jobs.

9: Repeat step 4 to step 7 until we reach at the end of the tree. However, we get the best sequence with minimum makespan.

7. Performance Evaluation

In this section, performance evaluation of our proposed efficient GA comparing with Tabu search and Branch and Bound algorithm is presented by the illustration of experimental environment and computational result.

7.1. Experimental Environment

In our presented EGA, we use object oriented programming language with C++. We are familiar with it and we use it very easily to feel comfort to implement the above algorithms. Inputs such as number of jobs, number of machines and job to machine timing are taken using notepad and our implemented C++ program outputs optimum sequence and optimum time (makespan).

7.2. Computational Results and Discussion

In this paper, efficient genetic algorithm is used for solving different size of flow shop scheduling problems. In this experiment, we organize some tests for the job range from 5 to 30 jobs and the machine range between 5 and 30 machines with their processing time range between (6, 25). Different size of problems considered is shown in table 2.

Table 2: Different size of problems (n× m)

Number of jobs n	Number of machines m
7	5
9	9
10	15
25	15
30	30

Table 3: The comparison result of Branch and Bound, Tabu Search and Efficient GA.

Number of jobs n	Number of machines m	Problem Size	Algorithm	Makespan (Min.)
7	5	Small	Branch and Bound	107
			Tabu Search	107
			Efficient GA	107
9	9	Small	Branch and Bound	144
			Tabu Search	144
			Efficient GA	144
10	15	Large	Branch and Bound	272
			Tabu Search	272
			Efficient GA	263
25	15	Large	Branch and Bound	459
			Tabu Search	459
			Efficient GA	448
30	30	Large	Branch and Bound	1032
			Tabu Search	1032
			Efficient GA	1005

Table 2 represents experimental input. Using Table 1 data to 7× 5 small size problem; the proposed EGA, Tabu search and Branch and Bound algorithm give better sequence is J5 -J1- J4- J3- J2 -J7 -J6 and the total completion time of this sequence is 107 minutes. Table 3 illustrates that When the problem size is increased gradually, the total completion time for Tabu search and Branch and Bound algorithm grows up little by little. In large scale problem, the presented EGA gives optimal solution comparing with Tabu search and Branch and Bound algorithm. Table 3 and Figure 5 represent the makespan comparison result of Branch and Bound, Tabu Search and efficient GA. It is found that total completion time for efficient GA is reduced with the comparison of Tabu Search and Branch and Bound Algorithm.

We also see that our efficient genetic algorithm can minimize the waiting time between two jobs which can maximize our productivity.

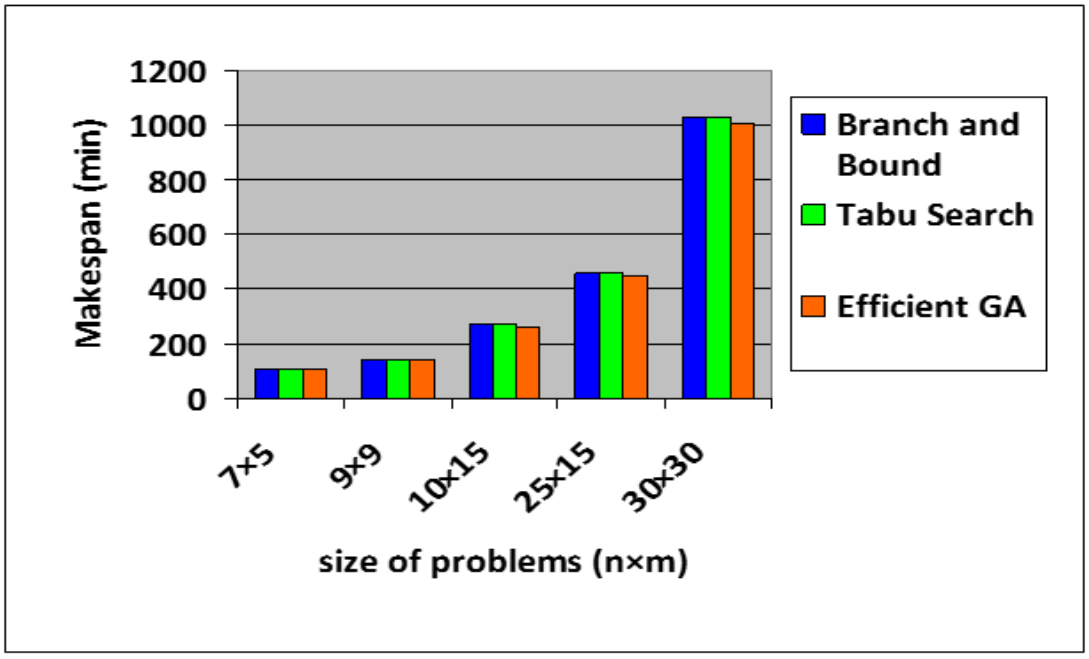


Figure 5: makespan comparison of Branch and Bound, Tabu Search and Efficient GA.

According to the results in table 4, it is found that the performance of the proposed EGA is considerably improved.

Table 4: Performance improvement of proposed EGA comparing with Tabu Search and Branch anBound in large size problems.

Number of jobs n	Number of machines m	Problem Size	Performance improvement of efficient GA comparing with Tabu Search and Branch and Bound
10	15	Large	3.31%

25	15	Large	2.39%
30	30	Large	2.62%

8. Conclusion

As the garments factory is increasing day by day in the Asian Subcontinent for economic growth, flow shop scheduling problems of increasing machines and jobs show the stiffness. Reducing this difficulty, we present EGA for n× m flow shop scheduling problem where n jobs are processed on m machines. Reducing idle time and waiting time is the main target of our presented efficient GA with the comparison of implemented Tabu Search and Branch and Bound Algorithm. In large scale problem, optimum sequence with minimum makespan is obtained for EGA but it is not suitable in large scale problem for Tabu Search and Branch and Bound Algorithm. Table 4 above shows that, in 10× 15 , 25×15 and 30×30 large size problems, the performance of the efficient GA is improved by 3.31%, 2.39% and 2.62% respectively comparing with Branch and Bound Algorithm and Tabu Search. So, the implemented application of our efficient genetic algorithm is the best from others. It can reduce total completion time of the tasks and can increase the quality of producing different products.

9. Recommendations

For future work, the proposed EGA can be extended. Further research should try to minimize the total completion time as much as possible and increase the overall productivity of manufacturing company. Completely eliminate the total idle time should be the target in our further experiment.

References

[1] Stafford, E. F., Tseng, F. T., Gupta, J. N. D. “Comparative evaluation of MILP flowshop models.” J.of the Ope. Res. Soc., vol. 56, pp. 88–101, 2005.

[2] Tseng, F. T., & Stafford, E. F. “New MILP models for the permutation flowshop problem.” Journal of the Operational Research Society, vol. 59(10), pp. 1373-1386, 2008.

[3] Mashhadi, M. M., Stafford, E. F., & Tseng, F. T. “Development of a new model for the flowshop problem.” in Industrial Engineering and Engineering Management, 2007 IEEE International Conference on IEEE, Dec. 2007, pp. 935-939.

[4] Yenisey, M. M., & Yagmahan, B. “Multi-objective permutation flow shop scheduling problem: Literature review, classification and current trends.” Omega, vol. 45, pp. 119-135, 2014.

[5] K. R Baker. Introduction to sequencing and scheduling. New York : wiley, 1974.

[6] AHG Rinnoy and Kan. Machine Scheduling Problems: Classification, Complexity and Computations. The Hague: Martinus Nijhoff, 1976.

[7] Yoshida and Hitomi, "Optimal two stage production scheduling with set up times separated." AIIETransactions, vol. II, pp. 261-263, 1979.

[8] Ahmad Pour Darvish and Heydari, "On flow shop scheduling problem with processing of jobs in a string of disjoint job blocks: fixed order jobs and arbitrary order jobs." JISSOR, vol. XXIV, pp. 1- 4, 2003.

[9] Singh, T.P., K, Rajindra & Gupta Deepak. "Optimal three stage production schedule the processing time and set up times associated with probabilities including job block criteria." in Proceeding of National Conference FACM, 2005, pp. 463-470.

[10] E.G, Coffiman. Computer and job Scheduling Theory. New York: Wiley, 1976.

[11] S.French. Sequencing And Scheduling:An Introduction to the Mathematics of the job shop. Chichester, UK: Ellis Horwood, 1982.

[12] Garey, M. R., Johnson and D. S. Computers and Intractability: A Guide to the Theory of NP-completeness. San Francisco: W.H. Freeman and Company, 1979.

[13] Lomnicki, Z.A. "A branch-and-bound algorithm for the exact solution of the three-machine scheduling problem." Operational Research Quarterly, vol. 16, pp.89-100, 1965.

[14] Gupta, D., Singla, P. and Bala, S. "Application of Branch And Bound Technique for n* 3 Flow Shop Scheduling with Breakdown Interval." International Journal for Engineering Research and Application, vol. 2, pp. 1675-1677.

[15] Osman I H and Potts C N, "Simulated annealing for permutation flow-shop scheduling." Omega, vol. 17, pp. 551-557. 1989.

[16] Widmer, M. and Hertz, A. "A new heuristic method for the flow shop sequencing problem." European Journal of Operational Research, vol. 41(2), pp. 186-193, 1989.

[17] Taillard, E. "Some efficient heuristic methods for the flow shop sequencing problem." European journal of Operational research, vol. 47(1), pp. 65-74, 1990.

[18] Glass, C. A., Gupta, J. N. D., Potts and C. N. "Two-machine Flow Shop Scheduling with Missing Operations: Special Cases and Approximation Algorithms Preprint OR75." Faculty of Mathematical Studies, University of Southampton, U.K, 1995.

[19] Leisten, R., and Kolbe, M. "A note on scheduling jobs with missing operations in permutation flow

shops." International journal of production research, vol. 36(9), pp. 2627-2630, 1998.

[20] Sadjadi, S. J., Aryanezhaad, M. B., Ziaee, M. "The general flowshop scheduling problem: mathematical models." J. of app. sci., vol. 8(17), pp. 3032-3037, 2008.

[21] Reeves, C. R. "A genetic algorithm for flowshop sequencing." Computers & operations research, vol. 22(1), pp. 5-13, 1995.

[22] Murata, T., Ishibuchi, H., & Tanaka, H. "Genetic algorithms for flowshop scheduling problems." Computers & Industrial Engineering, vol. 30(4), pp. 1061-1071, 1996.

[23] Etiler, O., Toklu, B., Atak, M., Wilson, J. "A genetic algorithm for flow shop scheduling problems." J. of the Ope. Res. Soc., vol. 55, pp. 830–835, 2004.

[24] Holland, J. H. Adaptation in natural and artificial systems. Ann Arbor: University of Michigan Press, 1975.

[25] Nowicki, E., & Smutnicki, C. "A fast tabu search algorithm for the permutation flow-shop problem." European Journal of Operational Research, vol.91(1), pp.160-175, 1996.

[26] Ben-Daya, M., & Al-Fawzan, M. "A tabu search approach for the flow shop scheduling problem." European journal of operational research, vol. 109(1), pp. 88-95, 1998.

[27] Solimanpur, M., Vrat, P., Shankar, R. "A neuro-tabu search heuristic for the flow shop scheduling problem." Comp. & Ope. Res., vol. 31, pp. 2151-2164, 2004.

[28] Ignall, E., & Schrage, L. "Application of the branch and bound technique to some flow-shop scheduling problems." Operations research, vol. 13(3), pp. 400-412, 1965.

[29] Ladhari, T, Haouari, M, "A computational study of the PESP based on a tight lower bound." Computers & Operations Research, vol. 32, pp. 1831-1847, 2005.