Bee Colony Optimization Solution to Single Machine Just in Time Scheduling Problem

V. C. Mahavishnu, A. N. Senthilvel, and S. Umamaheswari

***Abstract*—Objective of this paper is to create optimized Schedule for a set of jobs on a single machine which minimize the total early and tardy penalty. No preemption of jobs is allowed while scheduling. During the process of scheduling, no machine idle time is allowed and the jobs are assumed to be continuously available. This problem of reducing the machine idle time is an NP class problem. The proposed bee schedule algorithm uses the swarm intelligence approach to solve the single machine early/tardy scheduling problem. The solution obtained using the proposed algorithm contains a local selective search procedure to further improve the schedule. This helps to fine tune and achieve more optimal solution to this scheduling problem. The pairwise interchange procedure is also implemented on the schedule obtained in several passes till the range value is obtained, thus attaining the best solution to the problem. The proposed algorithm is tested and found that it outperforms the existing algorithms like dynamic programming and genetic algorithms. Experimental analysis proves that the proposed algorithm is suitable for similar problems also.**

***Index Terms*—Bee schedule algorithm, early/tardy scheduling, NP class problem, single machine scheduling.**

I. Introduction

In this paper, a single machine scheduling problem is considered where the jobs are supposed to be completed within due time, otherwise they may incur early/tardy penalty. The Scheduling based on both the earliness and tardiness penalties are needed in many real life problems. Production activities in a manufacturing plant or scheduling of the aircrafts that are waiting for the landing and take-off process, etc. are few examples for the earl/tardy scheduling problem.

Consider the example of scheduling the production of products in a manufacturing plant where the goods produced have holding costs and goods may spoil by time when the products are produced in advance. Also late delivery leads to loss of loyal customers and the shipping costs are also increased. In such places the just in time production of goods is needed. The early/tardy scheduling deals with these just in time scheduling where the jobs are to be scheduled with the due date constraint and least penalty value, Conway (1967) [1]. The machines and the jobs are available continuously for processing. The early tardy scheduling problem is that given a set of n independent jobs $J_1, J_2 \ldots J_n$, each of which has to be processed on a single machine. The preemption of jobs is not allowed. Each job J_j has a processing time p_j and the job should be finished exactly on its due date d_j. The early and the tardy penalties of job J_j are ep_j and lp_j respectively.

The earliness E_j of a job J_j can be defined as $E_j = \max(0, d_j - C_j)$ and tardiness T_j of the job J_j can be defined as $T_j = \max(C_j - d_j, 0)$, where C_j is the completion time of the job J_j. The main idea of the Early/Tardy Scheduling Problem (ETSP) is to find a schedule with the minimum total penalty.

$$\text{Total penalty} = \sum_j^n (ep_j E_j + lp_j T_j) \quad (1)$$

The early/tardy scheduling problem enforces the condition that the machine is available continuously for the processing of the jobs i.e., a machine can be idle only when there is no jobs waiting for processing. This suits those production environments where the demands are high or operating and startup costs of the machine exceed the cost of producing some jobs early, Conway (1967).

For the ETSP, there are many procedures to find the schedules have been proposed in the recent years. Among these procedures Li (1977) [2] used the branch-and-bound methods to find the exact solution based on the Lagrangian relaxation procedure. Abdul-Razaq and Potts (1998) [3] proposed the exact solution using Dynamic programming which is based on state-space relaxation for single-machine scheduling. Tanaka (2007) [4] proposed a solution based on the successive sublimation technique with relaxations using dynamic programming.

The heuristic approaches also provide a solution to the ETSP where the filtered beam search procedure was used by Ow and Morton (1989) [5]. The filtered beam search procedure was very slow for the scheduling of more than 50 jobs. Heuristic procedure based on neighborhood search that is better than filtered beam search procedure in terms of both solution quality and running time. The greedy heuristics with some improvements to reduce the total costs were proposed by Valente and Alves (2005) [6].

The Genetic algorithms were used to solve this type of scheduling problem. There were different genetic algorithms of which Valente *et al*. (2006) [7] proposed twelve variants of a hybrid genetic algorithm. All these twelve genetic algorithms were from the current generation, where the best 10% of the population unaltered to the next generation. No mutation operator was used in these algorithms. These genetic algorithms employ a local search in a single pass or up to eight passes of an adjacent interchange procedure. During each pass, the adjacent interchange procedure starts with the first job in the schedule and interchanges two adjacent jobs whenever doing so reduces the cost of the schedule. The dispatching procedure and the NSearch algorithm were also used with genetic algorithms. These methods were faster than the previous approaches mentioned

Manuscript received March 19, 2014; revised April 28, 2014.

The authors are with the Department of Computer Science and Engineering at Coimbatore Institute of Technology, Coimbatore, India (e-mail: mahavishnu.vc@gmail.com, senthilvel.cit.cse@gmail.com, umamaheswari@cit.edu.in).

above. Pan *et al*. (2006) [8] presented a discrete particle swarm optimization algorithm for minimizing the total earliness and tardiness penalties. The problem with common due dates for all jobs is considered on a single machine for the optimization.

Swarm intelligence (SI) is a type of artificial intelligence based on the collective behavior of decentralized, self-organized systems. Over the past twenty years, a number of studies have addressed the collective behavior of animals in a fixed social hierarchy. Kennedy, Kazemi, Pedersea have described the parameters and described the swarm intelligence and particle swarm optimizations (PSO) [9]-[11].

These studies have led to the development of various distributive collective problem solving strategies. More formally, Engelbrecht defines Swarm Intelligence (SI) as the property of a system whereby the collective behaviors of unsophisticated agents, which interact locally with their environment, cause coherent functional global patterns to emerge. These concepts include emergence, social intelligence and adaptation, Shi and Beernaert (2001) [12]. Sha. and Hsu [13] has proposed a hybrid algorithm based on the PSO to solve the scheduling problem.

A swarm intelligence technique based on the collective behavior and self-organization characteristics of swarm like honey bees were somewhat similar to this ETSP. Based on this foraging behavior of honey bee swarm, Karaboga (2005) [14] proposed the Artificial Bee Colony (ABC) algorithm. This algorithm was very simple and was used mostly for many scheduling problems. Tasgetiren and Pan *et al*. (2011) [15] describes an application of ABC algorithm for solving the flow shop scheduling problem. ABC algorithm is fine tuned to meet the requirement of the ETSP, in which a local selective search is used to further improve the schedules obtained through the algorithm. This will help us to find a more optimal solution to this scheduling problem. The pairwise interchange procedure is implemented on the schedule obtained to minimize the total Early and Tardy penalties.

This paper is organized as follows: Section II provides details about the bee colony optimizations; Section III describes the procedures used for solving the ETSP. The Computational results are analyzed in the Section IV and Section V contains the conclusions from the proposed algorithm.

II. Bee Colony Optimization

The bee colony optimization is a type of particle swarm optimization techniques. The optimizations are based on the nectar collecting technique used by the honey bees. The honey bees in a hive work as three category or groups i.e., scout bees, employed bees and the onlooker bees. Scout bees are the ones involved in the process of searching food sources around the hive in a random manner. They continue this process of searching the food until the source with the food is found. Once the food source is found out, the scout bees re-categorize themselves to the category of employed bees. The employed bees work to get the food from the source to the hive. The employed bees act as a communication agent to the onlooker bees. The employed bees perform a wangling dance which helps the onlooker bees to find out the food source and the amount of food in the source. The onlooker bees wait in the dance area and watch many dances by the different employed bees and select the food source with abundant food. The probability that the onlooker bees select a food source is proportional to the amount of the nectar in that food source. Once an onlooker bee selects a food source, it is re-categorized as an employed bee. Once the food in the source is exhausted, the employed bees are re-categorized as scout or onlooker bees.

Based on this behavior of honey bees, Bee colony optimization algorithms were proposed. Further this concept was extended for the selection and permutation problems. The solution to the problem is considered similar to the food source and the probability of the solution to be the best solution is similar to the amount of food in the source. The bee colony algorithms have a fixed number of solutions at the start and they are all random solutions. An exhaustive search is made until the optimal solution is obtained. In every phase of the algorithm, a new solution is searched in the neighborhood and if the solution is better continue the algorithm with that schedule. The phases, stop criteria and the solutions are dependent on the problem definition.

The probability that a neighborhood solution is selected for the next stage depends upon the probability that the neighborhood solutions lead to the optimal solution. The selection procedure is based on the roulette wheel selection procedure. The solution which will lead to the global optimum is selected and further processed. If a solution is not improved by selecting the food source for a predetermined number of times, the algorithm is stopped and the solution is declared to be the best solution but not the optimal one. Another random solution is created and proceeded to get the new solution that may lead to the global optimal solution. Based on many survey by akay and karaboga (2009) [16], it is found that the bee swarm intelligence is more advantage to the other existing algorithms.

III. Bee Schedule Algorithm

The proposed bee schedule algorithm works mainly in six stages. The input to the algorithm is a file containing the input details for the algorithm. The job ID, job processing time, early penalty, late penalty and the due dates for the several jobs in that file. The first step is the initial random solution generation from which the further process proceeds. This step is done based on the first come first serve ordering.

The next step is the creation of a random nearby solution where the solution is obtained by the ordering of the jobs with the odd job id first criteria. The next stage is the selection method where we decide to do the selective-multi-point insertion where the jobs are placed at positions where the jobs will incur less penalties and then the alternate point swap is done or the algorithm directly flows to the alternate point swap procedure without doing the selective multi point swap when the solution is not improved. At this stage of alternate point swap, a schedule is obtained which is further improved by the local selective search procedure. The pairwise interchange procedure is applied to the improved schedule to get the global optimal schedule. The flow of the algorithm for solving the early/tardy scheduling problem is shown in Fig. 1.

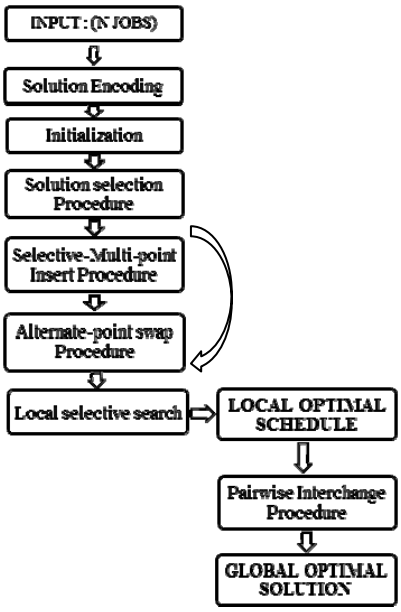

Fig. 1. Bee schedule algorithm.

A. Real Time Queue

The early/tardy scheduling problem is a type of combinatorial optimization problem. Hence the first come first serve basis is considered for the generation of the first schedule. The above scenario is analogous to areal time queue. This schedule denotes the ordering of execution of the jobs. The early penalty and the late penalty are calculated for the schedule ***s***. This schedule ***s*** is saved for the further processing. The next step is to find a random nearby solution.

B. Neighboring Solution

The schedule based on the first come first serve basis is considered and the random nearby solution is generated. In this process of the random generation, the random schedule is changed by sequencing the jobs with the criteria that process the odd jobs first. The schedule s obtained in the previous stage is taken and a new empty schedule s* is created. Based on the criteria the jobs are taken from the schedule s and inserted into the schedule s*. The odd jobs are followed by the even jobs and a new schedule is created. This solution is considered to be Neighboring solution. This process is repeated until all the jobs in the schedule s are taken completely and inserted into s*.

There are two different schedules that are based on the given input. The selection of one of the schedules is to be done at this point to proceed to the next stage of the algorithm. The schedule with the higher probability to get an optimized solution is chosen and the next step in the algorithm is started with this solution. The problem structure of the early/tardy scheduling problem is a type of permutation problem. Permutation of the n jobs is to be done to get the n! different combinations of the neighboring solution. The procedures like the selective-multi-point insert and the alternate point swap are used along with the local selective search and the pairwise interchange to get the neighboring solutions.

C. Selective-Multi Point Insert Procedure

The selective-multi-point insert procedure to find the neighboring solution uses the parts of the schedule previously obtained. This procedure focuses on selecting points from the previous schedule where the jobs placed at those points produces good results. There is a high probability of producing good results by placing these points in our new solution. To implement this procedure, the previous schedule obtained in the previous stage is taken and a new empty schedule is created. The best points from the previous schedule are copied to the same position in the new schedule. The total number of points to be copied from the previous schedule is determined by the range value. The range value is selected by a random value from 1 to 10. This range value determines the percentage of the number of jobs that are to be copied from the previous schedule to the next schedule. All points except those inserted from the previous schedule are empty. In order to fill the empty slots in the schedule, the remaining jobs are sorted and arranged in the empty locations. Thus the schedule is complete and the total penalty for this schedule can be created from (1). This schedule is used for the alternate point swap procedure. There are certain cases where the schedule created by the multi-point insert procedure will be equal to the previous schedule. In such scenarios the multi-point insert procedure is avoided to reduce the complexity.

D. Alternate Point Swap Procedure

This procedure is important because this helps to reach the global optimal solution. The previous procedures concentrate on the local optimal solution. The initial step in the alternate point swap procedure is to copy the initial solution and select alternate points in random from the schedule. Considering that the alternate points selected are x,y and z, and the two swap operations are to be carried out with these alternate points. Initially the points x and y are swapped. The new job at the position x is swapped with the job in the position y. This procedure of swapping two times using the alternate points is more effective compared to the single swap. Thus the alternate point swap procedure will lead us to the global optimal solution.

The solution obtained by the alternate point swap is more close to the global solution. This schedule is further improved by applying a local selective search with the help of the alternate point swap procedure and pairwise interchange procedure.

E. Local Selective Search Procedure

The quality of the schedule obtained in the alternate point swap is improved by the multiple passes of the local selective search procedure. This is an exhaustive search where the solution of the alternate point swap is inserted at the various possible combinations of the alternate points that are chosen to create a new schedule. The stop criteria used for the local selective search procedure is when the schedule is not improved for five consecutive searches. The local selective search procedure is more effective when the alternate points chosen randomly are swapped after selecting the best combination by permutation of the alternate points.

F. Pairwise Interchange Procedure

Improved schedule obtained using the local selective search procedure is further improved with the pairwise interchange procedure. The pairwise interchange procedure swaps the job in the schedule with the adjacent one. The new

schedule does not totally deviate from the original schedule and the interchanging positions will help to improve the schedule and thus achieving the global optimal solution.

The algorithm does not stick to a single schedule and improve that schedule, but it choses various random schedules and selects the best schedule and optimizes it. Thus this algorithm is more advantageous than the other procedures available for this problem.

IV. Computational Analysis

The proposed algorithm was compared with the existing solutions of the dynamic programing (DP), heuristic approaches and the genetic algorithms (GA). The input file to the algorithm consists of a set of jobs with the job id, processing time pj and the early penalty epj and tardy penalty lpj and the due date dj. The input datasets were comma separated files having instances with 100, 500, 750or 1000 jobs each.

TABLE I: Comparison of BSA with Other Algorithms

Algorithm	100 Jobs		500 Jobs		750 Jobs		1000 Jobs	
	APD	ACT	APD	ACT	APD	ACT	APD	ACT
SS-GA	0.06	3.6	0.08	29.2	0.05	37.2	0.06	49.3
SS-GA-MNAI	0.05	3.7	0.05	28.4	0.02	36.9	0.03	59.2
DP	0.03	4.6	0.07	26.8	0.02	42.5	0.04	54.2
BSA	0.01	3.8	0.02	20.9	0.01	25.7	0.00	41.9

Table I describes the average percentage deviation (APD) and the average computation time (ACT) for the various algorithms. The overall penalty values obtained with the proposed algorithm were compared with dynamic programming and the genetic algorithms to get the average percentage deviation. The APD is calculated by comparing the results of instances with the best known or the optimal solutions as done by Valente *et al*. The results prove that there is a significant reduction in the overall penalty values with the proposed algorithm. The proposed algorithm works well with all types of the input jobs given to the algorithm and APD values proves that the proposed algorithm produces better results compared to the existing algorithms.

The overall penalty values obtained with the proposed algorithm were compared with the dynamic programming and the genetic algorithms. The results prove that there is a significant reduction in the overall penalty values with the proposed algorithm. The Fig. 2 describes the results of the analysis.

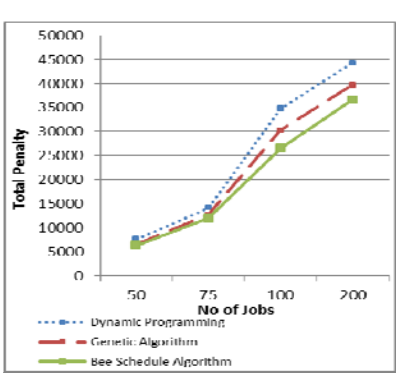

Fig. 2. Total Penalty Analysis.

The experimental results show that the proposed algorithm has improved the schedule significantly compared to the dynamic programming and also better than the genetic algorithm. Bee schedule algorithm has good results for the instances with 100 or more jobs. Considering the complexity of the algorithm as a parameter for comparison the dynamic programming has an exponential complexity whereas the proposed algorithm has a polynomial complexity. Thus the proposed algorithm is better compared to the existing algorithms in terms of the complexity also. Although the proposed algorithm produces good results for most inputs, they may not be the optimal solution always.

V. Conclusions

The bee schedule algorithm was tested against the existing algorithms to solve the early/tardy scheduling problem. Performance of the bee schedule algorithm was proved better compared to other existing methods. Results were analyzed with instances having a maximum of 500 jobs. The results also show that the bee schedule algorithm can also be used to solve all combinatorial optimization problems and similar problems also.

In future work, we would like to extend this bee schedule algorithm to the other real time optimization problems like rail route optimizations with timing constraints, laying pipelines by cutting minimum number of roads, providing Automated Teller Machines(ATM) in many locations with the security constraints.

References

[1] W. R. Conway, W. L. Maxwell, and L. W. Miller, *Theory of Scheduling,* Dover Publications, 1967.
[2] G. Li, "Single machine earliness and tardiness scheduling," *European Journal of Operational Research*, vol. 96, 1977, pp. 546–558.
[3] T. S. A. Razaq and C. N. Potts, "Dynamic programming state-space relaxation for single-machine scheduling," *Journal of the Operational Research Society*, vol. 39, 1988, pp. 141–142.
[4] S. Tanaka, "An exact algorithm for single-machine scheduling without idle time," in *Proc. the third Multidisciplinary International Scheduling Conference*, MISTA 2007, pp. 614–617.
[5] P. S. Ow and T. E. Morton, "The single machine early-tardy problem," *Management Sciences*, vol. 35, 1989, pp. 177–191.
[6] J. M. S. Valente and R. A. F. S. Alves, "Improved heuristics for the early/tardy scheduling problem with no idle time," *Computers and Operations Research*, vol. 32, pp. 557–569, 2005.
[7] J. M. S. Valente, J. F. Gonçalves, and R. A. F. S. Alves, "A hybrid genetic algorithm for the early/tardy scheduling problem," *Asia-Pacific Journal of Operational Research*, vol. 23, pp. 393–405, 2006.
[8] Q. K. Pan, M. F. Tasgetiren, and Y. C. Liang, "Minimizing total earliness and tardiness penalties with a common due date on a single-machine using a discrete particle swarm optimization algorithm," *Lecture Notes in Computer Science*, vol. 4150, pp. 460–467, 2006.
[9] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proc. the IEEE International Conference on Neural Networks*, 1995, pp 1942-1948.
[10] A. L. Kazemi and C. K. Mohan, "Discrete multi-phase particle swarm optimization," *Information Processing with Evolutionary Algorithms, Berlin*, Germany: Springer, 2006, pp. 306-326.
[11] M. E. H. Pedersea, "Good parameters for particle swarm optimization," Technical Report, Hvass Laboratories, 2010.
[12] Y. Shi and R. C. Eberhart, "Fuzzy adaptive particle swarm optimization" in *Proc. IEEE International Conference on Evolutionary Computing*, vol. 1, 2001, pp. 101-106.
[13] D. Y. Sha and C. Hsu, "A hybrid particle swarm optimization for job shop scheduling problem," *Computers and Industrial Engineering*, vol. 51, no. 4, pp. 791-808, Dec 2006.

[14] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Technical Report TR06, Computer Engineering Department, Erciyes University, Turkey, 2005.
[15] M. F. Tasgetiren, Q. K. Pan, P. N. Suganthan, and T. J. Chua, "A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem," *Information Sciences*, vol. 181, 2011, pp. 2455–2468.
[16] B. Akay and D. Karaboga, "A survey: algorithms simulating bee swarm intelligence," *Artificial Intelligence Review*, vol. 31, 2009, pp. 61–85.

V. C. Mahavishnu was born in India, He received his B.E. degree from Anna University Chennai in 2012 and the M.E. degree from Anna University Chennai in 2014, both in computer science and engineering. He is currently an active researcher planning for his Ph.D. degree. His research interests include image processing, hadoop, cloud computing, and scheduling.

A. N. Senthilvel is working as an assistant professor (SG) in the Department of Computer Science and Engineering at CIT, Coimbatore, India. He is currently pursuing his Ph.D. degree in the area of scheduling.

S. Umamaheswari is working as an assistant professor (SG) in the Department of Electronics and Communication Engineering at CIT, Coimbatore, India. Many researchers are being guided by her in various disciplines.