International Journal of Computational Science
1992-6669 (Print) 1992-6677 (Online) © Global Information Publisher
2007, Vol. 1, No. 2, 215-223

Single Machine Scheduling for Jobs with Individual Due Dates and a Learning effect: Genetic Algorithm Approach

Pei Chann Chang [1], Shih-Hsin Chen [2] V.Mani [3]

[1] Department of Information Management, Yuan Ze University, 135, Yuan-Dong Road, Tao-Yuan, Taiwan, 32026, R.O.C.
{iepchang}@saturn.yzu.edu.tw
[2] Department of Industrial Engineering and Management, Yuan Ze University, Tao-Yuan 32026, Taiwan, R.O.C.
{s939506}@mail.yzu.edu.tw
[3] Department of Aerospace Engineering, Indian Institute of Science, Bangalore India 560-012.
{mani}@aero.iisc.ernet.in

Abstract. In this paper, we present a genetic algorithm approach that considers the single machine scheduling problem. There are n jobs, in which each job j ($j = 1, 2, ..., n$) has a normal processing time p_j, a due date d_j, earliness penalty α_j, and tardiness penalty β_j. The objective is to find the sequence of jobs that minimizes the weighted sum of earliness and tardiness penalty costs, with a learning effect. The machine idle times are not considered. The worker involved in doing the same operations on a machine, learns the task and the worker become efficient in that job. Thus, in scheduling problems the job processing time depends on the position of the job in a sequence. This is the learning effect. The problem of finding the optimal sequence of jobs is a difficult combinatorial optimization problem. It can be easily seen that there are $n!$ sequences are possible for this problem. Because of the difficult nature of this problem, we use genetic algorithm to obtain the best/optimal sequence of jobs. Various issues related to genetic algorithm such as solution representation, selection methods, and genetic operators are presented. We show via numerical examples (test problems) that the genetic algorithm approach takes only small computation time to solve fairly large problems (50 jobs).

Keywords: scheduling, single-machine, learning effect, genetic algorithms.

* Corresponding Author. Tel: 886-3-4636165, Fax: 886-3-4635319. Email: iepchang@saturn.yzu.edu.tw.

1 Introduction

Single-machine scheduling problem is one of the well known problems studied by many researchers in the past fifty years. In most of these studies, the processing time of a job is assumed to be a constant. Or in other words, the processing time of a job is independent of its position in the sequence. In practical situations, this assumption may not be true. This is due to the fact that workers ability to learn when they are processing similar tasks. Because of the learning effect, the processing time of a job depends on its position in the sequence. The concept of "learning effect" is known in management literature.

The learning effect can arise in scheduling of jobs due to the fact that workers are processing same type of jobs on the same machine. So it is easy for workers to improve their performance and the processing time of a job will reduce because of the learning. The learning effect, for the case of single machine scheduling problem is first considered in [1]. In this study, the objective is to minimize the deviation from a common due date, in presence of learning effect. Another objective considered is the minimization of flow time [2]. In both the studies [1] and [2], the optimal sequence of jobs is obtained by solving an assignment problem. The learning effect in a two machine flowshop scheduling is presented in [3]. The objective is to find a sequence of jobs that minimize the total completion time in presence of learning effect. A branch and bound technique is used for the solution. To speed up the computation several dominance properties are derived. A heuristic algorithm is also introduced in [3], to improve the efficiency of the branch and bound technique. In these studies [1,2,3], the importance of considering the learning effect in production scheduling is discussed.

The learning effect for the problem of flow time minimization on parallel identical machines, is considered in [4], and a polynomial time solution is presented. The problem of flowshop scheduling with a learning effect is also studied in [5], and is shown that the classical Johnson's rule is not the optimal solution to minimize makespan. Single-machine scheduling problem with a learning effect to minimize the number of tardy jobs is discussed in [6]. Various issues related to scheduling problems with a learning effect, is also studied in [7,8,9].

It is shown in [10] that the problem of minimizing the total tardiness on one machine is NP-hard. Another related research [11] that consider the case in which the processing times decrease in a piecewise linear fashion, which is a good approximation to study the learning effect. A survey of scheduling with time dependent processing times presented in [12], provide a framework to show how the time-dependent processing time problems have been generalized from the classical scheduling theory.

In this paper, we consider the single machine scheduling problem, in which, there are n jobs, and each job j ($j = 1, 2, ..., n$) has a normal processing time p_j, a due date d_j, earliness penalty α_j, and tardiness penalty β_j. The objective is to find the sequence of jobs that minimizes the weighted sum of earliness and tardiness penalty costs, with a learning effect. The machine idle times are not considered. This is a difficult combinatorial optimization problem [10]. It can be easily seen that there are $n!$ sequences are possible for this problem. Because of the difficult na-

ture of this problem, we present a genetic algorithm approach to obtain the optimal/best sequence. Genetic Algorithm (GA) is the most well-known technique for solving combinatorial optimization problems. Genetic algorithms developed in [13] attempts to design artificial systems based upon the adaptive process of natural systems. A complete description about genetic algorithms is available in [14,15,16]. A review of application of genetic algorithms in production and operations management is given in [17].

This paper is organized as follows. In the next section, we describe the scheduling problem considered in our study with a learning effect. In Section.3, we present a brief description of genetic algorithm to this scheduling problem. In Section.4, we present the numerical results obtained for the problems, using a genetic algorithm. A conclusion is presented in the last section.

2 Single-machine scheduling with a learning effect

We consider the single machine scheduling problem, in which, there are n jobs, and each job j ($j = 1, 2, ..., n$) has a normal processing time p_j , a due date d_j , earliness penalty α_j , and tardiness penalty β_j . This type of scheduling problems has been studied in [18,19,20,21]. The objective is to find the sequence of jobs that minimizes the weighted sum of earliness and tardiness penalty costs, with a learning effect. The machine idle times are not considered. The sequence is the order in which the jobs are processed in the machine. All the n jobs available for processing at time $t = 0$. Because of the learning effect, the processing time of a job depends on its position in the sequence. Hence, the processing time of the jobs are given as

$$p_{jr} = p_j r^a \tag{1}$$

Here p_{jr} is the processing time of job j , if it is in position r of the sequence, and a is the learning index ($a < 0$). This above equation is introduced in [1] to include the learning effect. We can see from the above equation, the processing time of a job decreases as function of the position in the sequence. The value of a is obtained from learning curves. An analysis of learning curves in machine shops is discussed in [22]. The objective is to find the sequence of jobs that minimizes the weighted sum of earliness and tardiness penalty costs, with a learning effect. Hence, the objective function for our problem is

$$Minimize \quad \sum_{j=1}^{n} \{\alpha_j E_j + \beta_j T_j\} \tag{2}$$

Let C_j is the completion time of job j . The earliness and tardiness of job j are given as

$$E_j = Max\{0, (d_j - C_j)\} \tag{3}$$

$$T_j = Max\{0, (C_j - d_j)\}. \tag{4}$$

In this paper, we consider this above problem with a learning effect and present a genetic algorithm to obtain the optimal/best sequence.

3 Genetic algorithm for Single-machine scheduling with a learning effect

Genetic Algorithms (GA) is a search algorithm based on the mechanism of natural selection that transforms a population (a set of individuals) into a new population (i.e., next generation) using genetic operators such as crossover, mutation and reproduction. A survival of fittest strategy is adopted to identify the best strings and subsequently genetic operators are used to create a new population for next generation. More details about how genetic algorithm works for a given problem can be found in literature [14,15,16].

In genetic algorithms, the search space contains all search nodes for a given combinatorial optimization problem. GA starts with an initial population of search nodes from the search space. Each search node in the population is evaluated using the objective function and a fitness value is assigned to each search node. New search nodes are generated for next generation based on fitness value and applying genetic operators to the current search nodes. This process is continued for generation after generation until the algorithm converges. To apply genetic algorithm to our scheduling problem with learning effect, we need to address the following factors.

- string representation of search nodes
- population initialization
- selection function and genetic operators
- fitness function
- termination criterion

String Representation: The string representation is the process of encoding a sequence for the scheduling problem. The string representation scheme depends on the structure of the problem in GA and also depends on the genetic operators used. For our single-machine scheduling problem, the sequence is a string of integers. The integers corresponds to job numbers. For example, the string $\{4\ 3\ 5\ 1\ 2\}$, in our problem represents the sequence in which the jobs are processed, when there are 5 jobs. In general, the length of the string is equal to the number of jobs (n). We know that for n jobs, there are $n!$ job sequences. We can see that in this string representation all the possible ($n!$) sequences are represented and they are unique.

Population Initialization: In genetic algorithm based solution approach, initial sequences (population) are generated using random generation procedure. The population size, and the method of obtaining initial solutions will affect the convergence of the problem. Genetic algorithms iteratively improve the sequences, hence if the initial sequences are good sequences the convergence of the problem will be faster. The population size is problem-dependent and has to be determined through numerical simulation.

Genetic Operators: In genetic algorithms, the basic search mechanism is done by genetic operators. Genetic operators are used to create new sequences based on existing sequences in the population. Crossover, mutation and reproduction are the commonly used genetic operators.

Crossover Operation: Crossover operation uses two sequences (search nodes) to produce two new sequences. During crossover operation the parents exchange parts of their solutions. The idea is to combine the good parts of solutions of the parents to produce offsprings. A number of crossovers for combinatorial optimization problems is given in [16]. They are single point crossover, two point crossover, uniform crossover and partially mapped crossover. In our study, we use two-point point crossover.

Mutation Operation: Mutation operation works on a single sequence to produce a single new sequence. This operation is needed to ensure diversity in the population and to avoid the premature convergence and local minima. In our study, we have used swap mutation presented in [16].

Fitness Function: Fitness is the driving force in genetic algorithms. In our scheduling problem, fitness function assigns a fitness value to each of the sequences in a generation. Fitness function must be capable of evaluating every sequence in the search space. For our problem the search space contains all $n!$ possible sequences. Genetic algorithm does not know anything about the problem domain or fitness function. The only information used in the execution of genetic algorithm is the observed value of fitness for the sequences present in the population. Genetic algorithm is guided by the fitness value to search for the most efficient sequence for the problem. The fitness function for our problem is the objective function, and is minimization of the sum of the deviations. The objective function for our problem is given in (2). The genetic operators will try to maximize the fitness function for a maximization problem. Our scheduling problem is a minimization problem and so needs a transformation of the fitness value. For this purpose, we refine our selection criterion which selects a solution with lower fitness value (objective function value). The selection method used in our scheduling problem is explained below.

Selection function and Elitism Strategy: In genetic algorithm, the selection of a sequence from the existing population (sequences) plays an important role. The idea is that better sequence (sequence with better fitness value) should have a better chance of being selected for genetic operations to produce new sequences. In genetic algorithms there are several selection methods such as roulette wheel selection and its extensions, scaling techniques, elitist models and ranking methods are presented. In our study, we have used binary tournament selection method. The criterion of binary tournament selection is to randomly select two sequences (solutions) and compare their fitness values. The solution with better fitness is selected for maximization problems. Since, the scheduling problem is a minimization problem, solution with a lower fitness is selected. The elitism strategy is to select a number of elite solutions from external archive. The number of elite solutions depend on the population size and elitism rate.

Termination Function: In a genetic algorithm, in each generation, sequences are selected on the basis of their fitness and subject to genetic operators to produce new sequences for the next generation. The evolution process of successive generations continues until a termination criterion is satisfied. The most frequently used stopping criterion are population convergence criteria and a

specified maximum number of generations. Population convergence criteria for our problem is that all the sequences in the population are the same in two successive generations. This sequence is the best or optimal sequence to the problem. Another stopping criteria is to stop the evolution process when the maximum number of generations is reached. The best sequence is the one in the population with maximum fitness value.

4 Simulation Results

The genetic algorithm to obtain the sequence for single-machine scheduling problem with a learning effect is tested with known problems, and with standard test problems from literature.

It is known that the solution obtained from genetic algorithms, for combinatorial optimization problems, can not be guaranteed to be optimal; i.e., no formal proof of optimality. But, for a large class of difficult combinatorial optimization problems, it has been shown that the genetic algorithm produces solution that are close to the optimal or among the best available solutions [16]. So in our studies, we call the sequence obtained from genetic algorithm as “best" sequence instead of optimal sequence. The genetic algorithm to obtain the sequence, for single-machine scheduling problem with a learning effect is implemented in Java on a Pentium-IV machine. The genetic algorithm used the following parameters given in Table.1 in all our simulations.

Table 1. Parameters used in simulation

Parameters	Description	Value
P_m	Mutation probability	0.30
P_c	Crossover probability	0.50
M	Maximum number of generations	100
N	Population size	100

The following steps are carried out in a genetic algorithm for single-machine scheduling problem with a learning effect.

1. **STEP.1**: Initialization: An initial population (N) sequences are randomly generated.
2. **STEP.2**: Evaluation: The fitness for each of the sequence in the population is calculated according to the fitness function.
3. **STEP.3**: Perform selection function using binary tournament selection method and elitism strategy, to select sequences for genetic operations.
4. **STEP.4**: Genetic Operations: Perform crossover and mutation operations based on probability of crossover and mutation. Here we may get more sequences than the population size (N). Perform reproduction using elitist model to obtain N best sequences.
5. **STEP.5**: Now, we have the best N sequences from previous step. Repeat steps 2, 3, and 4, until the algorithm converges.

The advantage with genetic algorithm is that it starts with a random sequences and modify the sequences in successive generations, and the best sequence is obtained. The only information used is the fitness value.

For our problem, this genetic algorithm is used to obtain the sequence of jobs that minimizes the weighted sum of earliness and tardiness costs, with a learning effect. Machine idle times are not considered. For our problem, we have considered three values of $\alpha = -0.152, -0.322,$ and -0.515, which corresponds to 90%, 80% and 70% learning curves respectively.

For the single machine scheduling problem is considered in this study, many test problems are provided with job dependent earliness and tardiness penalties in [23]. These test problems are generated in the following manner and are available in the internet. For a given value of n (number of jobs), the processing times are generated randomly from the uniform distribution $U = [10,100]$. The due dates d_j are generated from $U = [d_{min}, d_{min} + \rho P]$, where $d_{min} = Max\{0, P(\tau - \rho/2)\}$ and $P = \sum_{j=1}^{n} p_j$. The two parameters τ and ρ are tardiness and range parameters. These test problems are available in [23] for $n \in \{20,30,40,50\}$, $\tau \in \{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$ and for the value of $\rho \in \{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$. The earliness ($\alpha_j$) and tardiness ($\beta_j$) for all jobs are generated randomly from the uniform distribution $U = [1,5]$.

We have considered the situation when the number of jobs is 20, 30, 40 and 50, with a learning effect. These problems are taken from [23], and the problem numbers are $sks222$, $sks322$, $sks422$ and $sks522$. Genetic algorithm is able to obtain the “best" sequence in a very short time. The objective function value and the computation time required by our genetic algorithm is given below in Table.2 for learning rates $\alpha = -0.515$, $\alpha = -0.322$, and $\alpha = -0.152$. In our simulations, for each of these problems, the genetic algorithm was run 20 times and the average computation time is given in Table.2. Also, the minimum, mean and maximum value of the objective function value obtained from these 20 runs are given in Table.2. From this Table.2, we see that our genetic algorithm approach, takes only a small computation time to obtain the best sequence. The source code for the genetic algorithm and the results obtained are given in the website [24].

The parameters used in genetic algorithms $\boldsymbol{P_m}$, $\boldsymbol{P_c}$, and $\boldsymbol{N}$ are usually determined by trial-and-error. We have tried with different values of crossover probability ($\boldsymbol{P_c}$), and different values of population size ($\boldsymbol{N}$). We obtain the same optimal sequence in our simulation and the change in computation time is not significant.

We have also used branch and bound technique for this problem, and noticed that the computational time depends on the value of ρ and τ. This is because of the number of nodes in the branch and bound technique. For example, when τ=0.2 and ρ=0.2, the branch and bound computation time for 50 jobs is 48.38, and the computation time for genetic algorithm is 3.5 seconds. It is important to note that the branch and bound technique obtains the optimal sequence, but the sequence obtained from genetic algorithm can not be guaranteed to be optimal; i.e., no formal proof of optimality. But, for a large class of combinatorial optimization problems, it is shown that the genetic algorithm produces solution that are optimal or close to optimal solution.

Table 2. Objective function and computation time for different Learning rates

Learning rate	number of jobs	Minimum	Objective function value Mean	Maximum	Computation time
α = -0.515	20	743.16	753.14	792.43	0.99223
α = -0.515	30	1260.90	1317.40	1369.90	1.58460
α = -0.515	40	1351.50	1358.00	1379.00	2.37970
α = -0.515	50	2081.70	2115.50	2140.30	3.41820
α = -0.322	20	1748.80	1750.90	1767.20	0.98490
α = -0.322	30	3400.60	3458.50	3626.20	1.57800
α = -0.322	40	5602.10	5672.20	5794.90	2.38380
α = -0.322	50	4454.00	4525.60	4675.80	3.41620
α = -0.152	20	3287.10	3291.10	3319.20	0.98477
α = -0.152	30	6823.00	6827.30	6862.80	1.56870
α = -0.152	40	13726	13786	13890	2.35200
α = -0.152	50	13966	14000	14145	3.40210

5 Conclusions

In this paper, we consider The single machine scheduling problem, in which, each job has an individual due-date, earliness and tardiness penalties is considered in presence of a learning effect. Because of the learning effect, the processing time of a job depends on its position in the sequence. The objective is to find the sequence of jobs that minimizes the weighted sum of earliness and tardiness costs, without considering machine idle times with a learning effect. Since, this is a difficult combinatorial optimization problem, we have used genetic algorithm approach to obtain the best/optimal sequence. We shown via numerical examples that the genetic algorithm approach takes only small computation time to solve fairly large problems (50 jobs).

References

1. Biskup, D.: Single-machine scheduling with learning considerations. *European Journal of Operational Research* **115** (1999) 173-178
2. Mosheiov, G.: Scheduling problems with a learning effect. *European Journal of Operational Research* **132** (2001) 687-693
3. Lee, W.C., Wu, C.C.: Minimizing total completion time in a two-machine flowshop with a learning effect. *International Journal of Production Economics* **88** (2004) 85-93
4. Mosheiov, G.: Parallel machine scheduling with a learning effect. *Journal of the Operational Research Society* **52** (2001) 1165-1169
5. Wang, J.B., Xia, Z.Q.: Flow-shop scheduling with a learning effect. *Journal of the Operational Research Society* **56** (2005) 1325-1330
6. Mosheiov, G., Sidney J.B.: Note on scheduling with general learning curves to minimize the number of tardy jobs. *Journal of the Operational Research Society* **56** (2005) 110-112

7. Biskup, D., Simons, D.: Common due date scheduling with autonomous and induced learning. *European Journal of Operational Research* **159** (2004) 606-616
8. Bachman, A., Janiak, A.: Scheduling jobs with position-dependent processing times. *Journal of the Operational Research Society* **55** (2004) 257-264
9. Kuo, W.H., Yang, D.L.: Minimizing the makespan in a single machine scheduling problem with a time-based learning effect. *Information Processing Letters* **97** (2006) 64-67
10. Du, J., Leung, J.Y.T.: Minimizing total tardiness on one machine is NP-hard. *Mathematics of Operations Research* **15** (1990) 483-495
11. Cheng, T.C.E., Ding, Q., Kovalyov, M.Y., Bachman, A., Janiak. A.: Scheduling jobs with linearly decreasing processing times. *Naval Research Logistics* **50** (2003) 531-555
12. Cheng, T.C.E., Ding, Q., Lin, B.M.T.: A concise survey of scheduling with time-dependent processing times. *European Journal of Operational Research* **152** (2004) 1-13
13. Holland, H.J.: *Adaption in natural and artificial systems.* University of Michigan Press, Ann Arbor, USA (1975)
14. Goldberg, D.E.: *Genetic algorithms in search, optimization and machine learning.* Addison-Wesley, New York, USA (1989)
15. David, L.: *Handbook of genetic algorithms.* Van Nostrand Reingold, New York, USA (1991)
16. Michalewicz, Z.: *Genetic Algorithms + Data Structures = Evolution Programs.* AI Series Springer-Verlag, New York, USA (1994)
17. Chaudhry, S.S., Luo, W.: Application of genetic algorithms in production and operations management: a review. *International Journal of Production Research* **19** (2005) 4083-4101
18. Ow, P.S., Morton, E.T.: The single machine early/tardy problem. *Management Science* **35** (1989) 171-191
19. Azizoglu, M., Kondakci, S., Krica, O.: Bicriterion scheduling problems involving total tardiness and total earliness penalties. *International Journal of Production Economics* **23** (1991) 17-24
20. Su, L.H., Chang, P.C.: A heuristic to minimize a quadratic function of job lateness on a single machine. *International Journal of Production Economics* **55** (1998) 169-175
21. Su, L.H., Chang, P.C.: Scheduling n jobs on one machine to minimize the maximum lateness with a minimum number of tardy jobs. *Computers and Industrial Engineering* **40** (2001) 349-360
22. Nadler, G., Smith, W.D.: Manufacturing progress functions for types of processes. *International Journal of Production Research* **2** (1963) 115-135
23. Sourd, F.: http://www-poleia.lip6.fr/~sourd/
24. Chang, P.C.,: http://ppc.iem.yzu.edu.tw/publication/sourceCode/ SingleMachineLearningEffect/