Proceedings of the World Congress on Engineering 2011 Vol II
WCE 2011, July 6 - 8, 2011, London, U.K.

An Application with Non-identical Parallel Machines using Genetic Algorithm with the Help of Fuzzy Logic

Pelin Alcan, Hüseyin Başlıgil

***Abstract*—After industrial revolution in Europe, manufacturing systems have become very important research areas. In shifting and improving world, manufacturing is a positive function of consumption of humankind. In this regard, there has been too many algorithms and methodologies about scheduling to find a solution to combinatorial complex problems. In the literature, most of the articles about production scheduling are related with identical parallel machines. Even though it is a common problem in the industry, only a small number of studies deal with non-identical parallel machines. This paper is about fuzzy approach to scheduling problem of non-identical parallel machines using genetic algorithm. The genetic algorithm is proposed here fits the non-identical parallel machine scheduling problem of minimizing the maximum completion time.**

Index Terms—: ***Genetic algorithm, Scheduling, Non-identical parallel machine, Fuzzy approach***.

I. Introduction

Scheduling is an essential function in production management. Scheduling determines what is going to be made, when, where and with what resources [1]. In the literature, several heuristics and dispatching rules are proposed to solve such hard combinatorial optimization problems and Genetic Algorithm (GA) ranks among the most preferred ones in view of its characteristics such as high adaptability, near optimization and easy realization. But, even though it is a common problem in the industry, only a small number of studies deal with non-identical parallel machines [2]. Most of the scheduling problems are NP-hard. In this work, the scheduling problem is also NP-hard. The most important problem studied in this paper regards the scheduling of independent jobs on non-identical parallel machines using Genetic Algorithm in order to minimize "makespan" with the help of fuzzy numbers. The application has been done with the program of JAVA. The organization of the paper is as follows: in section 2, production scheduling and non-identical parallel machines are presented. Section 3 discusses the fuzzy sets and fuzzy numbers. Section 4 presents genetic algorithm. Our implementation of GA to the problem is the subject of Section 5. Section 6 concludes the paper.

Pelin Alcan is with the Yildiz Technical University, Mechanical Faculty, The Department of Industrial Engineering, Yıldız-İSTANBUL(corresponding author to provide phone: +902123832922; e-mail: palcan@yildiz.edu.tr).
Hüseyin Başlıgil is with the Yildiz Technical University, Mechanical Faculty, The Department of Industrial Engineering, Yıldız-İSTANBUL (e-mail: basligil@yildiz.edu.tr).

II. Production Scheduling and Non-identical Parallel Machines

Production scheduling is an essential activity in manufacturing. The production scheduling is an important decision making in operational level and it is a difficult problem depending on the number of calculations required to obtain a scheduling that optimizes the chosen criterion. In modern manufacturing environment, many scheduling problems arise. In the factory, depending on machine layout and job flow, several kinds of shop exist [3]. Parallel machine scheduling (PMS) is to schedule jobs processed on a series of same function machines with the optimized objective. Suppose that m machines $Mi(i = 1, \dots, m)$ process n jobs $Jj\ (j = 1, \dots, n)$. Almost all researchers studied PMS problems under the hypothesis that each job can be processed on at most one machine at a time although preemption is allowed [4]. In the parallel machine scheduling problems, when p_{ij} = p_j for all i and j, the machines have the same speed, thus the processing times of a job are identical on different machines and the machines are called identical. When p_{ijs} = $p_j\ /s_i$ where p_j is the processing time of job j and s_i is the speed of machine i, then the machines are called uniform. If the $p_{ij's}$ are arbitrary then the machines are called unrelated. And both of the uniform and unrelated cases belong to non-identical parallel machine schedules [5].

III. Fuzzy Sets and Fuzzy Numbers

To deal with vagueness of human thought, Zadeh (1965) [6] first introduced the fuzzy set theory, which was oriented to the rationality of uncertainty due to imprecision or vagueness. A fuzzy set is a class of object s with a continuum of grades of membership. Such a set is characterized by a membership (characteristic) function, which assigns to each object a grade of member ship ranging between zero and one [7]. A triangular fuzzy number (TFN, $l \leq m \leq u$, $\tilde{M} = (l, m, u)$), has linear representations on its left and right side such that its membership function can be defined as;

$$\mu_M = \begin{cases} 0, & x < l \\ \frac{x-l}{m-l}, & l \leq x \leq m \\ \frac{u-x}{u-m}, & m \leq x \leq u \\ 0, & x > u \end{cases} \qquad (1)$$

Most approaches are biased on the possibility concept and/or the probability measure of fuzzy events concept. For the problem, we present a method of Lee and Li (1988) [8] to bear with a mean and a standard deviation of the fuzzy events to order fuzzy processing times. According to this, mean and standard deviation of M triangular fuzzy number is equal to;

$$\overline{x(M)} = \frac{1}{3}(a + b + c) \qquad (2)$$

$$\sigma_{(M)} = \frac{1}{18}(a^2 + b^2 + c^2 - ab - ac - bc) \qquad (3)$$

IV. Genetic Algorithm

Various heuristics have been developed in order to find near-optimal solutions in a relatively short time. Most often applied heuristics in practice are dispatching rules that have low computational complexity and are easy to implement. In contrast to other local search methods, such as Simulated Annealing or Tabu Search, which are based on handling one feasible solution, GA utilizes a population of solutions in its search, giving it more resistance to premature convergence on local minima [9]. Many heuristics have been proposed such as Simulated Annealing (SA), Branch And Bound (B&B), Neural Network (NN), and Genetic Algorithm (GA),… Among these various approaches to different scheduling problems, there has been an increasing interest in applying GA in view of its characteristic such as high adaptability, near optimization and easy realization. The primary difference between GA and other heuristics is that GA maintains a set of solutions (population) rather than a unique solution [2].

A. Encoding

Genetic Algorithm has its own language. In the literature, it is called as “coding”. Encoding very depends on the problem. To perform genetic algorithm, initial solutions have to be done encoded in conformity with fitness function.

B. Generation of initial population

Members of the initial population (chromosomes) are the parents of the next generations and the efficiency of the algorithm is highly dependent on their “quality”.

C. Calculation of fitness function

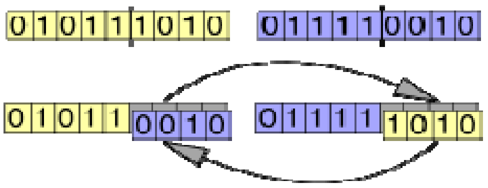
Fitness is the performance evaluation of chromosomes. After the generation of a new population, fitness value of each chromosome is calculated (Fk) [10].

D. Reproduction

Reproduction is the process in which parents copy themselves according to the probabilities that are proportional to their fitness values. As a result, parents with higher fitness values will have higher probabilities of producing their offspring in the next generation [11].

E. Crossover

Crossover is a genetic operator that combines (mates) two chromosomes (parents) to produce a new chromosome (offspring).



Parent 1: 1011000\10100001001
Parent 2: 0101011\11000110101
Child 1: 1011000\11000110101
Child 2: 0101011\10100001001

Fig. 1. Crossover operator

F. Mutation

Mutation is a genetic operator that alters one ore more gene values in a chromosome from its initial state. This can result in entirely new gene values being added to the gene pool.

Child 1: 101100011000110101
101101011000110101

G. Optimality criterion

After mutation a new generation is obtained, an optimality test is to be performed as the last step of the algorithm.

V. Application

The application is a real case study in this article. In this section, the scheduling problem of n jobs on m non-identical machines is solved using GA described above. The processing times on m machines have been converted to fuzzy numbers. The algorithm is coded in JAVA, and run on an Intel T2300 1,66 GHz PC. The objective is to minimize makespan$(Cmax)$. Velocity of machines are given as;

$$(V1, V2, V3, V4,) = (1, 2, 3, 4).$$

To measure the performance of this objective healthy, we will use an objective function which has the smallest $Cmax$ and the smallest maximum delay time value at the same time in the job-machine scheduling. Mathematical equations of multipurpose scheduling problem with fuzzy due dates and fuzzy processing times are given;

$$\min \tilde{f}(x) = (\tilde{C}_{\max}(x), \tilde{T}_{\max}(x)) \qquad (4)$$

$$\tilde{C}_{\max}(x) = \tilde{C}_{n,m}(x) \quad (5)$$

$$\tilde{T}_{\max}(x) = \max\left\{\tilde{T}_1, \tilde{T}_2, \ldots, \tilde{T}_n\right\} \quad (6)$$

Here, with the (4)th. equation, it is required to compose a $f(x)$ function with the minimum of together completion time and the maximum lateness of the last job. The makespan is the completion time of the last job to leave the system.

Constraint set (5) ensures that the completion time of the last job is equal to the completion time of job n on machine m.

Constraint set (6) ensures that the maximum lateness is equal to the biggest value of all job's lateness. The tardiness of job j is defined as;

$$T_i = \left\{C_{i,m} - d_i, 0\right\} \quad (7)$$

In the mathematical notation, the x value denotes that "job scheduling". The solution represents effective points in the problem. If there is not another x vector which provides eq.(8), x^* is characterized as the effective solution.

$$\begin{aligned} &\tilde{f}_i(x) \le \tilde{f}_i(x^*) && \forall i \quad i = 1,2,3 \\ &\tilde{f}_i(x) < \tilde{f}_i(x^*) && \textit{at least for one } i \end{aligned} \quad (8)$$

First of all, total job numbers (n), total machine numbers (m), total machine groups (g), fuzzy processing times of the jobs and due dates are entried to the program of JAVA. Later the steps of GA with the program are given;

Step1: N alternative solutions are produced randomly for $t = 0$,

Step2: rj positive random numbers are produced for the chromosomes (job orders) into the GEN(t) batch. After, the selection probability of the chromosome is attained for the momentary iteration. The program is run to access the smallest $f(t)$ value according to the rule of effective solution.

$$w_k = \frac{r_k}{\sum_{j=1}^{3} r_j} \quad (9)$$

$$\tilde{f}_{[i]} = w_1.\tilde{C}_{\max_i} + w_2.\tilde{T}_{\max_i} \quad (10)$$

Step3: Effective solutions are modified in the initial batch and then added to the solution group.

Step4: From the batch, the individuals are selected as binary codes according to the selection probability. Following this process, crossover and mutation are applied to the individuals.

Step5: In step 5, fitness function values of the recent batch are computed. The chromosomes which optimized each of the weighted object from GEN (t) are added to the GEN (t+1).

Step6: The effective solution set is updated.

Step7: Finally, 4th, 5th and 6th steps are applied to yield the condition of the end for the algorithm. t = t +1 is ensured. After that, it is gone to the next iteration. Problem sizes with 5 jobs-4 machines, 5 jobs-5 machines, 5 jobs-6 machines, 10 jobs- 4 machines, 10 jobs-5 machines, 10 jobs-6 machines have been considered in our numerical experiments.

But in this paper we will give the data tables of only one instance (10 jobs-6 machines) from the numerical experiments. For each problem size, problem instances have been randomly generated.

Table 1 shows the structure of the mean processing times of the jobs.

In this application, on a manufacture line with 8 machines, 5 non identical parallel turn machine and 3 non identical parallel drill machines are used. Firstly the products are got through from the turn machine and then move to the drill machine.

TABLE 1.
MEAN PROCESSING TIMES OF THE JOBS

	Jobs										
Machines		1	2	3	4	5	6	7	8	9	10
	1	2,3	3,5	6,2	5,4	12,1	11,1	13	5,5	3,2	1,1
	2	2,6	3,4	5,5	3	13,5	10,9	12,5	10	5	3,7
	3	3,5	3,3	3,4	4,5	11	8,8	10,7	11,2	4,4	2,2
	4	2,8	3,6	4,8	4,4	9,8	9,3	9,9	3,7	5,7	3,1
	5	5,3	2,4	6,4	6,7	11,3	8,4	5,8	6,8	6	6,2
	6	4	3,7	3,6	3,4	9,6	10,7	8,4	8,3	3,8	5,1

The best and the worst processing times are given in Table 2.

TABLE 2.
THE BEST AND THE WORST PROCESSING TIMES OF THE JOBS

The Best	Jobs										
Machines		1	2	3	4	5	6	7	8	9	10
	1	1	1	3	2	8	7	8	2	2	1
	2	2	1	1	2	4	8	10	7	4	2
	3	1	2	3	4	8	6	8	10	3	2
	4	3	2	2	3	8	6	8	2	5	2
	5	3	1	4	6	10	5	3	5	3	4
	6	2	2	2	2	8	8	5	4	2	4

The Worst	Jobs										
Machines		1	2	3	4	5	6	7	8	9	10
	1	8	6	10	8	15	13	17	9	5	3
	2	5	5	6	4	15	13	15	12	6	7
	3	5	5	7	6	14	10	12	13	6	3
	4	5	6	7	6	11	12	11	4	7	5
	5	7	4	8	9	14	10	7	8	11	9
	6	4	5	7	5	11	14	10	11	6	9

When the program is started, it is asked the numbers of the jobs and the machines from the user. With the codes of the data entry, problem solutions can be found until 1000 jobs-1000 machines. Then, fuzzy processing times of the jobs are recorded as mean, the best and the worst values.

The groups of the machines are defined respectively. After that, the rates of the elitism and mutation, the iteration number and the due date coefficient of the jobs are enlisted with the help of scrolls.

In our application, the finalizing step condition N is acceded as the value of 100.

Table 3 gives the final C_{max} values of the jobs.

TABLE 3.
FINAL C_{MAX} VALUES OF THE JOBS

Test Problems	C_{max}	Order of the machines
5 jobs - 4 machines	12,34	3,2,4,2,4
10 jobs - 4 machines	21,22	3,2,4,2,4,3,4,1,1,1
5 jobs - 5 machines	8,96	3,1,2,2,4
10 jobs - 5 machines	18,55	3,1,2,2,4,4,5,4,1,1
5 jobs - 6 machines	8,4	3,5,2,6,1
10 jobs - 6 machines	13,56	3,5,2,6,1,4,5,4,1,1

Output data of the 10 jobs and 6 machines are given;

```
Total Time:13.564512999824501
Fitness value:19.480390821317044
------------------------
------------------------
------------------------
Total idle time for the 1st machine:
0.0
------------------------
Total idle time for the 2nd machine:
 9.912465093664995
------------------------
Total idle time for the 3rd machine:
11.83761913357052
------------------------
Total idle time for the 4th machine:
2.31941452864643
------------------------
Total idle time for the 5th machine:
8.359425186428298
------------------------
Total idle time for the 6th machine:
10.714978164976975
------------------------
Cmax of the 5th job:13.564512999824501
------------------------
Cmax of the 6th  job:7.7705677889456135
------------------------
Cmax of the 7th job:5.205087813396203
------------------------
Cmax of the 8th  job:11.245098471178071
------------------------
Cmax of the 9th job:5.097983127294722
------------------------
Cmax of the 10th job:1.7131881545460965
```

VI. CONCLUSION

In this paper, we have investigated GA approach for minimizing makespan for the non-identical parallel machines with fuzzy triangular numbers. From our computational experiences, we can give the following conclusions and recommendations:

- Dispatching rules are suitable only for small scale problems and no single dispatching rule guarantees good result in various problems.
- Research efforts have therefore concentrated on heuristic approaches. Among these approaches, GA outperforms others in view of its characteristic such as high adaptability, near optimization and easy realization. There are so many applications of GA to solve parallel machine scheduling problem; but, even though it is a common problem in the industry, only a small number of them deal with non-identical parallel machines.

Further research can be done to use other iterative algorithms such as ant colony algorithms. The choice of good parameters for them should be tested. In addition to the uncertainty of processing times, it is possible to extend the algorithm by adding cost information and important levels of objective function which they are indefinite.

REFERENCES

[1] A. Cardon, T. Galinho, Vacher J-P., , "Genetic algorithms using multi-objectives in a multi-agent system", Robotics and Autonomous Systems, 2000, 33, pp. 179–190.
[2] S. Balin, "Non-identical parallel machine scheduling using genetic algorithm", Expert Systems with Applications, 2011, 38, 6, pp. 6814-6821.
[3] C. Moon, M. Lee, Y. Seo, Y.H. Lee, "Integrated machine tool selection and operation sequencing with capacity and precedence constraints using genetic algorithm", Computers & Industrial Engineering, 2002, 43, 3, pp. 605-621.
[4] W. Xing, J. Zhang, "Parallel machine scheduling with splitting jobs", Discrete Applied Mathematics, 2000, 103, pp. 259-269.
[5] K. Li, S. Yang, "Non-identical parallel-machine scheduling research with minimizing total weighted completion times: Models, relaxations and algorithms". Applied Mathematical Modelling, 2009, 33, pp. 2145–2158.
[6] L. Zadeh, "Fuzzy sets, Information control". 1965, 8, pp. 338-353.
[7] C. Kahraman, U. Cebeci, D. Ruan, "Multi-attribute comparison of catering service companies using fuzzy AHP: The case of Turkey", Int. J. Production Economics, 2004, 87, pp. 171–184.
[8] E.S. Lee, R.J. Li, "Comparison of fuzzy numbers based on the probability measure of fuzzy events". Computers & Mathematics with Applications, 1988, 15, 10, pp. 887–896.
[9] B.J. Park, H.R. Choi, Kim H.S., "A hybrid genetic algorithm for the job shop scheduling problems", Computers & Industrial Engineering, 2003, 45, pp. 597–613.
[10] D.E. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning", Addison-Wesley, 1989, MA.
[11] H. Zhou, Y. Feng, L. Han, , "The hybrid heuristic genetic algorithm for job shop scheduling", Computers & Industrial Eng, 2001, 40, 3, 191-200.