Gazi University Journal of Science
GU J Sci
24(1):113-123 (2011)

Scheduling Two Parallel Machines with Sequence-dependent Setups and A Single Server

A. Kursad TURKER[1♠], Cagri SEL[1]

[1] *Kırıkkale University, Faculty of Engineering, Department of Industrial Engineering, Kırıkkale, TURKEY*

Received: 16.07.2010 Accepted: 19.08.2010

ABSTRACT

This paper presents a scheduling problem on parallel machines with sequence-dependent setup times and setup operations that performed by a single server. The main purpose is to get minimum makespan of the schedule. The system is formulated as genetic algorithm with problem sizes consisting of two machines and 10, 20 and 30 jobs. A genetic algorithm is developed using random data sets. The optimum results are obtained using a string based permutation algorithm which scans all alternatives. As a result, proposed algorithm is effective to solve P2,S|STsd|Cmax scheduling problem on reasonable runtime and the results of the algorithm which are close to optimum solution values. Effectiveness of the solution is presented considering approximation rates of the genetic algorithm solutions to the optimum results obtained for P2,S|STsd|Cmax problem.

Key Words: *Parallel Machine Scheduling, Genetic Algorithm, Permutation Algorithm.*

1. INTRODUCTION

Many manufacturing systems have parallel machine architectures. A proper allocation of resources is required for this type of manufacturing environment to optimize objectives and achieve goals. There are many approaches to achieve fundamental scheduling solutions of the basic parallel machine problems without setup times. Also, parallel machine scheduling problems with setup times are relatively take place in the literature. In spite of being widespread, setup times can be operated by inserting into processing times without any server constraint in scheduling problems which is presented by researches. Limited server problems require a certain schedule to allocate the servers. Moreover, setup times which are performed by limited server can be sequence dependent. When there is just a single server, only one setup operation can be allocated to the server and it causes to block times. These conditions increase the complexity of the problem and affect the performance of the solution. Abdekhodaee et al. (2002) presents a parallel machine scheduling problem with a single server and sequence independent setup times. It is emphasized that the problem is strongly NP-Hard. In this article, more complex problem known as NP-Hard with sequence dependent setup times are considered [2].

♠Corresponding author, e-mail: kturker@kku.edu.tr

2. LITERATURE REVIEW

First comprehensive literature review is provided on scheduling problems involving setup times (costs) by A. Allahverdi et al. (1999). In the study which includes researches from middle of the 1960s to 1999, it is clarified that the majority of scheduling research assumes setup as negligible or part of the processing time and while this assumption simplifies the analysis, it adversely affects the solution quality for many applications. Consequently, it is emphasized that most of the parallel machine scheduling problems are studied with sequence independent setup times and there is need for research on the parallel-machine including sequence dependent setup operations [4].

The objective of the paper presented by Allahverdi et al. (2006) as a second literature review is to provide an extensive review of the scheduling literature on models with setup times (costs) from 1999 to 2006. In the result, single server parallel machine scheduling problems with sequence independent and unit or equal setup times are evaluated as a problem which has computational complexity. When it comes to a sequence dependent setup time, it is clearly harder to solve this problem than sequence independent problems [3].

One of the researches about dedicated parallel machines which have sequence dependent setup times is presented by Guinet (1993). The debated problem is to get the convenience schedule providing non-stop process on the machines. It is aimed to decrease maximum or average completion times. For this purpose, a new allocation algorithm is introduced by using Hungarian method [7].

Another parallel machine study is performed by Sivrikaya-Serifoglu and Ulusoy (1999). A parallel machine problem which consists of earliness and tardiness penalties is examined in details. Main purpose of the paper is to minimize the earliness and tardiness penalties for parallel machines including sequence dependent setup times. Due dates are different from each other. Moreover, each job has different ready times. Hence, these states complicate the problem. Two distinct genetic algorithm methods are used to solve the problem. First of these methods is a genetic algorithm which consist of a multi-component chromosome representation and appropriate crossover approach for the structure. Second method includes a genetic algorithm without a crossover operator. As a result of 960 random problems, the genetic algorithms are determined as an efficient method to solve parallel problems. Finally, experiments are clarified that first genetic algorithm structure is better to get solutions than the second one for vast majority of complex and big sized problems [13].

Kurz and Askin (2001) emphasized that parallel usage of multi-process stations is common strategy to obtain adequate capacity. Also it is speculated that setup operations are important for changeover of products. Main goal of the study is to get minimum makespan when there is sequence dependent setup times and non-zero ready times. An integer programming model is introduced and the model is compared to heuristics, genetic algorithms, and traveling salesman problem. As a result, a heuristic is produced to get the appropriate solution of the problem [11].

Wilson et al. (2004) studied on a scheduling problem of furniture production cut and sew process. Makespan minimization through efficient schedule is the fundamental goal of the study. Production process includes two stages, and each of these stages has identical parallel machines operating in a flow line. A heuristic is introduced for second stage including multi-setup operations. The heuristic is compared to a single setup per group at each stage and integrated to a genetic algorithm. Results of the study show that the heuristic effectively adds minimal setups to a single setup schedule while improving makespan of the schedule significantly [14].

Abdekhodaee et. al. (2006) considered a solution on two identical semi-automatic machines for problem of scheduling two operations non-preemptive jobs. There is just a single server to perform the setup operations or first operation. The second operation is executed automatically, without the server. Main purpose of the study is the makespan minimization. Firstly, effective and efficient solution strategies are presented for special cases as equal process and setup times. These special cases are considered to deal with the real problem. Heuristic methods can be used to solve the problem by reducing the problem complexity to regular level. Second alternative approach which is proposed in the article is genetic algorithm to deal with this kind of NP-Hard scheduling problems. Performances of these algorithms are reported in the paper [1].

Huang et al. (2009) presented a genetic algorithm which gives solutions on parallel dedicated machine scheduling problem with single server and sequence dependent setup times. The goal is to find the minimum makespan of system. Problem is formulated as an integer programming model. A special case of the problem is presented in the article. The special case reduces complexity of the problem by grouping the jobs and allocating the dedicated machines to jobs. A hybrid genetic algorithm is introduced to solve general cases using greedy heuristic. The algorithm is examined by random data sets and real-world data sets from the printing industry .The results of the experiments speculated that the algorithm is efficient and effective for both types of data sets [8].

Recently, there has been an increasing interest in scheduling problems with sequence dependent setup times. However, there is no research on the scheduling problem on identical parallel machines with single server and sequence dependent setups in the form which is presented in this paper. Researches which are reviewed from the scheduling literature are summarized and presented on Table 1.

3. PROBLEM DESCRIPTION

Scheduling problem which is considered in this article is minimization of makespan on two identical parallel machines with sequence dependent setup times and a single server. There exist two distinct constraints for the problem. One of the important constraints is that a job can be operated just on a single machine at the same time.

Also, each job can be performed on any of the machines. Another constraint is that only a single setup operation can be performed on the server at the same time.

Parallel machine problems consist of allocation and permutation problems. Optimal solution alternatives are determined by solving these two problems. The problem which is considered in this paper includes sequence dependent setup times. This situation increases the complexity of the problem for completion time. Hence, it is considered that minimum makespan is main goal of the problem. Schematic presentation of the problem is shown below on Figure 1.

Table 1. Literature Review.

Date	Title	Author	Problem
1993	Scheduling sequence dependent jobs on identical parallel machines	A. Guinet	PD|STsd|Cmax
1999	Parallel machine scheduling with earliness and tardiness penalties	F. Sivrikaya G. Ulusoy	$WE\sum Ej$ + $WT\sum Tj$
1999	A review of scheduling research involving setup considerations	A. Allahverdi J.N.D. Gupta T. Aldowaisan	Literature Review
2001	Heuristic scheduling of parallel machines with sequence dependent setup times	M. E. Kurz R. G. Askin	P|rj,sij|Cmax
2004	Scheduling non-similar groups on a flow line: multiple group setups	A. D. Wilson R. E. King T. J. Hodgson	P|STsi,rj|Cmax
2006	A survey of scheduling problems with setup times or costs	A. Allahverdi C. T. Ng T. C. E. Cheng M. Y. Kovalyov	Literature Review
2006	Scheduling two parallel machines with a single server: the general case	A. H. Abdekhodaee A.Wirth H. Gan	P2,S|pi,si|Cmax
2009	Parallel dedicated machine scheduling problem with sequence dependent setups and a single server	S. Huang L. Chai X. Zhang	PD,S|STsd |Cmax

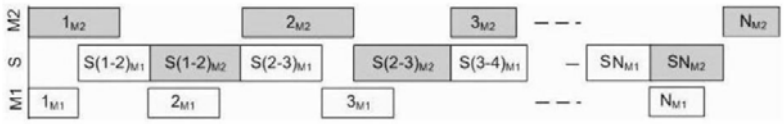

Figure 1. Schematic Presentation of Problem.

For convenience and readability, the parameters and notations of scheduling problems are summarized for problem P2,S|STsd|Cmax on Table 2 below:

Table 2. Parameters and Notifications.

Symbol		Explanation	
M	:	set of machines,	$M = \{1, 2\}$;
J	:	set of jobs,	$J = \{1, 2, ..., n\}$;
$m(j)$	:	machine to process job j;	
p_j	:	processing time of job j;	
s_{ij}	:	setup time for job j if its preceding job on the machine is job i;	
P, S	:	parallel machine with single server;	
ST_{sd}	:	sequence-dependent setup time;	
C_{max}	:	objective is to minimize makespan;	

4. GENETIC ALGORITHM SOLUTION

Genetic algorithms (GA) are algorithms based on natural selection, natural genetics and common methods to solve the NP-Hard scheduling problems. The genetic algorithms (GA) have the great advantage and success in the solution of NP problems. There are various important applications on this way [5]. Fundamental genetic algorithms consist of three important components. The first important component is chromosomes which mean the solution alternatives. Secondly, each of these chromosomes includes genes and these genes are minor parts of the solution. For parallel machine scheduling problems, the genes generally presents either allocation or permutation. All of these compositions form the population as sample space of model. Summarily, combination of the genes composes chromosomes and combination of these chromosomes composes population. Fundamental structure of GA is constituted over these notions.

The general structure for the genetic algorithm implemented here follows [9, 11];

1. Initialize a population of chromosomes.
2. Evaluate the chromosomes in the population to get fitness of each chromosome.
3. Find the new incumbent chromosome. If the stopping criteria met, go to Step 4.
4. Reproduce to create the next population, allowing chromosomes with higher fitness values to have higher chances of reproduction.
5. Apply the crossover operations to the current population
6. Apply the mutation operations to the current population
7. Go to step 2.

4.1. Chromosome Representation

For complex scheduling problems, genes present both allocation and permutation. Hence, chromosome structure is designed as multi-component and the representation incorporates both the job sequencing and the machine selection. There are n genes each of which corresponds to one of the jobs. Each gene contains two data: one of them specifies the job and another specifies the machine to which the job is assigned. For example, when there is a problem of n=10 jobs and m=2 machine, one of chromosomes is [7-1, 6-2, 1-1, 5-2, 3-1, 8-2, 9-1, 10-2, 4-1, 2-2]. This presentation indicates that 7, 1, 3, 9, 4 jobs are operated on machine 1 and 6, 5, 8, 10, 2 jobs are operated on machine 2 respectively.

4.2. Population Initialization

The performance of GA depends on chromosome diversity. Hence, population initialization strategy is an important process. Generally, it is a reasonable approach to determine the initial individuals using a heuristic, rather than randomly generating initial population. In our approach, the initial population is generated by using three distinct methods. First method is a mechanism which chooses the best 100 individuals among the population which is 10 times greater than the determined population size. The second one is a mechanism which chooses 100 individuals randomly among the population which is 10 times greater than the determined population size. The last method is a mechanism which chooses 100 individuals using pareto's rule of 60-30-10 percent, which means to choose 60 individuals from 600, 30 individuals from remaining 300 and 10 individuals from last 100, among the 10 times larger population. The Pareto method is considered as a mechanism to support chromosome diversity. The effect of these methods is compared in detail on Section 6.

4.3. Chromosome Evaluation

GA is a maximization algorithm. However, objective function of the studied problem is minimization of Cmax which is completion time. Fitness function is evaluated with a transfer functions using Cmax value to use GA as a minimization structure. Another notation used for transformation is IdealCmax. IdealCmax is the best solution alternative neglecting idle times for machines because of single server constraint. IdealCmax is calculated with the sum of total minimum process times and total minimum sequence dependent setup times.

The fitness function (1) is obtained to minimize the makespan of the system. The function defines the minimum completion time of all the machines. Transfer function is shown below:

$$\text{Fitness(i)} = \text{idealCmax} - \text{Cmax(i)} \quad (1)$$

Table 3. Notations of Fitness Function.

Symbol		Explanation	
$Fitness(i)$	:	Fitness value for i. chromosome	$i = \{1, 2, ...popsize\}$;
$Cmax(i)$	:	Completion time	$i = \{1, 2, ...popsize\}$;
$IdealCmax$	:	The best solution alternative	

4.4. Stopping Criteria

Stopping criteria of the algorithm is the iteration limit for the algorithm. Maximum generation number is determined as stopping criteria. When maximum generation number is provided by the counter used by the algorithm to count iterations, the algorithm stops and presents the best solution. Maximum generation number is determined as large as possible on the Table 4 to get most appropriate results.

Table 4. Maximum Generation Numbers.

Problem Size n jobs m machines	Maximum Generation Number
n = 10 m = 2	5000
n = 20 m = 2	3000
n = 30 m = 2	1000

4.5. Reproduction Operator

Reproduction operator is a mechanism that determines the individuals selected for new generation. Roulette wheel mechanism is applied in our approach. In the mechanism, individuals have opportunities depending on their fitness value. Selection mechanism is, therefore, not applied for the 10% percent of the individuals that have the best fitness value existing in the population. These individuals are autonomously transferred to new generation. In other words, selecting mechanism is actuated for just 90% percent for the remaining individuals of the populations.

The selection mechanism (2) is obtained to determine the individuals in accordance with the fitness value. Roulette Wheel formula applied for selecting mechanism and notations, are presented below;

$$Roulette(i) = \frac{Fitness(i)}{\sum_{j=popsize*\%10}^{popsize} Fitness(j)} \quad (2)$$

Table 5. Notations of Roulette Wheel.

Symbol		Explanation	
$Fitness(i)$	:	Fitness value for i. chromosome	$i = \{popsize * \%10 + 1, popsize * \%10 + 2, ...popsize\}$;
$Roulette(i)$	:	Completion time	$i = \{1, 2, ...popsize\}$;

4.6. Crossover Operator

Kellegoz et al. (2008) considered one machine problem with the performance criterion of minimizing total weighted tardiness. The problem consisting one machine and n independent jobs is known as NP-Hard. Eleven genetic crossover operators which have been widely used to solve other types of hard scheduling problems are compared to test the performances. As a result of experiments, order based crossover (OBX) and position based crossover (PBX) crossover operators are found to be effective for this kind of scheduling problems [15]. In accordance with these results, OBX algorithm is applied

to our genetic algorithm on all components indicating jobs.

In OBX method, the order of jobs selected with 0.5 probability in first parent is transferred to the corresponding position on the first offspring. Then, other jobs not selected for the first offspring ever are transferred from second parent by conserving their absolute order. After changing the roles of parents, the same procedure is applied to the production of the second offspring in the other parent. Schematic presentation of OBX method given by Kellegoz et al. (2008) is shown on Figure 2 [10]. However, crossover mechanism of the components indicating machines is operated using a different approach. In this approach, just a single point of machine array is chosen and swapped randomly from chromosomes. Crossover probability is operative as 0.5 for the crossover mechanism of the components indicating machines.

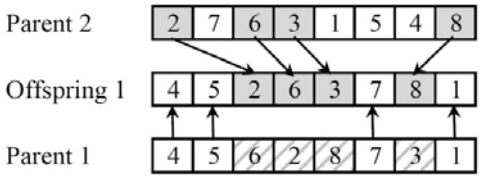

Figure 2. Order based crossover operator (OBX)[9].

4.7. Mutation Operator

The mutation operator is designed to bring more diversity into the search procedure. The chromosomes which are located in new generation are mutated with a low mutation probability using bit mutation. Chromosomes are selected with 0.25 probabilities for mutation process. The same probability is applied for the mutation of the genes which composes the chromosome. One of the machines is chosen randomly and a new machine is arbitrarily reassigned to the job at the position.

4.8. Parameter Tuning

There are three factors which affect the performance of genetic algorithm. These are population size, crossover and mutation probability. The population size was arranged from a uniform distribution over the values 10 and 100 step 10, crossover and mutation probabilities are examined with 0.25, 0.5, and 0.75. For each instance and each level, five independent runs were executed, and the relative Cmax value was computed. After initial experimentation, the parameters were found as 100, 0.5, and 0.25 respectively.

5. PERMUTATION ALGORITHM

The permutation algorithm is designed to obtain the optimum results of the problem sizes consisting of two machines and 10, 20 and 30 jobs. This algorithm is a permutation algorithm based on a string permutation algorithm calculated via recursion method. This permutation algorithm is used to evaluate all the alternatives of problem sets. The number of alternative solutions increases exponentially. Since all alternatives are scanned one by one, the permutation algorithm takes long time. The number of alternative solution for all of the problem sizes consisting of two machines and 10, 20 and 30 jobs are shown on Table 6.

Table 6. Number of Alternative Solutions.

Problem Size n jobs m machines	Number of Alternative Permutations
n = 10 m = 2	3715891200
n = 20 m = 2	2551082656125828464640000
n = 30 m = 2	2,848130895159583247366408199418 7e+41

6. NUMERICAL RESULTS AND DISCUSSION

The genetic algorithm is experimented with randomly generated problems. The alternative problems generated for general situations clarified that the algorithm is efficient to find reasonable results. The genetic algorithm finds significantly close Cmax values and schedules to optimal results. Statistical information about solution is presented on Table 7 the parameters of randomly generated problems are processing times, sequence dependent setup times, setup time structures, number of machines. Low and High limits of these parameters are shown on Table 8 below:

Table 7. Statistical Information.

Problem Sizes	Init. Approach	Mean	Standard Deviation	Max Error	Per.Alg. Results
n = 10 m = 2	Best 100	206.96	0.532993	2	206
	Pareto 60-30-10	207.16	0.618095	2	206
	Random 100	207.08	0.565685	2	206
n = 20 m = 2	Best 100	490.04	1.456302	6	486
	Pareto 60-30-10	490.28	1.355864	7	486
	Random 100	490.22	1.250143	6	486
n = 30 m = 2	Best 100	769.46	2.383703	12	762
	Pareto 60-30-10	769.88	1.814229	11	762
	Random 100	769.56	2.442481	11	762

Table 8. Factors of Randomly Generated Problems.

Factors	Low (integer)	High (integer)
Range of processing times	10	100
Range of sequence dependent setup times	1	10
Setup times structure	Symmetric	Asymmetric
Number of machines	2	2

The effectiveness of algorithm is tested by using the results of 50 reputations for the alternative approach and problem sizes. The test results are analyzed to find convenient error distribution presenting that initialization approach which uses the best 100 chromosome is efficient in minimizing the error values. Experimental results analyzed using distribution plot indicating error values means distance from optimal solution are presented summarily as Figure 3. As a result of the distribution analysis, it is clarified that the results of the genetic algorithm solution using best 100 initialization approach provides closer results to the optimal solution than that of other approaches. It is, also, greater than other approaches to achieve the optimal results as shown on Table 9.

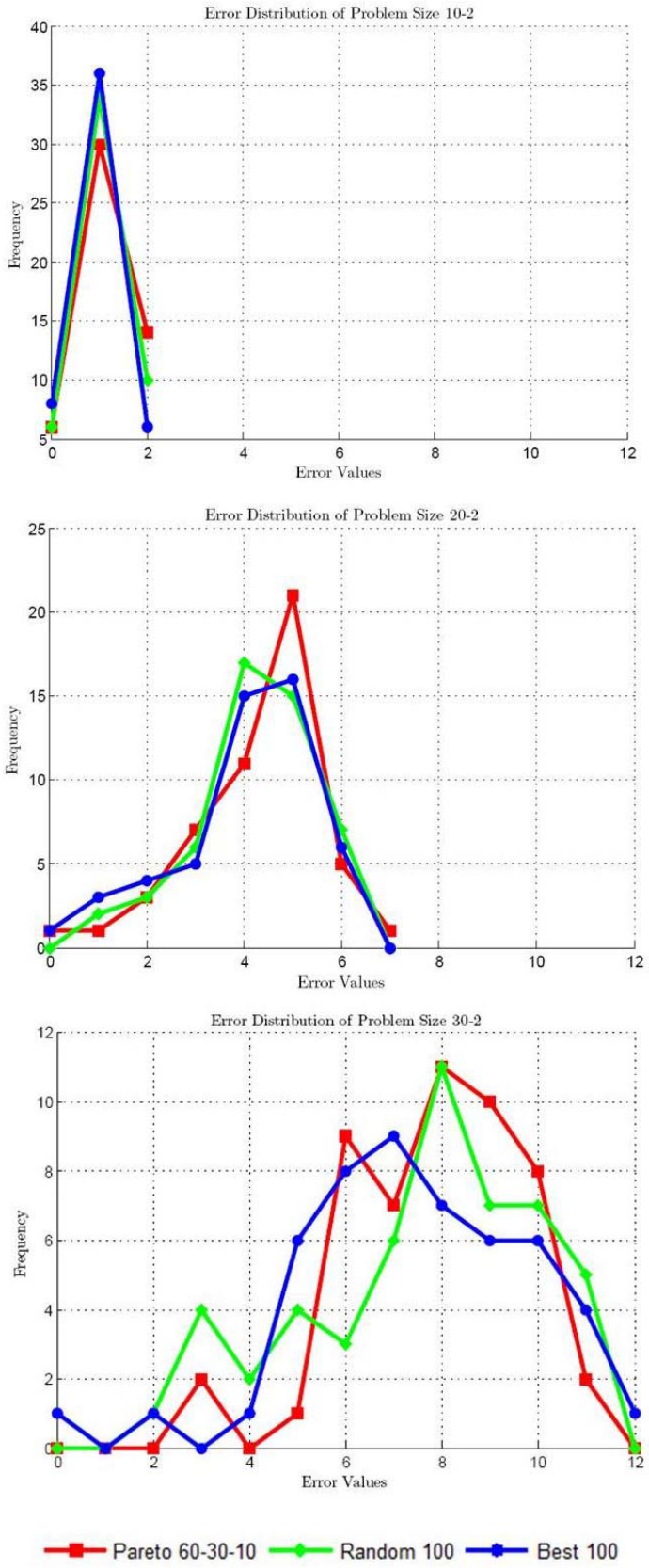

Figure 3. Error Distribution of Problems.

Table 9. Experimental Results.

Problem Size n jobs m machines n jobs	Best 100 Approach	Pareto 60-30-10 Approach	Random 100 Approach
n = 10 m = 2			
1	207	207	208
2	207	207	207
3	207	208	207
4	208	**206**	207
5	207	207	207
6	207	207	207
7	207	207	207
8	208	208	208
9	**206**	207	207
...	...	...	...
50	**206**	**206**	207
Number of Optimal Results	8	6	6
n = 20 m = 2			
1	490	492	490
2	492	492	489
3	492	491	491
4	490	490	487
5	492	488	491
6	490	488	492
7	491	491	490
8	489	490	489
9	488	490	491
...	...	...	...
31	**486**	491	491
...	...	...	...
50	490	490	489
Number of Optimal Results	1	1	0
n = 30 m = 2			
1	769	770	771
2	771	772	770
3	771	770	770
4	770	765	765
5	768	772	768
6	769	768	769
7	771	771	773
8	771	769	771
9	768	770	768
...	...	...	...
26	**762**	769	764
...	...	...	...
50	771	771	767
Number of Optimal Results	1	0	0
Total Optimal Results	10	7	6

7. CONCLUSIONS

The scheduling problem on parallel machines with sequence-dependent setup times and the setup operations are performed by a single server is examined using a genetic algorithm approach. The main purpose is to get minimum makespan. The problem has two components as job sequencing and machine selection. The representation of chromosomes chosen incorporates both of these components on the chromosome. The crossover operator operated on job sequencing is able to effectively and efficiently process to provide solution diversity. Also, three alternative initialization approaches are introduced as the mechanism choosing the best 100 individuals, 100 individuals randomly and 100 individuals using Pareto's rule of 60-30-10 percent among the population which is 10 times greater than the determined population size. The genetic algorithm initialized using these three alternative approaches are experienced with problem sizes consisting of two machines and 10, 20 and 30 jobs.

The optimum results are obtained using a string based permutation algorithm which scans all alternative solutions. As a result, the algorithm is effective and reasonably fast to find close Cmax values to optimum on P2,S|STsd|Cmax scheduling problem. The genetic algorithm can solve the problem faster than the permutation algorithm. Further work in this area may also incorporate factors more than two machines and more complex setup and processing time patterns and it is possible to produce researches about minimization of the deviations from the optimum results.

REFERENCES

[1] Abdekhodaee, A.H., Wirth, A., Gan, H.S., "Scheduling two parallel machines with a single server: the general case", ***Computers & Operations Research*** 33: 994-1009 (2006).

[2] Abdekhodaee, A.H., Wirth, A., "Scheduling parallel machines with a single server: Some solvable cases and heuristics", ***Computers & Operations Research***, 29: 295315 (2002).

[3] Allahverdi, A., Gupta, J.N., Aldowaisan, T., "A review of scheduling research involving setup considerations", ***OMEGA The International Journal of Management Sciences***, 27: 219-239 (1999).

[4] Allahverdi, A., Mg, C.T., Cheng, T.C., Kovalyov, M.Y., "A survey of scheduling problems with setup times or costs", ***European Journal of Operational Research***, 187: 985-1032 (2006).

[5] Cakar, T., Koker, R., Demir, H.I., "Parallel robot scheduling to minimize mean tardiness with precedence constraints using a genetic algorithm", ***Advances in Engineering Software***, 39: 4754 (2008).

[6] Cheng, T.C.E., Gupta, J.N.D., Wang, G., "A review of flowshop scheduling research with setup times", ***Production and Operations Management***, 9: 262-282 (2010).

[7] Guinet, A., "Scheduling sequence dependent jobs on identical parallel machines to minimize completion time criteria", ***Int. J. Prod. Res.***, 31 (7): 1579-1594 (1993).

[8] Huang, S., Chai, L., Zhang, X., "Parallel dedicated machine scheduling problem with sequence-dependent setups and a single server", ***Computers & Industrial Engineering***, 58: 165-174 (2009).

[9] Kellegoz, T., Toklu, B., Wilson, J., "Comparing efficiencies of genetic crossover operators for one machine total weighted tardiness problem", ***Applied Mathematics and Computation***, 199: 590-598 (2008).

[10] Kellegoz, T., Toklu, B., Wilson, J., "Elite guided steady-state genetic algorithm for minimizing total tardiness in flowshops", ***Computers & Industrial Engineering***, 58: 300-306 (2010).

[11] Kurz, M.E., Askin, R.G., "Heuristic scheduling of parallel machines with sequence- dependent set-up times", ***Int. J. Prod. Res.***, 39(16): 3747-3769 (2001).

[12] Sel, C., "Scheduling parallel machines using genetic algorithms with sequence dependent setup times and a single server", Master thesis, ***Kirikkale University,*** (2010).

[13] Sivrikaya, F., Ulusoy, G., "Parallel machine scheduling with earliness and tardiness penalties", ***Computers & Operations Research***, 26:773-787 (1999).

[14] Wilson, A.D., King, R.E., Hodgson, T.J., "Scheduling non-similar groups on a flow line: multiple group setups", ***Robotics and Computer-Integrated Manufacturing***, 20: 505-515 (2004).

[15] Yang, W., Liao, C., "Survey of scheduling research involving setup times", ***International Journal of Systems Science***, 30: 143-155 (2010).