(REFEREED RESEARCH)

A GENETIC ALGORITHM FOR THE UNRELATED PARALLEL MACHINE SCHEDULING PROBLEM WITH JOB SPLITTING AND SEQUENCE-DEPENDENT SETUP TIMES - LOOM SCHEDULING

BÖLÜNEBİLİR VE SIRA BAĞIMLI HAZIRLIK SÜRELİ İŞLER İÇEREN, İLİŞKİSİZ PARALEL MAKİNE ÇİZELGELEME PROBLEMİ İÇİN GENETİK ALGORİTMA-DOKUMA TEZGÂHI ÇİZELGELEME

Duygu YILMAZ EROĞLU[1], H. Cenk ÖZMUTLU[1], Seyit Ali KÖKSAL[2]

[1]Uludag University, Department of Industrial Engineering, Bursa, Turkey
[2]Boyteks Bursa Factory, Bursa, Turkey

Received: 23.05.2013 Accepted: 30.12.2013

ABSTRACT

This paper addresses the unrelated parallel machine scheduling problem with sequence-dependent setup times and job splitting to minimize maximum completion time (makespan). We consider a real-life problem of scheduling looms in a textile industry. Each machine has its own processing times according to the characteristics of the machine as well as the job types. There are machine- and sequence-dependent setup times, and all of the jobs are available at time zero. All of the jobs can be divided into sub-jobs in order to deliver the orders on time. Job splitting has rarely been studied in the literature, especially in the case of parallel machines. Because of the problem's NP-hard structure, heuristics and metaheuristics have been used to solve real-life large-scale problems. Genetic algorithms (GA) are the most preferred approach of this type given their capabilities, such as high adaptability and easy realization. The proposed GA's chromosome representation is based on random keys. The schedule is constructed using a sequence of random key numbers. The main contribution of this paper is to introduce a novel approach that performs job splitting and scheduling simultaneously; to the best of our knowledge, no work has been published with this approach. An important improvement proposed in this paper is assigning the number of sub-jobs dynamically. In addition, the new approach is tested on a real-life problem, and the computational results validate the effectiveness of the proposed algorithm.

Key Words: Loom, Scheduling, Unrelated parallel machine, Job splitting, Sequence-dependent setup times.

ÖZET

Bu çalışmada, bölünebilir ve sıra bağımlı hazırlık süreli işler içeren, ilişkisiz paralel makine çizelgeleme probleminde, en büyük tamamlanma zamanının en küçüklenmesi hedeflenmiştir. Çalışmada, tekstil endüstrisinde, dokuma tezgâhlarının çizelgelenmesi gerçek problemi dikkate alınmıştır. Her makineye özgü, işi tipine ve makine yapısına bağlı olarak değişen işleme süreleri söz konusudur. Makine ve iş sırası bağımlı hazırlık süreleri de mevcuttur ve tüm işler sıfır anında hazırdır. Tüm işler, zamanında teslimatı sağlayabilmek için, alt işlere bölünebilmektedir. İşlerin bölünmesi durumu, özellikle de paralel makinelerde, literatürde nadiren çalışılmıştır. Problemin NP-zor yapısından dolayı, gerçek hayata dair, büyük boyutlu problemlerin çözümü için sezgisel ve meta-sezgisel yöntemler kullanılmaktadır. Genetik algoritmalar (GA), yüksek adaptasyon ve kolay gerçekleşme özelliklerinden dolayı en çok tercih edilen yaklaşımlardır. Önerilen genetik algoritmanın kromozom temsili, rassal anahtar sayılara dayanmaktadır. Çizelge, rassal anahtar sayıların sıralanması ile oluşturulmaktadır. Bu makalenin literatüre ana katkısı, iş bölme ve çizelgelemeyi eş zamanlı yapabilen yeni bir yaklaşım önermesidir. Bilindiği kadarıyla literatürde bu yaklaşımda bir çalışma mevcut değildir. Makalede önerilen diğer önemli geliştirme ise alt iş sayılarının dinamik olarak atanmasıdır. İlave olarak, yeni yaklaşım gerçek hayat probleminde test edilmiş ve hesaplama sonuçları, önerilen algoritmanın etkinliğini doğrulamıştır.

Anahtar Kelimeler: Dokuma tezgâhı, Çizelgeleme, İlişkisiz paralel makine, İş bölme, Sıra bağımlı hazırlık süreleri.

Corresponding Author: Duygu Yılmaz Eroğlu, duygueroglu@uludag.edu.tr, Tel.: +90 505 707 48 59

1. INTRODUCTION

This paper focuses on the problem of scheduling jobs that involve splitting and machine- and sequence-dependent setup times on unrelated parallel machines, with the goal of minimizing the "makespan". Splitting the jobs into sub-jobs and processing them on different weaving machines is necessary because of the due date pressure and competitive challenges of the modern textile industry. If the jobs' processing times are dependent on the assigned machines and there is no relationship among these machines, then the machines are considered to be unrelated (1). The mentioned problem has a set of n jobs ($N = \{1, . . . ,n\}$), which can be split into sub-jobs. The maximum number of sub-jobs has been found to be dependent on the customer order lengths. The mentioned sub-jobs must be processed on one machine in a set $M = \{1, . . . ,m\}$ of m unrelated parallel machines (R_m). The setup times are sequence- and machine-dependent (S_{ijk}). Each machine has its own matrix of setup times, and these matrices are different from each other. The setup time on machine i between jobs j and k is different compared with the setup time between jobs k and j on the same machine.

In scheduling theory, the makespan (C_{max}) is defined as the completion time of the final job (when it leaves the system). A smaller C_{max} implies a higher utilization, which is closely related to the throughput rate of the system. Therefore, reducing C_{max} will lead to a higher throughput rate (2). For that reason, minimization of the "makespan" is the objective of this study. This problem will be referred to as $R_m/S_{ijk}/C_{max}$ with job splitting. Minimizing the makespan on a scheduling problem with m identical parallel machines and sequence-dependent setup times is *NP-hard* (3). Thus, a more complex case of the problem with job splitting and unrelated parallel machines is also *NP-Hard*.

The problem of $R_m/S_{ijk}/C_{max}$ (without job splitting) was studied previously by Eroglu et al. (4). Their proposed genetic algorithm is based on random keys and is tested on a set of problems that were taken from the literature (*SchedulingResearch* 2005 (5)), and their computational results validate the effectiveness of the proposed algorithm.

This paper addresses a more complex case of the $R_m/S_{ijk}/C_{max}$ problem, in which the job splitting property is added to the system definition. The chromosome representation of the proposed *GA* is based on random keys. The proposed algorithm in this study performs job splitting and scheduling simultaneously, where to the best of our knowledge, no work has been published on an algorithm with these properties. The algorithm is tested on the real life-scheduling problem of weaving looms in the Boyteks Textile Industry and Trade Co./Bursa, which is a trademark of Boydak Holding, one of the largest organizations in Turkey. This company produces upholstery fabric for furniture manufacturers. With one of the highest production capacities of jacquard woven and knitted mattress ticking fabrics in the world, Boyteks produces high quality upholstery and curtain fabrics in its facilities in the Bursa Organized Industrial Zone, with a covered area of 23,700 m². The results demonstrated that proposed *GA* performed in an acceptable *CPU* time with a better makespan value than to that of the actual schedule that was used by the company.

This paper is organized as follows. Section 2 is a literature review. Section 3 describes the problem. In Section 4, the proposed genetic algorithm is given. In Section 5, the results of the numerical example are reported. Section 6 concludes the paper.

2. LITERATURE REVIEW

The interest in scheduling problems with setup times began in the mid-1960s. These types of problems have received continuous interest from researchers since then. Allahverdi et al. (6) presented a survey of scheduling problems with setup times or costs. The paper classified scheduling problems into those with batching and non-batching considerations, and with sequence-independent and sequence-dependent setup times. It also classified the literature according to shop environments, including a single machine, parallel machines, flow shop, no-wait flow shop, flexible flow shop, job shop and open shop. Machine setup time is a significant factor for production scheduling in manufacturing environments, and sequence-dependent setup times have been investigated by researchers. Zhu and Wilhelm (7) presented a review of scheduling problems that involves sequence-dependent setup times (costs). Li and Yang (8) presented a review on non-identical parallel-machine scheduling research in which the total weighted completion times are minimized. Models and relaxations are classified in this paper, and heuristics and optimizing techniques are surveyed for the problems. Lin et al. (9) studied research that compares the performance of various heuristics for unrelated parallel machine scheduling problems. They proposed a metaheuristic, and computational results showed that the proposed metaheuristic outperformed other existing heuristics for each of the three objectives, minimized makespan, total weighted completion time and total weighted tardiness, when run with a parameter setting that is appropriate for the objective.

Because of the problem complexity, it is general practice to find an appropriate heuristic rather than an optimal solution for the parallel-machine scheduling problem. Park et al. (10) used a neural network and heuristic rules to schedule jobs with sequence-dependent setup times on parallel machines. To calculate the priority index of each job, they utilized a neural network, and their computational results showed that the proposed approach outperformed the Lee et al (11) original *ATCS* (Apparent Tardiness Cost with Setups) and a simple application of *ATCS*. Hop and Nagarur (12) focused on scheduling problems of n printed circuit boards (*PCB*s) for m non-identical parallel machines, and a composite genetic algorithm was developed to solve this multi-objective problem. Test results of the proposed methodology showed that the solutions were efficient and were obtained within a reasonable amount of time. Behnamian et al. (13) compared makespan results solved by ant colony optimization (*ACO*), Variable Neighborhood Search (*VNS*), Simulated Annealing (*SA*) and the *VNS* hybrid algorithm for the problem of parallel machine scheduling problems with sequence-dependent setup times. Yang (14) proposed an evolutionary simulation optimization approach for solving the parallel-machine scheduling problem. The proposed methodology's findings were benchmarked against lower-bound solutions, and efficient results were obtained. Balin (15) attempted to adapt a *GA* to the non-identical parallel machine scheduling problem and proposed an algorithm with a new crossover operator and a new optimality criterion. The new algorithm was tested on a numerical example by implementing it in simulation software. The results showed that, in addition to its high computational speed for a larger-scale problem, the *GA* addressed the non-identical parallel machine scheduling problem of minimizing the makespan. Keskinturk et al. (16) aimed to minimize the average relative percentage of imbalance and used the *ACO* algorithm for load balancing in parallel machines with sequence-dependent setup times. The results of tests on various random data showed that the *ACO* outperformed the *GA* and heuristics.

In this section, some of the previous studies on the unrelated parallel machine scheduling problem with the objective of minimizing the makespan are discussed. All of these studies considered the following assumptions:

- machine-dependent and job sequence-dependent setup times,
- all of the jobs are available at time zero.

A meta-heuristic *Meta-RaPS* was introduced by Rabadi et al. (17), and its performance was evaluated by comparing its solutions to those obtained by existing heuristics. The results of the meta-heuristic showed that the solutions were efficient. A two-stage Ant Colony Optimization (*ACO*) algorithm was proposed by Arnaout et al. (18). The performance of this meta-heuristic was evaluated by using the benchmark problems, and the solutions were found to be efficient. Another method was suggested by Chang and Chen (19) for the same *NP-Hard* problem. A set of dominance properties were developed, including inter-machine and intra-machine switching properties, which are necessary conditions of job sequencing orders in a derived optimal schedule. They also introduced a new meta-heuristic by integrating the dominance properties with a genetic algorithm (*GADP*). The performance of this meta-heuristic was evaluated by using benchmark problems from the literature, and the solutions were efficient. Vallada and Ruiz (1) proposed a genetic algorithm that includes the crossover operator with a limited local search as well as a fast local search procedure; this method was tested on both small and large problem sets and outperformed the other evaluated methods. Eroglu et al. (4) proposed a genetic algorithm that was based on random keys. To present the performance of the algorithm, the same problem set (5) was used. The results showed that the *GA,* which is the foundation of this study, outperformed the other algorithms.

The jobs that are considered in this paper can be split into sub-jobs, a feature that is very seldom studied in the literature. Studies can be categorized as splitting the job into sub-jobs of discrete units or continuous units. Some studies that involve discrete units are as follows: Kim et al. (20) focused on the dicing of semiconductor wafer manufacturing, which is the major bottleneck operation of the whole process. They proposed a simulated annealing algorithm for the problem of allotting work parts of *L* jobs into *M* parallel unrelated machines, where a job is referred to a lot composed of *N* items. Setup times were job sequence-dependent. The proposed *SA* method outperformed a neighborhood search method in terms of the total tardiness. Kim et al (21) suggested a two-phase heuristic algorithm for the problem of scheduling a drilling process in the *PCB* manufacturing system. It was assumed that a job can be split into a discrete number of sub-jobs and that they would be processed on identical parallel machines independently. In the first phase of the algorithm, an initial sequence is generated by existing heuristic methods for the parallel machine scheduling problem. In the second phase, each job was split into sub-jobs, and then jobs and sub-jobs were rescheduled on the machines by using a certain method. The performance of the suggested algorithm was proved by the results of the computational experiments, which performed better than an existing method. Shim and Kim (22) also focused on *PCB* manufacturing system bottleneck operations of the drilling process. For the problem of scheduling jobs that can be split into sub-jobs, they developed several dominance properties and lower bounds and then suggested a branch and bound algorithm using them. The suggested algorithm solved problems of moderate size in a reasonable amount of computational time. Xing and Zhang (23) proposed a heuristic algorithm for the $P/split/C_{max}$ problem and analyzed the worst case performance of the algorithm. Yalaoui and Chu (3) considered a simplified real-life identical parallel machine scheduling problem with sequence-dependent setup times and job splitting to minimize the makespan. The proposed method was composed of two phases. In the first phase, the problem was reduced to a single machine scheduling problem and was transformed into a Traveling Salesman Problem (*TSP*), which could efficiently be solved by using Little's method. In the second phase, a derived initial solution was improved in a step-by-step manner, accounting for the setup times and job splitting. Tahar et al. (24) proposed a new method, which is an improvement of the method proposed by Yalaoui and Chu (3). For the problem of splitting the job into continuous units, Serafini (25) studied the scheduling problem of looms in the textile industry. Jobs might be independently split over several specified machines, and preemption was allowed. It was assumed that there were uniform parallel machines and that there were no setup times. For the mentioned problem, heuristic algorithms were proposed for the objective of minimizing the maximum weighted tardiness.

3. PROBLEM DESCRIPTION

In this paper, we consider the scheduling problem for weaving looms in the Boyteks Textile Industry and Trade Co. (www.boyteks.com), which is a textile manufacturing company that produces upholstery fabric in Turkey. Changing technology forces the firms to invest in new equipment. Machine and customer order diversity make the machine environment unrelated. To respect the committed delivery dates, demands must be divided into smaller sub-lots. The main cause of sequence-dependent setup times is the composition change between orders. The setup times are also machine dependent. Therefore, our aim is to decide on the number of sub-lots and schedule them on unrelated parallel machines with machine- and sequence-dependent setup times simultaneously, with the objective of minimizing the makespan. To the best of our knowledge, there is no previous algorithm that addresses the problem considered in this study.

The mentioned firms' orders were analyzed over a period of four years, and some of the implications are the following:

- 92% of the total orders are equal to or smaller than ten thousand meters. These orders can be considered to be small orders, which have also taken the largest amount of study time for the planning. Small orders will be this paper’s main research area.
- The splitting process for the orders is usually performed by experienced employees. After analyzing a number of sub-lots for small orders, the following data have been found.
  - Orders smaller than 500 meters have not been split,
  - Orders between 500 and 1,000 meters have been split into a maximum of 2 sections (sub-lots),
  - Between 1,000 and 2,000 meters have been split into a maximum of 3 sub-lots,
  - Between 2,000 and 5,000 meters have been split into a maximum of 4 sub-lots,
  - Between 5,000 and 10,000 meters have been split into a maximum of 5 sub-lots.

In our algorithm, splitting will be made according to the length of the main order. For example, if the order length is between 5,000 and 10,000 mt., then the number of sub-lots can be 1, 2, 3, 4 or 5. The developed algorithm will create the schedule and find the number of necessary sub-lots simultaneously. Our algorithm, the solution procedure and the comparisons will be explained in the following sections.

4. PROPOSED GENETIC ALGORITHM

A *GA* is a searching technique that mimics the natural evolution processes in living organisms. A genetic representation (a string of symbols) for each solution in a population is one of the important issues. The string is referred to as a chromosome, and the symbols are referred to as genes. After generating of an initial population and determining the fitness function values for each chromosome, the *GA* manipulates the selection process by operations such as reproduction, crossover and mutation. *GA*s are a type of algorithm that was introduced in the 1970s by Holland (26). As the searching technique of genetic algorithms (*GA*s) (27) became popular in the mid-1980s, many researchers started to apply this heuristic to scheduling problems.

In the genetic algorithm, the representation of the problem parameter set is important. The encoding must be designed to utilize the algorithm's ability to transfer information between chromosome strings efficiently and effectively (26). To qualify our encoding scheme, the permutation of jobs in this research is shown through random keys. Each job has a random number between 0 and 1. These random keys show the relative order of the jobs for each machine. In addition, each chromosome also carries the number of sub-jobs. A detailed description of the developed *GA* is provided in the following subsections.

4.1 Encoding Scheme

A chromosome is presented as a string of random keys. Figure 1 illustrates a sample chromosome. In the first section of the chromosome, the string contains *n* (number of jobs) sections. Each section is further divided into *m* (number of machines) sections. Each section for machines is divided into sub-sections (genes). The number of sub-sections for a related machine is determined according to the maximum number of sub-jobs for a related job. The sequence of jobs will be determined by random key numbers (generated between 0-1) of each gene. In the second section, the string contains *n* (number of jobs) sections. For each section, a random number will be generated between one and the maximum number of sub-jobs for each job, to determine the number of sub-jobs. The chromosome structure of the example is shown in Figure 1. For this chromosome structure, we have 2 machines and 3 jobs. The maximum number of sub-jobs for Job_1, Job_2, and Job_3 are 3, 2 and 1, respectively. For Job_1, because the number of sub-jobs has the value of 2 in the second section of the chromosome, the smallest two random numbers will be selected from the first section of the chromosome among all of the numbers generated for Job_1. The selected values (0.2 and 0.3 is for Job_1) are given in bold in Figure 1. It is clear that the first and second sub-jobs of Job_1 will be processed on $Machine_1$. Similarly, the number of sub-jobs is 2 for Job_2, and they will be processed on $Machine_1$ and $Machine_2$. The number of sub-jobs is 1 for Job_3, and it will be processed on $Machine_2$.

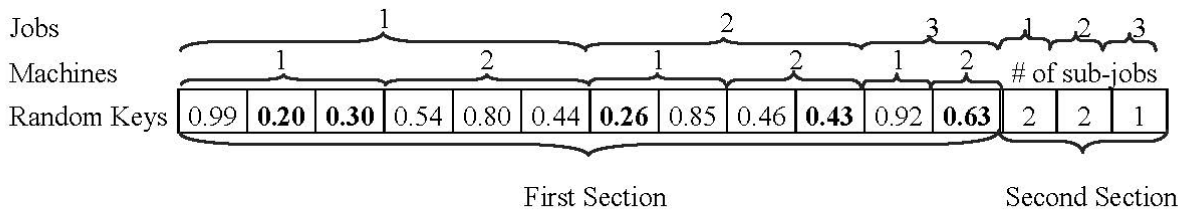


Figure 1. Representation of a chromosome in the *GA*

4.2 Fitness Function (C_{max})

As mentioned earlier, there are machine-dependent processing times and sequence-dependent setup times.

- p_{ij}: the processing time of Job_j, $j \in N$ at Machine$_i$, $i \in M$.
- S_{ijk}: machine-based sequence-dependent setup time on Machine$_i$, $i \in M$ when processing Job_k $k \in N$ after having processed Job_j, $j \in N$.

In the case of considering the three-job two-machines scheduling problem, for example, Table 1 shows the processing times for Machine$_1$ and Machine$_2$. Table 2 and Table 3 are the setup times of Machine$_1$ and Machine$_2$, respectively, for the mentioned jobs. According to the chromosome that was shown in Fig. 1, Machine$_1$ will process two sub-jobs of Job_1 (Sub-job$_{1.1}$, Sub-job$_{1.2}$) and one sub-job of Job_2 (Sub-job$_{2.1}$). Machine$_2$ will process one sub-job of Job_2 (Sub-job$_{2.2}$) and one sub-job of Job_3 (Sub-job$_{3.1}$). The sequence of jobs on Machine$_1$ will be determined according to random key numbers (0.20, 0.26 and 0.30). Increasing the arrangement of these random numbers will designate Machine$_1$'s job sequence. As a result, the sequence of sub-jobs is Sub-job$_{1.1}$, Sub-job$_{2.1}$ and Sub-job$_{1.2}$ on Machine$_1$. If a job is split into sub-jobs, the processing time of the required sub-job on the selected machine can be calculated by dividing a job's processing time on the related machine to the number of sub-jobs. The completion time of Machine$_1$ is 130, and the completion time of Machine$_2$ is 78, according to the p_{ij} and S_{ijk} values. Then, C_{max} will be 130. The mentioned chromosome's schedule and makespan value can be seen in Figure 2. The population size will determine the number of different chromosomes and the C_{max} values.

Table 1. The total processing times for the jobs on the machines

p_{ij}	Job_1	Job_2	Job_3
Machine$_1$	90	60	30
Machine$_2$	95	70	40

Table 2. Setup times for Machine$_1$

S_{ijk}	Job_1	Job_2	Job_3
Job$_1$	-	8	6
Job$_2$	2	-	4
Job$_3$	3	7	-

Table 3. Setup times for Machine$_2$

S_{ijk}	Job_1	Job_2	Job_3
Job$_1$	-	2	6
Job$_2$	4	-	3
Job$_3$	2	1	-

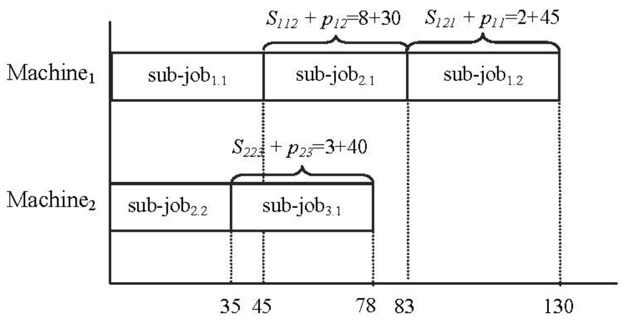


Figure 2. Resulting schedule and C_{max} for the chromosome in Fig. 1.

4.3. Genetic Operators

After generating an initial population, selection, crossover and mutation will be iteratively used to search for the best solution.

- Selection: Chromosomes will be selected into the mating pool based on the random selection method (28). In this method, moms and dads are randomly chosen from the population.
- Crossover: The crossover operator, which is applied according to the value of the crossover rate, P_c, is a method for sharing information between chromosomes. Single point crossover will be used as a crossover operator in our algorithm. It randomly chooses the crossing point and exchanges the genes between two parents in order to create offspring. The crossover operator will be used for the first and second section separately. Figure 3 illustrates this operation.

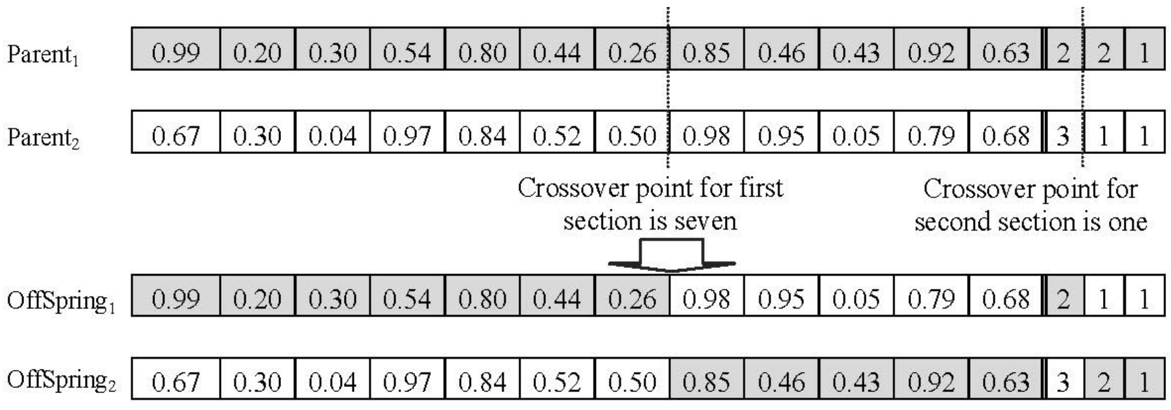


Figure 3. Crossover operation: Two parents mate in order to produce two offspring

- Mutation: The mutation operator is used to prevent convergence to a local optimum. In our algorithm, a mutation is performed as follows. Chromosomes to which a mutation operation will be applied are randomly selected from the population according to the mutation rate (P_m). The value of the randomly selected gene of the chromosome will be replaced with a new random number. This operation will be applied to all of the selected chromosomes. In this way, chromosomes with new schedules and makespan values can be obtained. The fitness value could be better, worse or the same after applying this operator. Mutation operation is demonstrated in Figure 4. This procedure will cause of new makespan value of 127.5, which is better than the previous makespan value.

5. NUMERİCAL EXAMPLE

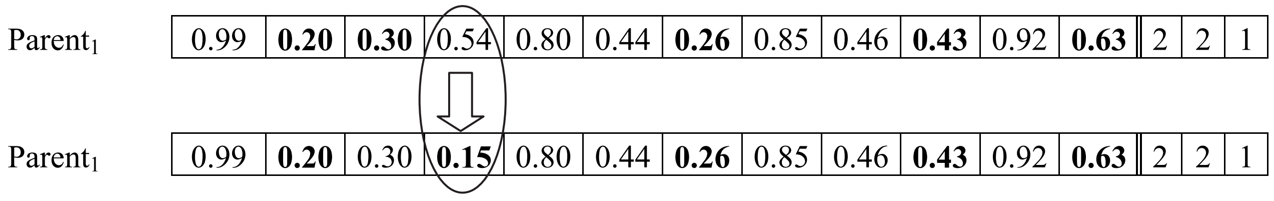
5.1 Experimental results and parameter values

The parameter configuration of *GA* is done by the Design-of- Experiment approach. The levels of the three factors are listed in the Table 4. Each parameter combination is replicated five times.

The graphic which is the upper-right side in the figure Figure 5 shows that the minimum C_{max} value is achieved on the 0.5 P_c line's bottom point that corresponds to the P_m value of 0.2. Similarly, the other graphic which is the lower-left side in the Fig. 5, shows that the minimum C_{max} value is occurred on 0.2 P_m line's bottom point that corresponds to P_c value of 0.5. Interaction plot in Fig. 5 indicates that there is significant interaction between the probability of crossover (P_c) and probability of mutation (P_m). When we set the probability of crossover and probability of mutation at the level of 0.5 and 0.2, respectively; the algorithm will yield a better solution quality. The population size (P_{size}) is set to 100 according to the graphic which is the upper-left side on the main effect plot in Figure 6. The mentioned graphic shows that, the minimum C_{max} value is achieved in the point that corresponds to the 100 value for the P_{size} parameter.

Table 4. Experimented parameter values of proposed genetic algorithm

Parameter	Value
Population size (P_{size})	80; 100
Probability of crossover (P_c)	0.5; 1.0
Probability of mutation (P_m)	0.2; 0.5



Parent$_1$	0.99	**0.20**	**0.30**	0.54	0.80	0.44	**0.26**	0.85	0.46	**0.43**	0.92	**0.63**	2	2	1
Parent$_1$	0.99	**0.20**	0.30	**0.15**	0.80	0.44	**0.26**	0.85	0.46	**0.43**	0.92	**0.63**	2	2	1

Figure 4. Mutation operation

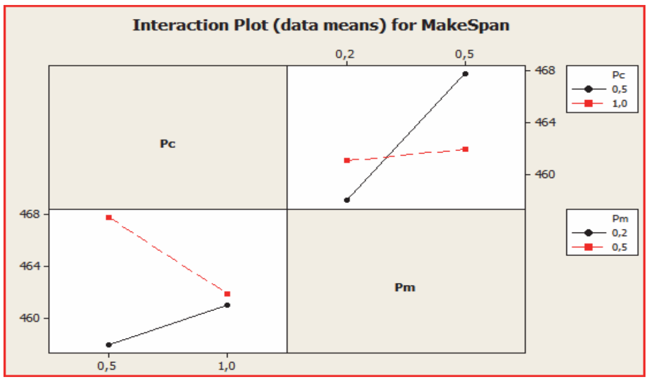


Figure 5. Interaction plot of P_c and P_m

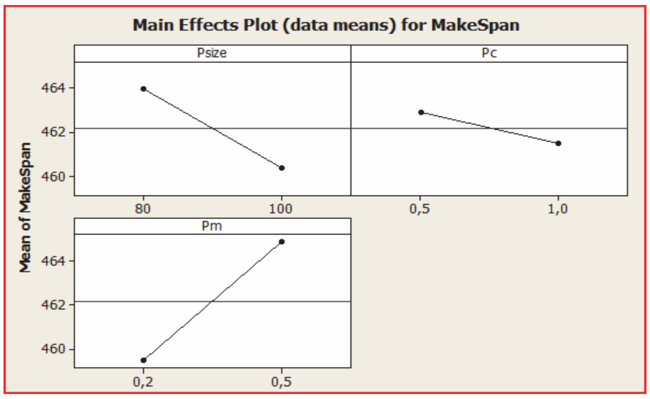


Figure 6. Main effects plot of the three factors

5.2 Computational results

To test the proposed algorithm, small orders that were explained in Section 3 were studied for one month. To satisfy the repeatability requirements, order lengths, processing times, and setup time distributions are determined according to the real order data. Minitab 16 (29) is used to find the distribution functions and their parameters that best fits to the data. The distribution of small order lengths is consistent with a lognormal distribution function, where the minimum, the location and the scale parameters are equal to 1.00, 6.22 and 1.88, respectively with 95% confidence level. There are 111 small orders. The distribution of the processing times of each job on the machines is consistent with a uniform distribution minimum value of 0.0024 and a maximum value of 0.1248 hours per unit length. There are 75 machines, and 23% of these machines are armure weaving machines. A total of 35% of the job-job-machine setup-time values of the armure weaving machines are consistent with the exponential distribution, with a minimum value of 0.37 and a scale parameter value of 10.49 hours respectively with 95% confidence level. Here, 77% of the 75 machines are jacquard weaving machines, and 42% of the job-job-machine setup time values of the jacquard weaving machines are consistent with an exponential distribution, with a minimum value of 0.18 and a scale parameter value of 18.85 hours respectively with 95% confidence level. It should be noted that each job can be produced by each machine (armure or jacquard), but the setup times and processing times will be different.

The proposed algorithm is coded in the C# language and run on a computer with an Intel Core 2 Duo processor running at 2.26 GHz. The probability of crossover and the probability of mutation and population size have been selected to be 0.5, 0.2 and 100 respectively according to experimental results. Five replications have been performed; the makespan values are calculated, and the elapsed times are registered for every 100 generations up to 1000 generations. Figure 7 shows the average makespan values and elapsed times of the studied problem for the analyzed generations. It is easy to conclude from Figure 7 that after 500 generations, there is no remarkable improvement in the makespan. Thus, the number of generations can be set to a value of 500. The results of the computations show that the suggested algorithm could find solutions for problems with 75 machines and 111 jobs in a reasonable amount of *CPU* time.

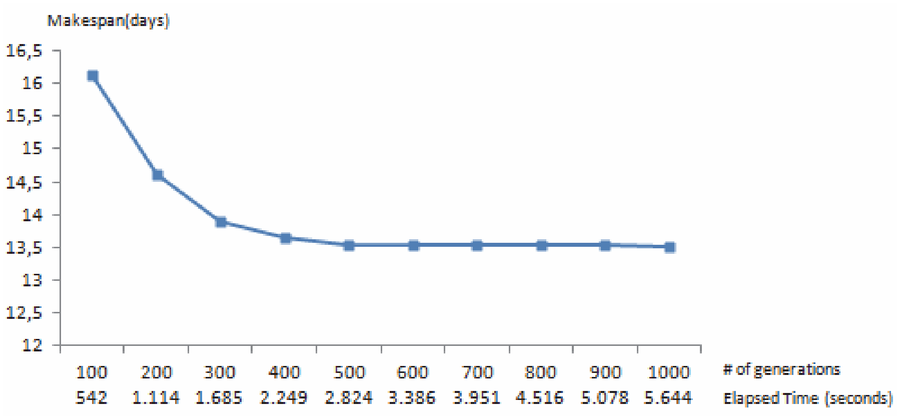


Figure 7. Average makespan values and elapsed times for the determined number of generations

The makespan value of the mentioned production scheduling problem of Boyteks is 19.74 days in the real system. It is clear that our algorithm's makespan result of 13.52 days is better than the existing values. As a result, the proposed genetic algorithm clearly outperforms, by a considerable margin, the existing method.

6. CONCLUSIONS

In this paper, we have proposed a genetic algorithm for the unrelated parallel machine scheduling problem with machine- and sequence-dependent setup times and with the job splitting property, to minimize the makespan. To our knowledge, in the literature, there is no solution algorithm for the problem considered in this study. We have developed an efficient genetic algorithm for a real-life scheduling problem that involves looms in the textile industry. The algorithm decides the number of sub-jobs for each order and makes the schedule simultaneously. The result of the computations showed that the suggested algorithm could find solutions for problems with 75 machines and 111 jobs in a reasonable amount of *CPU* time. According to the results, the proposed *GA* outperforms the existing system in terms of the resulting makespan.

It will be interesting to extend this work by including local heuristics in the proposed algorithm, to investigate whether an improvement can be accomplished. In addition, assigning sub-jobs to the machines that are closer to each other is another aspect that can be considered in the future. Therefore, an application to a multi-objective fitness function would be a new research area to be explored. Finally, other constraints, such as priority conditions, could be incorporated into future research.

ACKNOWLEDGMENTS

The authors thank the Boyteks Textile Industry and Trade Co. company for providing the case study.

REFERENCES

1. Vallada, E., & Ruiz, R. (2011). "A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times", *European Journal of Operational Research, 211*, 612–622.
2. Kashan, A.H., Karimi, B., & Jolai,F. (2010). "An effective hybrid multi-objective genetic algorithm for bi-criteria scheduling on a single batch processing machine with non-identical job sizes", Engineering *Applications of Artificial Intelligence, 23,* 911–922.
3. Yalaoui, F., & Chu, C. (2003). "An efficient heuristic approach for parallel machine scheduling with job splitting and sequence-dependent setup times", *IIE Transactions, 35(2),* 183–190.
4. Eroglu, D.Y., Ozmutlu H.C., & Ozmutlu S. (2013). "Genetic algorithm with local search for the unrelated parallel machine scheduling problem with sequence-dependent setup times", *International Journal of Production Research, Manuscript submitted for publication.*
5. SchedulingResearch. (2005). Accessed June 07, 2012 from http://SchedulingResearch.com
6. Allahverdi, A., Ng, C.T., Cheng, T.C.E., & Kovalyov, M.Y. (2008). "A survey of scheduling problems with setup times or costs", *European Journal of Operational Research, 187*, 985–1032.

7. Zhu, X., Wilhelm, & W.E. (2006). “Scheduling and lot sizing with sequence-dependent setup: A literature review”, *IIE Transactions, (38),* 987–1007.
8. Li, K., & Yang, S. (2009). “Non-identical parallel-machine scheduling research with minimizing total weighted completion times: Models, relaxations and algorithms”, *Applied Mathematical Modelling, 33*, 2145–2158.
9. Lin, Y.K., Pfund, M.E., & Fowler, J.W. (2011). “Heuristics for minimizing regular performance measures in unrelated parallel machine scheduling problems”, *Computers & Operations Research, 38*, 901–916.
10. Park, Y., Kim, S., & Lee, Y.H. (2000). “Scheduling jobs on parallel machines applying neural network and heuristic rules”, *Computers & Industrial Engineering, 38*, 189-202.
11. Lee, Y. H., Bhaskaran, K., & Pinedo, M. (1997). “A heuristic to minimize the total weighted tardiness with sequence-dependent setups”, *IIE Transactions, 29,* 45-52.
12. Hop, N.V., & Nagarur, N.N. (2004). “The scheduling problem of PCBs for multiple non-identical parallel machines”, *European Journal of Operational Research, 158*, 577–594.
13. Behnamian, J., Zandieh, M., & Ghomi, S.M.T.F. (2009). “Parallel-machine scheduling problems with sequence-dependent setup times using an ACO, SA and VNS hybrid algorithm”, *Expert Systems with Applications, 36*, 9637–9644.
14. Yang, T. (2009). “An evolutionary simulation–optimization approach in solving parallel-machine scheduling problems – A case study”, *Computers & Industrial Engineering, 56*, 1126–1136.
15. Balin, S. (2011). “Non-identical parallel machine scheduling using genetic algorithm”, *Expert Systems with Applications, 38,* 6814–6821.
16. Keskinturk, T., Yildirim, M.B., & Barut, M. (2012). “An ant colony optimization algorithm for load balancing in parallel machines with sequence-dependent setup times”, *Computers and Operations Research, 39*, 1225–1235.
17. Rabadi, G., Moraga, R., & Al-Salem, A. (2006). “Heuristics for the Unrelated Parallel Machine Scheduling Problem with Setup Times”, Journal *of Intelligent Manufacturing*, 17, 85–97.
18. Arnaout, J.P., Rabadi, G., & Musa, R. (2010). “A two-stage Ant Colony Optimization algorithm to minimize the makespan on unrelated parallel machines with sequence-dependent setup times”, *Journal of Intelligent Manufacturing*, 21, 693-701.
19. Chang, P.C., & Chen, S.H. (2011). “Integrating dominance properties with genetic algorithms for parallel machine scheduling problems with setup times”, *Applied Soft Computing, 11*, 1263–1274.
20. Kim, D.W., Kim, K.H., Jang, W., & Chen, F.F. (2002). “Unrelated parallel machine scheduling with setup times using simulated annealing”, *Robotics and Computer Integrated Manufacturing, 18*, 223–231.
21. Kim, Y.D., Shim, S.O., Kim, S.B., Choi, Y.C., & Yoon, H.M. (2004). “Parallel machine scheduling considering a job-splitting property”, *International Journal of Production Research, 42*, 4531–46.
22. Shim, S.O., & Kim, Y.D. (2008). “A branch and bound algorithm for an identical parallel machine scheduling problem with a job splitting property”, *Computers & Operations Research, 35*, 863 – 875.
23. Xing, W., & Zhang, J. (2000). “Parallel machine scheduling with splitting jobs”, *Discrete Applied Mathematics, 103*, 259-269.
24. Tahar, D.N., Yalaoui, F., Chu, C., & Amodeo, L. (2006). “A linear programming approach for identical parallel machine scheduling with job splitting and sequence-dependent setup times”, *International Journal of Production Economics, 99*, 63–73.
25. Serafini, P. (1996). “Scheduling jobs on several machines with the job splitting property”, *Operations Research, 44*, 617–628.
26. Holland, J.A. (1975). *Adaptation in natural and artificial systems*: University of Michigan, Ann Arbor.
27. Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning:* Addison-Wesley, Reading.
28. Haupt, R. L., & Haupt S. E. (2004). *Practical Genetic Algorithms,* New Jersey: Second edition, John Wiley & Sons Inc.
29. Minitab 16, Statistical Software (2010). State College, PA: Minitab, Inc. (www.minitab.com).