PARALLEL MACHINE SCHEDULING CONSIDERING JOB SPLITTING AND MACHINE ELIGIBILITY

by

Gamze Koyuncu

B.S., Systems Engineering, Yeditepe University, 2006

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Industrial Engineering
Boğaziçi University
2009

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my thesis supervisor, Assoc. Prof. Ali Tamer Ünal, for his continuous support, guidance, encouragement and motivation throughout this study. Working with him was a privilege. I was lucky to have one of the best faculty members with his strong technical background and kind personality. I am thankful to his invaluable comments and always being there to coach me through difficulties.

I would also wish to express my sincere thanks to Assoc. Prof. Necati Aras and Prof. Tülin Aktin for their suggestions as members of my thesis committee.

I would like to thank to all members of my family for their never-ending support, encouragement and patience.

I would like to thank my husband Osman Nihat Koyuncu for his enlightening ideas helping me during this work.

Also I would like to thank Engin Durmaz for his support.

Finally, I would like to thank TUBITAK very much for the financial support during my education.

ABSTRACT

PARALLEL MACHINE SCHEDULING CONSIDERING JOB SPLITTING AND MACHINE ELIGIBILITY

In this study, we investigate unrelated parallel machine problem with total tardiness objective. The properties of the problem are job splitting, family dependent setup structure and machine eligibility. Job splitting means that jobs can be splitted to be produced on different machines and in different times. Family dependent setup means that a setup is needed before producing a particular family if it is preceded by another family. Machine eligibility means that jobs can't be produced on all machines, but only the ones that are appropriate for producing them.

We propose a heuristic solution method consisting of three phases. In the first phase, jobs belonging to a family are combined into job batches. When making this aggregation, in order to decide the point to stop aggregation, we have two control parameters. These control parameters do not need to be the same for each family. After finishing Phase-1, generated job batches are used as inputs to phase-2. In phase-2, a new time structure is created based on the due dates of these aggregate jobs. Also in phase-2, an aggregate planning model is constructed and solved yielding production quantities in time buckets. These production quantities are used in Phase-3 for creating job batches which constructs schedule. For examining the performance of our heuristic, we compare it with a heuristic in the literature in which control parameters used in the first phase are the same for all families. According to experimentation results, our heuristic out performs the existing heuristic.

ÖZET

İŞ BÖLME VE MAKİNA SEÇİM KRİTERLİ PARALEL MAKİNA ÇİZELGELEME

Bu çalışmada birbirinden farklı paralel makinalarda toplam gecikmeyi en aza indirmeye yönelik bir problem ele alınmıştır. Problemin işin bölünebilmesi, aile bazında makina hazırlama süresi, makina seçilebilirlik gibi özellikleri vardır. İşin bölünebilmesi, bu işin farklı makinalarda farklı zamanlarda yapılabilmesi anlamına gelmektedir. Aile bazında makina hazırlama süresi, eğer belirli bir aile başka bir aileden sonra üretiliyor ise, makina hazırlama süresine gerek duyulması anlamına gelmektedir. Makina seçme özelliği işlerin her makinada üretilememesi, sadece onları üretmeye uygun makinalar tarafından üretileblmeleri anlamına gelmektedir.

Üç fazdan oluşan bir sezgisel yaklaşım metodu önermekteyiz. İlk fazda bir aileye ait olan işler iş grubu halinde bir araya getirilmektedir. İşleri bir araya getirirken, bir araya getirmeyi ne zaman durduracağımızı belirlemek için iki kontrol parametremiz mevcuttur. Bu kontrol parametrelerinin tüm aileler çin aynı olması gerekmemektedir. İlk faz sonucunda oluşturulan iş grupları ikinci fazın girdisi olarak kullanılmaktadır. Bu ürün gruplarının termin zamanları kullanılarak ikinci fazda yeni bir zaman yapısı oluşturulmaktadır. Ayrıca ikinci fazda bütüncül planlama modeli oluşturulup çözülmektedir. Bunun sonucunda, oluşturulan zaman yapılarındaki üretim miktarları bulunmaktadır. Bulunan bu üretim miktarları, üçüncü fazda çizelgeyi meydana getiren iş grupları oluşturmak için kullanılır. Sezgisel yöntemimizin performansını değerlendirmek için yazında var olan , birinci fazda kullandığımız kontrol parametrelerinin tüm aileler için sabit tutulduğu sezgisel metod ile karşılaştırdık. Deney sonuçlarına göre, önerdiğimiz sezgisel metodumuz diğer sezgisel metoddan daha iyi sonuç vermektedir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

d	Due Date
D	Demand
T	Tardiness
T_{max}	Maximum tardiness
LP	Linear Programming
MD	Maximum due date difference
MIP	Mixed Integer Programming
PPS	Production per setup
ResTB	Resource time bucket

1. INTRODUCTION

Scheduling is a very efficient device for optimizing use of any type of resources such as manpower, machines and facilities while considering some requirements, constraints and objectives. Scheduling consists of making systematic planning and prioritizing tasks which in turn helps companies to compute in today's competitive environment. Since customer satisfaction turns out to be one of the most significant goal in manufacturing, it is really important to complete jobs no later than their due dates which are achieved by scheduling. Scheduling is a very crucial aspect of manufacturing industry, since companies become less tolerant to delays in production.

Parallel machine environment is very common in manufacturing industry. Parallel machines can be either identical or unrelated machines. Unrelated parallel machines case is the most general one since the features of machines differ a lot in production lines due to differences in their technology, or there may exist machines which are processing more than one kind of jobs with different speeds. So the production line may be very fast when producing one kind of job, while it is too slow after making regulations and processing the other kind of job. Some examples of companies where we can see the parallel machine production environment in Turkey are: Trakya Otocam Company, Vestel TV Assembly Company and AKSA Acrylic Company. Due to these reasons we studied the unrelated parallel machine environment in this study.

Also we considered set up times which are needed between processing of jobs for preparation of the production line for the coming job. Setup times may be sequence dependent between jobs or they can be based on the similarity between the jobs. We call a set of products with similar setup characteristics a "family of products", and this type of setup structure is referred to as "family setup". There are two types of family dependent setup which are minor and major setup while the former is the setup required between production of jobs in one family due to small differences of their production requirements and this kind of setup is negligible in most of the cases and the latter one is the setup time between processing of jobs belonging to different families.

Other important aspects of our problem are job splitting and machine eligibility considerations. Job splitting property means that we can produce some part of a particular job in one machine while processing the other parts on other machines. These processing can be done on different times or at the same time. Machine eligibility constraint is a very important factor for determining on which machine to produce since every machine is not capable of processing all jobs.

In this study, we consider unrelated parallel machine scheduling with family dependent set up, job splitting and machine eligibility considerations. The objective function is to minimize the total tardiness.

This study is organized as follows: Chapter 2 gives a detailed overview of the machine scheduling literature involving related considerations to our problem. It discusses various solution methodologies like exact algorithm, heuristics, simulated annealing, genetic algorithm and tabu Search. Chapter 3 presents the formal problem definition, the mathematical model. Later, in Chapter 4, the heuristic that is suggested will be presented in detail. In Chapter 5, details of simulated annealing approach used in this work are given. In Chapter 6, implementation of the algorithm is done for 540 problem instances, and the results of experiments related to the suggested heuristic are shown. These experiments compare the heuristic proposed by Sansarcı (2007) and our work; moreover the effects of different factors on our heuristic are also given. Finally, in Chapter 7, with discussing results of the suggested heuristic, a conclusion is made.

2. LITERATURE REVIEW

Our problem area is unrelated parallel machine environment. Due to the different attributes of the machines, the processing time for a job depends on the machine. There exist many articles concerning unrelated parallel machines. One of these articles is written by Liaw et al. (2003) where the objective is to minimize the total weighted tardiness. They proposed a branch and bound algorithm for that problem and solved an assignment problem for finding lower bound. Also they generated a heuristic for finding upper bound for the branch and bound algorithm.

Ghirardi and Potts (2005) also studied unrelated parallel machines with the objective of makespan minimization. Recovering Beam Search method which is a Beam Search that allows recovering of previous steps when needed is used in the article. The authors stated that their algorithm performs well on large instances. Sung and Vlach (2005) presented an algorithm for minimizing weighted number of jobs that are completed exactly at their due dates since being not only late but also early usually means penalties. Authors considered the problems with the same objective for single machine and identical parallel machines and demonstrated that these are polynomial time solvable. Moreover the same problem with fixed number of unrelated parallel machines case is also studied and a polynomial time algorithm is presented and it was shown that the problem becomes NP-hard in the strong sense when the number of machines is not fixed. By comparing unrelated paralllel machine case with other cases of the problem, the complexity of unrelated parallel machine case has been demonstrated.

These articles are related to unrelated parallel machine problem without setup considerations. But as we have setup in our case, we have to consider the unrelated parallel machine articles regarding setup. Weng et al. (2001) constructed seven heuristic algorithms for sequence dependent setup case to minimize a weighted mean completion time. They either assigned a job to the machine with the least cost contribution or to the machine on which the job has the shortest processing time. They also tried a strategy where they assigned first the job with the smallest ratio of processing time plus setup time to weight. This strategy outperformed the rest significantly. They programmed these algorithms in C++ and concluded that algorithms are extremely fast and end up with solutions for up to 120 jobs and 12 machines.

Bank and Werner (2001) compared constructive and iterative heuristic algorithms for a solution in which the sum of weighted linear earliness and tardiness penalties is minimal. The constructive heuristics are composed of two stages which are assignment of jobs to machines and after assignment determining schedule of jobs for each machine in order to minimize objective function value.

Balasubramanian et al. (2004) proposed two genetic algorithms with three phases for minimizing total weighted tardiness in on parallel batch machines with incompatible job families. Differences between the two genetic algorithms come from batching decision making order. In both versions batching is done with heuristic algorithms. The first version makes batching decision on the first phase and then assigns those batches to machines and the second version assigns the jobs to machines and then makes batches on each machine. It is also concluded that the first version of genetic algorithm is better than the second version in solution quality and computation time.

Chena and Wu (2006) studied on unrelated parallel machines with setup considerations in order to minimize total tardiness. They presented a heuristic which is a combination of threshold-accepting methods, tabu lists and improvement procedures. After the heuristic is compared with simulated annealing method and optimal solution, it is seen that proposed heuristic results with optimal solutions for problems in small sizes and outperforms simulated annealing method for problems in larger sizes.

Rabadi et al. (2006) generated new meta-heuristic called Meta-RaPS and compared it with an existing heuristic in the literature called Partitioning Heuristic. From the comparison it was stated that their meta-heuristic outperforms the existing one for large scaled problems and end up with optimal solutions in small sized problems.

The previously mentioned articles were concerning sequence dependent setups on unrelated parallel machines. There are fewer articles about family dependent setups in the unrelated parallel machine literature. One of them is the article of Brucker et al. (1998) in which the authors worked on groups of jobs and made the batching decision of them. A group can be split into batches but there is no permission for interrupting a batch being processed if the process has started. There is a setup time if batches from different groups are being processed on a machine concurrently. This setup time is dependent on the group of batch that is going to be processed. The objective in the article is to come up with a solution according to which all the groups are being processed before their due dates. The completion time of a group is the completion time of the last processed job in that group. The problem is composed of three sub problems which are determination of batch sizes, determination of machines on which each batch will be processed and the order of batches that are assigned for each machine. It was stated in the article that the problem is NP-hard even for the case of two identical machines, unit processing times, unit set-up times and a common deadline. It is strongly NP-hard if machines are uniform, the number of jobs in each group is and processing times, set-up times and deadlines are unit. A family of approximation algorithms has been constructed in the article. Chen (2006) also studied family dependent setup structure on unrelated parallel machines. Their aim is to minimize maximum tardiness. They considered the production environment is in die casting departments and stated that a setup for dies is incurred if the type of the job scheduled is different from the previous one on that machine. A heuristic which is based on guided search, record-to-record travel, and tabu lists is proposed for the problem. The proposed heuristic is tested in terms of computational time and quality of the solution and is compared with optimal solutions and a simulated annealing method. As a result of these comparisons, it was seen that the heuristic outperforms simulated annealing method and also can end up with optimal solutions in small scaled problems.

Although there are few articles in the literature related to family dependent setups in unrelated parallel machines; we can find more articles about family dependent setup issue in identical parallel machine environment or single machine environment. In order to understand family dependent setup subject, we have to mention these articles also. One of the authors interested in single machine problems with family dependent setup concept is Chen (1997). In his article, there exist batches of jobs and a setup time is incurred when job from one batch is processed after a job from another batch. Two problems are studied in the article one is minimization of the total earliness and tardiness penalties provided that each due date of batches is externally given and the other one is minimization of the total earliness and tardiness penalties plus the total due date penalty where each due date is a decision variable. It was shown that the first problem is NP-hard and for the second problem a polynomial dynamic programming is proposed with two batches of jobs.

Webster (1997) analyzed scheduling of job families in unrelated parallel machine case in order to minimize weighted deviation about a common due date where a setup is done between processing of jobs from different families. It is shown that the total earliness/tardiness problem is NP-hard when the number of machines and families are arbitrary.

Chen and Powell (2003) studied family dependent setup in identical parallel machines where jobs to be processed can be divided into different families such that a setup is required whenever there is a switch from processing a job of one family to another job of a different family. The authors considered two problem instances one is to minimize total weighted completion and the other one is to minimize weighted number of tardy jobs. Column generation based branch and bound algorithm is proposed for these problems and it was seen that proposed algorithms are good enough to solve medium sized problems optimally.

Another study for parallel machine scheduling with family dependent setup consideration is done by Dunstalla and Wirth (2005). They generated heuristics for minimization of the total weighted completion time and the performance of the generated heuristics are calculated by making comparison between heuristic solutions and lower bounds and solutions obtained using an exact algorithm.

Omar and Teo (2006) also worked on this subject and they presented a mixed integer programming formulation model in order to minimize the sum of earliness and tardiness. It was stated in the article that their mixed integer programming formulation model can provide optimal solutions for up to 18 jobs with up to four job families.

Up to now, we have considered setup related portion of our problem which was unrelated parallel machine case with family setup, job-splitting and machine eligibility constraints. So we need to consider job-splitting issue and articles related to job-splitting. The article by Santilan (2002) is one of the articles on job-splitting property. In this article, jobs can be split into lots and the objective in the article is to minimize total tardiness. The problem is presented as a mixed integer programming problem. Decisions for lot sizing, assigning of these lots to machines and appropriate sequence should be made and since this problem is NP-hard, a local search heuristic based on simulated annealing is proposed. Heuristic methods are used for initial feasible solution and neighborhood solution generation. It was shown that the proposed approach yields optimal solutions in small sized problems, near optimal solutions in medium sized problems and good solutions in large sized problems.

Yalaoui and Chu (2002) also considered job splitting by considering the problem of minimizing total tardiness in a identical parallel machine environment. A branch and bound algorithm is given in the article which considers dominance properties, lower and upper bounding schemes developed by the authors.

Kim et al. (2004) studied on a total tardiness minimization problem where a job can be split into a discrete number of sub jobs that can be processed independently on parallel machines, and also simultaneously on different machines. A two-phase heuristic algorithm is proposed in the article where an initial sequence is constructed in the first phase and splitting each job into sub jobs and rescheduling jobs and sub jobs on the machines is executed in the second phase.

Shim and Kim (2006) concerned the objective of minimizing total tardiness with job splitting property on identical parallel machines. A branch and bound algorithm considering dominance properties and lower bounds is suggested and the algorithm is shown to be good to solve problems of moderate sizes in a reasonable amount of computation time.

Job splitting concept is also studied by Tahar et al. (2006). The problem is scheduling a set of independent jobs in order to make span minimization on a set of identical parallel machines and a heuristic algorithm is developed, which is a heuristic based on linear programming formulation is developed and it was seen that the method is very practical in real-life problems.

The articles that we have considered related to job splitting property are identical parallel machine environment cases, but since our problem is, an unrelated parallel machine related; we have to mention job splitting articles in unrelated parallel machines which are fewer than the identical parallel machine cases. Logendran and Subur (2004) studied minimizing total weighted tardiness on job splitting on unrelated parallel machines and a tabu search based six different heuristic solution is proposed for the problem which uses four different methods based on dispatching rules for generating an initial solution. It is assumed in the article that a job can only be split into two portions since large number of lot splits may result in higher work in process inventory due to the reason that lots which are completed earlier have to wait for the other split parts of the job. The six proposed heuristics were tested on small problems, compared with optimum solutions and seen that good solutions can be obtained by these heuristics. Also the heuristics used for generating initial solution are seen to be capable of obtaining initial solutions that significantly accelerate the tabu-search-based heuristics to attain the best solution. The use of long-term memory in tabu search based heuristics is significant since it helps to obtain a good solution and a variable tabu list size is preferred for solving small sized problems and a fixed tabu list size is preferred for solving medium and large sized problems.

Machine eligibility is another issue that we consider in our problem. Machine eligibility means that not all machines are capable of processing all jobs, so jobs can only be processed on their eligible machines. Centeno and Armacost (1997) present an algorithm for the problem of minimizing maximum lateness and apply the algorithm for semiconductor manufacturing firm which is an environment with identical parallel machines. Also they evaluated their work using real data from an operational environment of a semiconductor manufacturing firm and after comparing it with the actual scheduling system being used by the organization it was seen that a significant performance improvement is provided using the proposed scheduling algorithm.

Bekkia and Azizoglu (2007) proposed a branch and bound algorithm that employs dominance conditions and tight bounds for maximizing the total weight of the jobs processed and coded the branch and bound algorithm in Turbo Pascal. The computational results revealed that the bounding procedures are quite powerful and the branch and bound algorithm ends up with optimal solutions in reasonable time. Sheen et al. (2006) worked on minimization of the maximum lateness and generated a branch and bound algorithm for searching for the optimal solution of the problem which uses several immediate selection rules for solving this scheduling problem. They evaluated the performance of the branch and bound algorithm and experimental results showed that the proposed branch and bound algorithm can solve instances optimally in a reasonable time.

Liao and Sheen (2007) propose a polynomial time binary search algorithm for the problem of minimizing the make span with machine eligibility constraint. The authors aim to either verify the infeasibility of the problem or solve it optimally if a feasible schedule exists. The proposed algorithm first verifies the infeasibility of the problem and if there is no feasible schedule then the algorithm is terminated; otherwise the optimal value can be obtained by performing the binary search.

Machine eligibility in unrelated parallel machines is studied by Salem (1999) and developed four heuristic algorithms for finding efficient and quick solutions for minimization of make span. Also the performances of heuristics were evaluated by making comparisons of the make span values found by these heuristics with the optimal make span value and it was seen that the performance of heuristics were satisfactory. Another problem of unrelated parallel machine with machine eligibility is studied by Senniappan

(2001) in order to minimize the sum of completion time on all machines. He proposed a mathematical programming model for the problem and since the problem is complex, heuristics and a genetic algorithm were developed to generate quick and effective solutions. Also heuristics are used for the genetic algorithm to generate initial set of solutions in order to reduce computational time. The proposed solutions are evaluated by using them for an aluminum processing plant in Turkey. After the evaluation it was seen that the proposed methodology outperformed the company's existing procedure.

Rojanasoonthon (2004) worked on the same problem considering time windows. It was proven that the problem is NP-hard and mixed-integer linear programming formulations are presented in the article. Since the problem is difficult to solve, the author developed a dynamic programming-like heuristic and a greedy randomized adaptive search procedure. Also an exact method was also developed and a branch-and-price method is applied where the initial solution is provided by the greedy randomized adaptive search procedure. It was shown that the proposed procedure was found to be very effective, providing the true optimum for instances with up to 100 jobs and 2 machines and it is able to solve many instances that were believed to be beyond the capabilities of exact methods.

Logendrana et al. (2007) studied minimizing the weighted tardiness of jobs in unrelated parallel machines considering machine eligibility restrictions. For the problem six different search algorithms based on tabu search for and four different initial solution finding mechanisms, based on dispatching rules are developed. Four different initial solution finding mechanisms are important since better quality initial solutions might lead to identifying better quality final solutions. After computational experiments and statistical analysis performed, the search algorithm with short-term memory and fixed tabu list size is seen to better in solving small size problems, while the one with long-term memory and variable tabu list size is seen preferable for solving medium and large size problems.

Sansarcı (2007) studied the problem which is closely related to our problem since he worked on unrelated parallel machine environment with machine eligibility constraints, job-splitting property and family setup structure. The objective is to minimize total tardiness. In the study, a four phased heuristic algorithm using an aggregate planning approach is proposed. Aggregate planning model in order to determine the batch sizes, batch sequencing, and alternative machine selection is done using two control parameters which are related to aggregation. The two control parameters which are production-per-setup and aperture are used as inputs in this aggregation phase. For solving the aggregate planning model, a linear programming formulation is applied. After solving aggregate planning, the result of aggregate planning model is used in reducing the problem into several single machine total tardiness problems with a heuristic algorithm. A search algorithm is used for tuning the control parameters. In the study it was stated that in order to implement the proposed heuristic and investigate its performance, a problem set is generated. Test demonstrated that the heuristic performs best.

The most related problems to our problem are studied by Shim and Kim (2006), Brucker et al. (1998), Chen (2006), Logendran and Subur (2004) , Logendrana et al. (2007) and Sansarcı (2007). However, our problem differs from the problem that Shim and Kim (2006) studied since they did not consider machine eligibility and their problem is in identical parallel machine environment. Also the problem of Brucker et al. (1998) is different from our problem in ways that there is no splitting of jobs and machine eligibility in their problem although the objective of their problem is minimization of maximum lateness. Chen (2006) studied minimization of maximum tardiness in unrelated parallel machines; but they also did not consider job splitting and machine eligibility. Logendran and Subur (2004) studied a different problem from ours since our problem and their problem involves only job splitting property in common. Logendrana et al. (2007) studied a distinct problem since they studied only machine eligibility restriction but did not work on other features of our problem. The only study which considers the same problem is the one of Sansarcı (2007). Other problems in the literature differ from our problem and the difference is mainly about the setup structure, eligibility constraints, and job-splitting property.

3. PROBLEM DEFINITION

In this study, we worked on parallel machine scheduling in which there are n jobs, m machines and F product families. Each specific customer order is a job and machines are assumed to be unrelated parallel machines. In general parallel machines may be either identical or non-identical. Identical parallel machines have the same technological properties, so speed of production is the same for each identical parallel machine. But non-identical machines have different technological properties which lead to different production speeds. Unrelated parallel machines environment is a special case of the non-identical machines which have different production speeds for each product family.

Product families are formed considering the production needs of jobs for producing products and jobs with similar requirements are combined to form a family. Family concept is very significant since there is no need for setup between processing of jobs belonging to the same family but a family dependent setup is incurred between jobs of different families. So in order to minimize setup times, it is better to sequence jobs of the same family consecutively in many cases. When we plan to process a job from a family different than the current job, then some regulations and preparations are needed to be done on the machine.

Since unrelated parallel machines have different technological properties, all machines are not capable of producing all jobs and so there is a set of eligible machines for each job. These jobs can only be processed on machines in their eligibility set of machines. This situation turns to be the machine eligibility constraint for our problem.

Moreover, we do not have to finish processing of a job once we start it; we can split the job and finish some portion of it and then finish other portion in another machine and in another time. This situation does not constrain our problem but gives us a small freedom while assigning jobs to machines.

The objective of our problem is to minimize total tardiness since producing goods on time is one of the crucial aspects of customer satisfaction. Tardiness of a job is found from the difference between the delivery time and the due date of that job. If this difference is bigger than zero, than tardiness is that difference. Otherwise, tardiness is zero. So if the job is delivered before its due date, then its tardiness value is zero, otherwise the value is the difference between the delivery time and the due date.

3.1. Formal Problem Definition

In our problem for the minimization of total tardiness, we have n jobs and m unrelated parallel machines and F product families. For each job j where j is the job index we have due date information: d_j. Since there is a need for setup between processing of different families, we need to consider the case where no setup time is needed. So we call production of a family without interruption on a machine as a "batch" and form variables considering batch concept. Other related variables are:

b_{jf} is a binary parameter which shows that job j belongs to family f;
Q_f is the set of jobs j where $b_{jf} = 1$;
p_{fr} is the unit production time of a job of family f on machine r;
S_{fr} is the setup time required whenever a family f is produced after any other family or at the first place on machine r;
x^E_{fr} is the eligibility constraint which is a binary parameter that equals 1 if family f can be produced on machine r and zero otherwise;
X^B_{rb} represents the quantity of batch b on machine r;
x^{BF}_{rbf} is a binary variable showing if the batch b on machine r is of family f;

There exist other aspects of our problem that we need to mention which is one machine can process one job at a time and we know the machines which are eligible to produce jobs belonging to a certain family. Also setup times are constant and known.

The objective of our problem is total tardiness. A tardy job is the one which have a completion time later than its due date and tardiness of a tardy job is maximum of zero value and the difference between its completion time and due date. Total tardiness value is the sum of tardiness values of all jobs.

3.2. Mathematical Model

In this section of this study, we present a mathematical model for our problem. The problem can be formulated as a MIP. This mathematical model demonstrates the formal definition of the problem.

$$\min \sum_{j} T_j$$

s.t.

$$\sum_{r} y_{jr} = 1, \qquad \forall j \tag{3.1}$$

$$x_{jr} - y_{jr} \geq 0, \qquad \forall j, r \tag{3.2}$$

$$X_j^E r - x_j r \geq 0, \qquad \forall j, r \tag{3.3}$$

$$\sum_{j} x_{0jr} = 1, \qquad \forall r \tag{3.4}$$

$$\sum_{j} x_{kjr}, \leq 1 \qquad \forall k, r \tag{3.5}$$

$$\sum_{j \neq k} x_{kjr} + x_{0jr} - x_{jr} = 0, \qquad \forall j, r \tag{3.6}$$

$$T_j - C_j + d_j \geq 0, \qquad \forall j \tag{3.7}$$

$$Cjr - pjr * yjr - \sum_{j \neq k} skjr * xkjr - sjr * x0jr - \sum_{j \neq k} C_{kr} * x_{kjr} = 0, \qquad \forall j \tag{3.8}$$

$$Cj - Cjr \geq 0, \qquad \forall j, r \tag{3.9}$$

$$T_j \geq 0, y_{jr} \geq 0, x_{jr} = (0,1), x_{kjr} = (0,1), x_{0jr} = (0,1), C_j \geq 0, C_{jr} \geq 0 \tag{3.10}$$

where;

y_{jr} : Proportion of job j which is produced on machine r;

x_{jr} : 1 if job j is processed on machine r and 0 otherwise;

x_{0jr} : 1 if job j placed at the first place on machine r and 0 otherwise;

x_{kjr} : 1 if job j comes just right after job k on machine r and 0 otherwise;

T_j : Tardiness of job j;

C_j : Completion time of job j;

C_{jr} : Completion time of job j on machine r;

d_j : A constant which represents the due date of job j;

p_{jr} : A constant which represents the processing time of job j on machine r;

s_{jr} : A constant which represents the setup time for job j on machine r;

s_{kjr} : A constant which represents the setup time for job j on machine r if job k and job j belong to different families and is equal to 0 otherwise;

The explanations of the constraints are as follows:

1. Equation (3.1): For each job, proportions of that job which are produced on the machines must add up to 1.
2. Equation (3.2): If a job is not produced on a machine, than the corresponding proportion for that job is 0.
3. Equation (3.3): A job can be produced on a machine only if the machine is eligible for that job.
4. Equation (3.4): There can be only one job to be placed first for each machine.
5. Equation (3.5): Each job can follow only one job on each machine.
6. Equation (3.6): If a job is produced on a machine, it must either be the first job, or follow any other job on that machine.
7. Equation (3.7): Tardiness of a job is greater than or equal to completion time of that job minus its due date. They are not necessarily equal since tardiness cannot be negative.
8. Equation (3.8): Completion time of a partial production of a job on a machine is the sum of its processing time, setup time, and the completion time of the previous partial production of a job on the same machine.
9. Equation (3.9): Completion time of a job is greater than all of the completion times of the partial production of that job on the machines.
10. Equation (3.10): x_{jr}, x_{0jr} and x_{kjr} are binary integer variables while other variables are greater than or equal to 0.

4. PROPOSED HEURISTIC

In this study, we suggest to handle the problem with simulated annealing approach. While solving problem in neighborhoods that are found by simulated annealing procedure, we apply a three phased heuristic.

The first phase of the suggested heuristic tries to aggregate jobs which are included in the same family. Aggregated jobs form a job batch. So with that aggregation, problem reduces from scheduling each job to scheduling these job batches. Aggregation is made with two control parameters.

By using the job batches generated, a new time structure is constructed which is consisting of different types of time buckets that will be used in later phases. After aggregation of jobs of a family, production quantities in time buckets generated are found by solving an LP. As a result of solving the LP we make our procedure to use capacity effectively. The results of phase 2 will be used as inputs of the latter phase.

Production quantities of second phase are formed as batches in third phase. Those batches have machine information by which they are produced, the family information that they produce, start and finish times.

After formation of batches, we know in which machine, between which times a particular family is produced. Then jobs of a family are assigned to those batches producing the family. Jobs can be splitted into more than one batch. After the assignment of jobs of a family to the batches, tardiness of a job can be calculated. So; total tardiness value can be found.

4.1. First Phase

At the first phase of the suggested heuristic, jobs of a family are combined into job batch. In performing these job batches, the aim is to make enough quantity of production by combining jobs while making sure that jobs that are aggregated have closer due dates. So there are two control parameters which are production_per_setup and maximum_difference.

Production_per_setup tries to combine enough quantity of jobs so that the setup done for these combined jobs worth to produce. For example an instance in which we spent more time for setup than production is not a logical case. So there should be a ratio between time spent for production and time spent for setup which is production_per_setup. When production_per_setup value is large, it means that setup time is very crucial and we should produce larger quantities for that setup.

When trying to combine larger quantities of jobs, we may also get into the mistake of combining jobs which have due dates very far a way from each other which is not logical too. For not getting into that mistake, maximum_difference parameter controls the process. Maximum_difference parameter is the maximum difference value that can be between the due dates of jobs included in a job batch. When maximum_difference value is small, it means that we can only combine jobs with very close due dates. But when that control parameter is large, we can combine jobs that have different due dates, so we can produce a job that has later due date earlier than its due date since it is combined with a job that has very earlier due date.

For explaining the meaning of control parameter better: assume that a job batch consists of four jobs all of which have a production time of 10 with due dates 1, 2, 4 and 5. Also assume that there is a setup time of 2 before the production of this job batch. Since production per setup value is the ratio between production time and setup time, it is found as 20 in this example since production time is 40 due to $(10 + 10 + 10 + 10)$ and setup time is 2. The maximum difference between due dates of jobs is 4 due to difference between due dates of jobs four and one. The decision to combine these jobs in a job batch is done with the two control parameters, if the production per setup and maximum difference of the job batch is not disturbing the control parameters boundary, then these jobs can be formed into a job batch; otherwise they should be combined into more than one batch.

For making a reasonable amount of production after a setup, production per setup value is desired to be at high levels and for not combining jobs with very distinct due dates, maximum difference value is desired to be at lower levels.

In the first phase, an algorithm is run for gathering the jobs of a family into a batch according to two control parameters which are production_per_setup and maximum_difference. For making the logic of the algorithm clearer, we will define a notation.

4.1.1. Notation

The notation to be used in the algorithm is as follows:

MD_f : Maximum difference control parameter value for family f

PPS_f : Production per setup control parameter value for family f

d_j : Due date of job j

D_j : Demand of job j

p_{fr} : Production time for producing one unit of family f on machine r

S_{fr} : Setup time for producing one unit of family f on machine$r$

b_{jf} : If job j is included in family f, it is equal to 1; otherwise it is 0

d_{bf} : Due date of job batch b included in family f

d_{bff} : Due date of the first job in job batch b of family f

d_{bfl}: Due date of the last job in job batch b of family f

Q_{bf}: Total quantity of jobs included in job batch b of family f

A_f: set of machines that family f can be produced

4.1.2. Inferences

The inferences we can get at the first phase are as follows:

Time spent for production of job batch b of family f in machine r: $[Q_{bf} * p_{fr}]$;

Production per setup value of job batch b of family f in machine r: $[Q_{bf} * p_{fr}/S_{fr}]$;

Maximum due date difference between jobs in job batch b: $d_{bfl} - d_{bff}$

In phase 1, production_per_setup value of a job batch b is not wanted to be less than PPS_f of the related family. So jobs should be combined enough to satisfy this boundary. At the maximum_difference point of view, maximum difference between job due dates of a job batch is wanted to be smaller than MD_f of the family. So we can't combine jobs with due dates that have difference more than maximum_difference value of the family.

4.1.3. Algorithm

Step 0: Sort families in ascending order with respect to index. Start with the first non-processed family and go to Step 1.

Step 1: Sort jobs in family in ascending order with their due dates.

Step 2: Create a new job batch. Add job to this job batch.

Step 3: Calculate due date of job batch as follows:

For all jobs in the batch calculate:
{due date of the job + processing time of succeeding jobs of this job in the batch}
The maximum value among all jobs is the due date of the job batch.

Step 4: If all jobs are processed, go to step 5.

Iterate succeeding job in the sequence.

Let i show the job sequence.

Look if control parameters are satisfied.

For maximum_difference: $[d_i - d_{bff}] < MD_f$ should be satisfied.

For production_per_setup: $\sum_{r \in A_f} [Q_{bf} * p_{fr}] / \sum_{r \in A_f} S_{fr} < PPS_f$ should be satisfied.

If parameters are satisfied add job to the existing job batch. Go to step 3.

If maximum_difference of job batch is greater than or equal to MD_f or if production_per_setup is greater than or equal to PPS_f , go to Step 2.

Step 5: If all families are processed, stop. Otherwise, pick the succeeding family and go to step 1.

4.2. Second Phase

In the second phase of the suggested heuristic, outputs of first phase will be used as input and by using job batches of families generated in the first phase, new time structure will be constructed in this phase. An LP model is solved in second phase in order to find effective production values in constructed time buckets.

After creating job batches of all families in first phase, phase 2 uses these job batches for creating time buckets. Time bucket structure generated in phase 2 consists of family time buckets and resource time buckets.

Family time buckets are generated for each family, so each family has different family time buckets according to due dates of their job batches. For constructing family time buckets of a family, sequence job batches of that family in ascending order in their due dates. A family time bucket is the time between consecutive due date values.

For explaining meaning of family time bucket better, assume that there is a family with job batches which have due dates 3, 5, 7 and 9. So; first family time bucket for that family will be time between 0 and 3. Second family time bucket is between time 3 and time 5. Third family time bucket is time between time 5 and 7. Last family time bucket is between time 7 and time 9.

In order to construct resource time buckets, all job batches of all families are combined and then arranged in ascending order according to their due dates. Resource bucket is the time between consecutive due date values. Resource buckets are the same in all machines.

Assume that we have two families. First family is a family with job batches which have due dates 3, 5, 7 and 9. Second family have job batches with due dates 2, 5 and 8. When we combine all job batch due dates; we have 2, 3, 5, 7, 8 and 9. So resource time buckets for all families are: first resource time bucket between time 0 and 2, second resource time bucket between time 2 and 3, third resource time bucket between 3 and 5, fourth resource time bucket is between 5 and 7, fifth resource time bucket between 7 and 8, last resource time bucket is between 8 and 9.

4.2.1. Notation

The notation to be used in the algorithm is as follows:

T_{ft_f}: Tardiness of family f in family time bucket t_f which is the quantity of unsatisfied demand

S_{frt}: Setup time for producing family f at machine r in resource time bucket t

X_{frt}: Total quantity of production of family f, at machine r, in resource time bucket t

x_{frt}: Equals to 1 if family f can be produced on machine r in resource time bucket t

p_{fr}: Unit production time of family f at machine r

S_{fr}: Setup time needed for family f on machine r

C_t: Length of resource time bucket t

C_{t_f} : Length of the resource time bucket succeeding family time bucket t_f

D_{ft}: Total demand of family f to be satisfied at the end of resource time bucket t. Demand quantity is the demand of job batch of family f which has due date equal to finish time of resource time bucket t

LP model for phase 2 is presented with its objective function and constraints.

4.2.2. Objective Function

Objective function of the LP model is minimizing total tardiness value. Sum of tardiness (quantity of unsatisfied demand) of family f on family time bucket t_f multiplied by length of the resource time bucket succeeding family time bucket t_f is calculated.

$$\sum_{f}\sum_{t_f} T_{ft_f} * C_{t_f} \tag{4.1}$$

4.2.3. Constraints

The LP model has three constraints. First constraint tries to make sure that total time spend for setup and production inside a resource time bucket is less than or equal to total time of resource time bucket itself for all resource time buckets and for all machines.

$$\sum_{f} X_{frt} * p_{fr} + \sum_{f} S_{frt} - C_t \leq 0 \quad \forall t, \forall r \tag{4.2}$$

Second constraint is for guaranteeing that total setup time in the resource time buckets inside a family time bucket on a machine is equal to required setup for the corresponding family and machine.

$$\sum_{t \in tf} S_{frt} - S_{fr} = 0 \quad \forall f, \forall r \tag{4.3}$$

Third constraint is for quantity of unsatisfied demand of a family in a family time bucket, which is represented as tardiness, is greater than or equal to total demand up to the family time bucket and cumulative production of that family due to the family time bucket.

$$\sum_{t \leq tf}\sum_{r} X_{frt} * x_{frt} - \sum_{t \leq tf} D_{ft} + T_{ftf} \geq 0 \tag{4.4}$$

All of decision variables should be greater than or equal to zero.

$$T_{ftf} \geq 0, S_{frt} \geq 0, X_{frt} \geq 0 \ \forall f, \forall r, \forall t, \forall tf \tag{4.5}$$

4.3. Third Phase

After solving LP in the second phase, we know the production quantities in resource time buckets. But we have to assign jobs to those productions in order to generate a schedule showing in which machine each job is produced and between which times.

In order to generate a schedule, production quantities of a family in different resource time buckets are aggregated. This aggregation is done as follows: for resource time buckets in a machine belonging to the same family time bucket are combined and their total production quantity form a batch. The reason for that aggregation is that: the production quantities of resource time buckets in a machine belonging to the same family time bucket are partial productions of the same aggregate job.

Also there may be cases in which existing production quantities can't satisfy the demand. So; for those cases the unsatisfied proportion of demand is also made a batch and is assigned to the machine where it can be finished first.

After combining production quantities of resource time buckets in a machine belonging to the same family time bucket, and also generating batches for unsatisfied demand, we find start times and finish times of batches in order to represent those batches as schedule.

At the end of aggregation and finding start and finish times of job batches, a schedule is at hand with job batches in which we can know the quantity, the producing family, time information and the machine information.

Since we do not know exact finish times of jobs with the schedule at hand, and so real tardiness values; we have to assign the jobs to those job batches. While assigning jobs to job batches we can split jobs to more than one batch since we have the job splitting property. After assignment of jobs to batches, we know real completion time of all jobs. So with the real completion time of jobs, we calculate real tardiness values of all jobs.

4.3.1. Algorithm

Step 0: Sequence all family time buckets in ascending order with their due dates. Start with the first non-processed family time bucket and go to Step 1.

Step 1: Combine production quantities of resource time buckets belonging to a machine that is between start and finish time of the family time bucket. Create a job batch for these combined resource time buckets.

Step 2: Find start and finish times for that created batch with considering family setup structure.

Start time of the job batch: If there does not exist job batches in the machine of the job batch, then start time of the job batch is the start time of the resource time bucket that has the minimum start time among resource time buckets that have constructed the job batch.

Assume that three resource time buckets have constructed the job batch, which are the resource time buckets between times 2-5, 3-4 and 1-7. If there is not an existing job batch on the machine of these resource time buckets, then start time of job batch would be time 1 since the minimum start time is time 1.

If there exist job batches in the machine of the job batch, the last finishing one among those job batches is the preceding batch of the job batch. Then start time of the job batch is either start time of the resource time bucket that has the minimum start time among resource time buckets that have constructed the job batch or finish time of the preceding job batch. The selection is the one which has maximum value. Also if the preceding batch is of another family, then we have to add the corresponding family dependent setup value before starting the job batch. Finish time of the job batch: Start time of the job batch + (total quantity of production) * (unit production time of family f on machine r)

Step 3: If all family time buckets are being processed, then go to Step 4, otherwise move to to Step 1.

Step 4: All the resource buckets belonging to a family time bucket on a machine are combined into a batch.

We have the batch information processing a particular family. With this information we look whether the total demand of jobs belonging to each family is satisfied with the current production or not.

If the demand is not satisfied, the unsatisfied portion is formed into a batch. Assignment of this batch to a machine is done as follows: For all machines compute the following: [(Finish time of last batch scheduled at machine) + (Quantity of unsatisfied portion) * (Processing time of the family on machine)]

The machine with the minimum value is the machine to which the batch of unsatisfied portion will be assigned.

Step 5: For finding tardiness values of jobs, assign them to batches.

This assignment is done as follows:

For each family, sequence all jobs in ascending order according to their due dates. Also sequence the batches of the family in ascending order according to their start times.

Assign the first non processed job in the sequence to the first job batch in the sequence that has enough empty capacity. Remember that a job can be assigned to more than one batch since it can be splitted. Go to Step 6.

Step 6: If all jobs are assigned to a job batch, then go to step 7; otherwise make a move to step 5.

Step 7: After generating a schedule with job batches, family dependent setup times are added in this step. A rework is done on start and end times of job batches and starting from the first job batch on all machines, if there is a family dependent setup between a job batch and its preceding job batch, then this setup time is added into the schedule and job batches are shifted accordingly.

After completing third phase, we have a complete schedule in which we can get information about which job is processed on which machine(s), and between which times. So we get the tardiness information of each job.

When calculating tardiness values of jobs, we look at the batches that the job is assigned to. If a job is assigned to one batch which has a quantity equal to quantity of the job, then completion time of the job is the completion time of batch. If job is assigned to more than one batch, then completion time is calculated according to last batch that it is assigned. But we have to think that the batch a job has been assigned can have more than one jobs so when calculating completion time of a job, we have to consider other jobs assigned to the same batch.

In an instance that a batch is producing more than one job, the completion time of jobs included are not the same. Completion time of a job is the completion time of the jobs that are assigned to the batch before this job and the job itself.

As example, we have job 1 with quantity of 100, which is assigned to batch 1 with quantity 70 starting at time 2 finishing at time 4, and batch 2 with quantity 150 starting at time 4, finishing at time 7. Also we have job 2 with quantity 40 which is assigned to batch 2 after job 1 and a job 3 with quantity 130 which is assigned to batch 2 and batch 3 with quantity 50 starting at time 8 finishing at time 10.

Finishing time of job 1 is calculated from batch 2 since it is the last batch that job

1 is assigned. Batch 1 produces 70 units of job 1 and 30 units of job 1 is produced by batch 2. At batch 2 a production quantity of 150 is done in 3 unit times, so production time per unit is: 0.02. Since job 1 is produced firstly on batch 2 and 30 units of job 1 is produced, time spent for producing job 1 on batch 2 is 30 * (0.02) = 0.6 times. So since batch 2 starts at time 4, when we add 0.6 production time, we find that completion time of job 1 is time 4.6.

After finding completion time of job 1, we will find completion time of job 2 which is produced by batch 2 after job 1. Job 2 consumes 40 from batch 2. Since 40 units spend 40* (0.02) = 0.8 times to produce, job 2 will be completed at time (4.6) + (0.8) = 5.4.

Job 3 is produced by two batches which are batch 2 and batch 3; 80 units on batch 2 and 50 units on batch 3. Since completion time of batch 3 is the latest one among these and its all production is only for job 3, completion time of job 3 is time 10 which is the completion time of batch 3.

4.4. Illustration of the Proposed Heuristic

In this section, first a simple example will be studied in order to make the proposed solution more clear, and then a more complex example will be studied for better understanding of the proposed heuristic.

Assume we have 3 machines, 5 jobs and 2 families. Quantities, due dates and families of the jobs are presented in Table 4.4. In family 1 we have jobs with quantities 1500, 200 and 150. In family 2, we have jobs with quantities 1100 and 300. Jobs of family 1 can be produced on both of the machines where jobs of family 2 can be produced on only machine 12. Production time is 200 per unit for family 1 on all machines and for family 2 production time is 75 per unit on machine 3.

In phase 1 of the proposed heuristic, according to the control parameters "production_per_setup" and "maximum_difference", jobs of a family is grouped into job

Table 4.1. Job information of example 1

Job	**Quantity**	**Due date**	**Family**
1	1500	6	1
2	150	3	1
3	200	7	1
4	1100	5	2
5	300	10	2

batches. At the end of Phase 1, we have 3 job batches for family 1 and 2 job batches for family 2 which are composed of a single job. Control parameters are assumed production_per_setup as 100 and maximum_difference as 1 day for all families. So production_per_setup should be less than 100 and maximum_difference should be less than one day for aggregating jobs.

By using aggregate jobs formed at the Phase 1, we are going to construct the time structure of the problem given the due dates of the aggregate jobs. Table 4.2 shows the due date structure of the families and the aggregate jobs.

Table 4.2. Aggregate jobs of all families of example 1

Aggregate Job	**Quantity**	**Due date**	**Family**
1	1500	6	1
2	150	3	1
3	200	7	1
4	1100	5	2
5	300	10	2

In beginning of phase 2, family time buckets and resource time buckets are constructed. Family time buckets for family 1 is between time zero (0) and 3 and between 3 and 6 and between 6 and 7. Family time buckets for family 2 is between time zero (0) and 5 and between 5 and 10.

According to family time buckets, resource time buckets are generated for all machines. The resource buckets are: between time zero and 3, between 3 and 5, between 5 and 6, between 6 and 7, between 7 and 10. In Figure 4.3 family time buckets and resource time buckets can be seen for each family and machine.

After constructing family time buckets and resource time buckets, an aggregate plan-

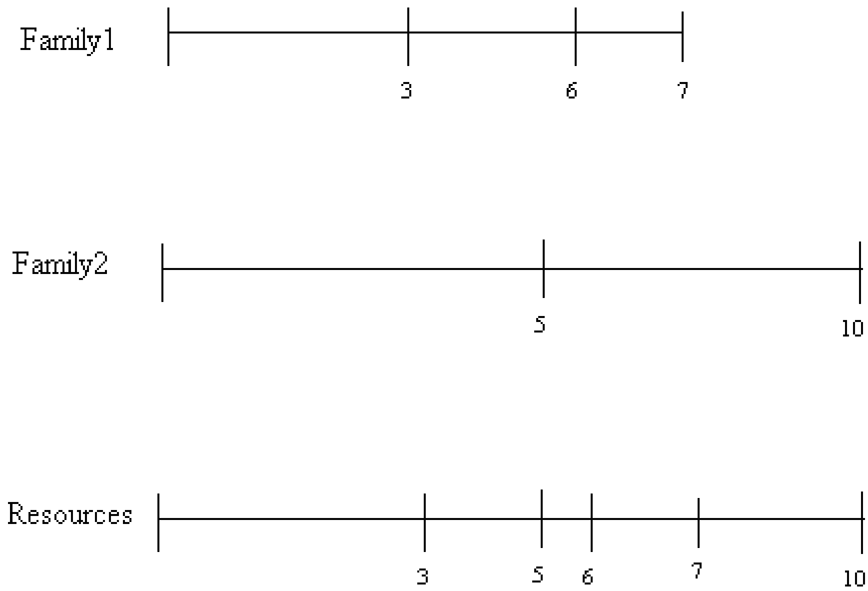

Figure 4.1. Family and resource time buckets of example 1

ning problem structure is constructed and solved in Phase 2. Values for production variables at the end of Phase 2 are presented in Table 4.3.

Resource time buckets are abbreviated as ResTB and ResTB1 is between time zero and 3, ResRB2 is between 3 and 5, ResTB3 is between 5 and 6, ResTB4 is between 6 and 7 and RestTB5 is between 7 and 10. Tardiness values of family time buckets

Table 4.3. Values of the production variables after running LP for example 1

Family	**Machine**	**ResTB1**	**ResTB2**	**ResTB3**	**ResTB4**	**ResTB5**
1	1	864	864	122	0	0
2	3	1400	0	0	0	0

are zero so the optimal objective value is also zero. Using the results of the LP solved in Phase 2, we will generate a schedule in Phase 3 which is represented as a set of batches, including the information of start time, finish time, family and machine.

In Phase 3, batches are generated. In machine 1, batches are 864 and 986 sized batches, in machine 3; a batch with size 1400 exists. These production quantities form a batch and the batches form a schedule with start times, and times, corresponding family and the resource information. After the formation of the batches, they are sequenced in ascending order according to their due dates starting from the earliest due date.

Batch of size 864 on machine 1 starts at time 1 and finishes at time 3 where the other batch of the machine is between time 3 and 5. The batch on machine 3 is between time 1 and 2.

Assignment of these batches to jobs is done like this: we have production of family 1 in machine 1 and we have jobs in family 1 with sizes 150, 1500 and 200 sequenced in ascending order of due dates. Batch of size 864 is producing jobs 150 and 1500. Batch of size 986 is producing jobs 1500 and 200. Job 1500 consume 714 from batch of size 864 and consume 786 from batch of size 986. For family 2, we have production on machine 3 and this batch produces all the jobs belonging to family.

Job with size 1500 finishes at time 4, job with size 1100 is produced at time 1, job with size 200 is produced at time 5, job with size 300 is produced at time 2, and job with size 150 is produced at time 1. According to these finishing times, tardiness of all jobs is zero.

After giving the first simple example, it is better to work on a more complex test example in which we can examine the specifications of our problem. In this second example, we have the machine eligibility feature in which jobs can be produced on some of the machines not on all of the machines.

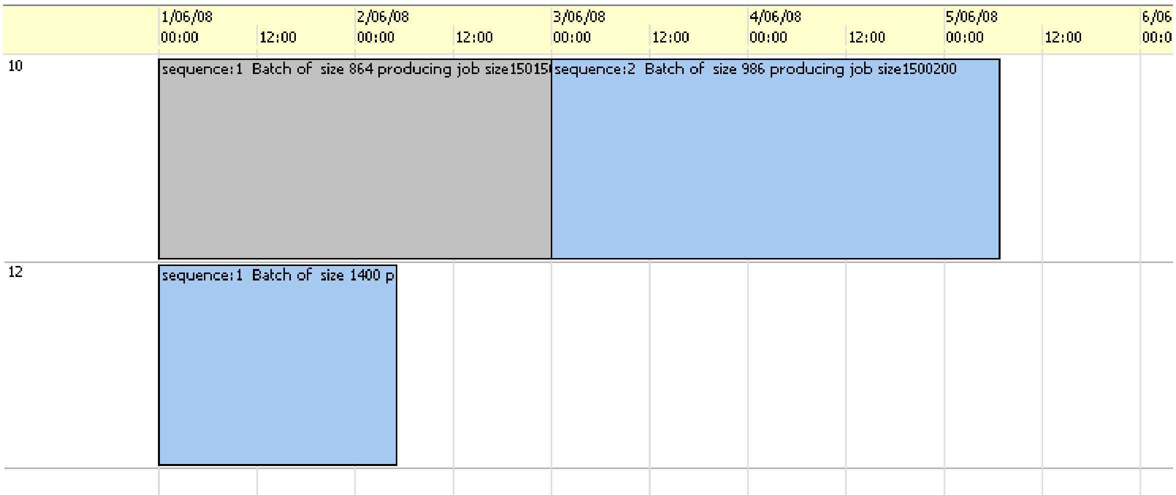

Figure 4.2. Gantt Chart of example 1

Assume we have 4 machines, 15 jobs and 3 families. Quantities, due dates and families of the jobs are presented in Table 4.4. In family 1 we have jobs with quantities 1000, 2000, 500, 900, 400 and 600. In family 2, we have jobs with quantities 1400, 100 and 100. In family 3, we have jobs with quantities 1000, 1500, 300, 700 and 800.Jobs of family 1 can be produced on only machine 2. Jobs of family 2 can be produced on any machine. Jobs of family 3 can be produced on machine 2 and machine 3. Production time is 100 per unit for family 1, 50 per unit for family 2, 150 per unit for family 3.

In phase 1 of the proposed heuristic, according to the control parameters "production_per_setup" and "maximum_difference", jobs of a family is grouped into job batches. At the end of Phase 1, we have 5 job batches for family 1, 3 job batches for family 2 and 5 job batches for family 3 which are composed of a single job. In this second example, we use control parameters different than example 1. Production per set up is 50 and maximum_difference is one day. So production per setup should be less than 50 and maximum difference of due dates of jobs in a job batch should be less than one day.

By using aggregate jobs formed at the Phase 1, we are going to construct the time structure of the problem given the due dates of the aggregate jobs. Family 1 has 6 aggregate jobs, family 2 has 3 aggregate jobs and family 3 has 5 aggregate jobs. Table 4.5 shows the due date structure of the families and the aggregate jobs.

Table 4.4. Job information of example 2

Job	**Quantity**	**Due date**	**Family**
1	1400	2	2
2	100	6	2
3	100	3	2
4	1000	3	1
5	2000	9	1
6	500	10	1
7	900	4	1
8	1000	10	3
9	1500	10	3
10	300	5	3
11	700	6	3
12	800	10	3
13	900	3	2
14	400	9	1
15	600	4	1

In beginning of phase 2, family time buckets and resource time buckets are constructed. Family time buckets for family 1 is between time zero (0) and 3, between 3 and 4, between 4 and 9, between 9 and 10. Family time buckets for family 2 is between time zero (0) and 2, between 2 and 3, between 3 and 6. Family time buckets for family 3 is between time zero (0) and 5, between 5 and 6, between 6 and 10.

According to family time buckets, resource time buckets are generated for all machines. The resource buckets are: between time zero and 2, between 2 and 3 and between 3 and 4 and between 4 and 5 and between 5 and 6 and between 6 and 9 and between 9 and 10. In Figure 4.3 family time buckets and resource time buckets can be seen for each family and resource.

Table 4.5. Aggregate jobs of all families of example 2

Aggregate Job	Quantity	Due date	Family
1	1000	3	1
2	1500	4	1
3	2000	9	1
4	400	9	1
5	500	10	1
1	1400	2	2
2	1000	3	2
3	100	6	2
1	300	5	3
2	700	6	3
3	800	10	3
4	1500	10	3
5	1000	10	3

After constructing family time buckets and resource time buckets, an aggregate planning problem structure is constructed and solved in Phase 2. Values for production variables at the end of Phase 2 are presented in Table 4.6.

Resource time buckets are abbreviated as ResTB and ResTB1 is between time zero and 2, ResTB2 is between 2 and 3, ResTB3 is between 3 and 4 and ResTB4 is between 4 and 5 and ResTB5 is between 5 and 6 and ResTB6 is between 6 and 9 and ResTB7 is between 9 and 10.

Table 4.6. Values of the production variables after running LP for example 2

Family	Machine	ResTB1	ResTB2	ResTB3	ResTB4	ResTB5	ResTB6	ResTB7
1	1	864	244	0	0	0	0	0
1	3	0	528	864	814	864	1222	0
2	3	1727	673	0	100	0	0	0
3	3	0	0	0	543	576	913	576
3	4	576	576	576	576	507	0	0

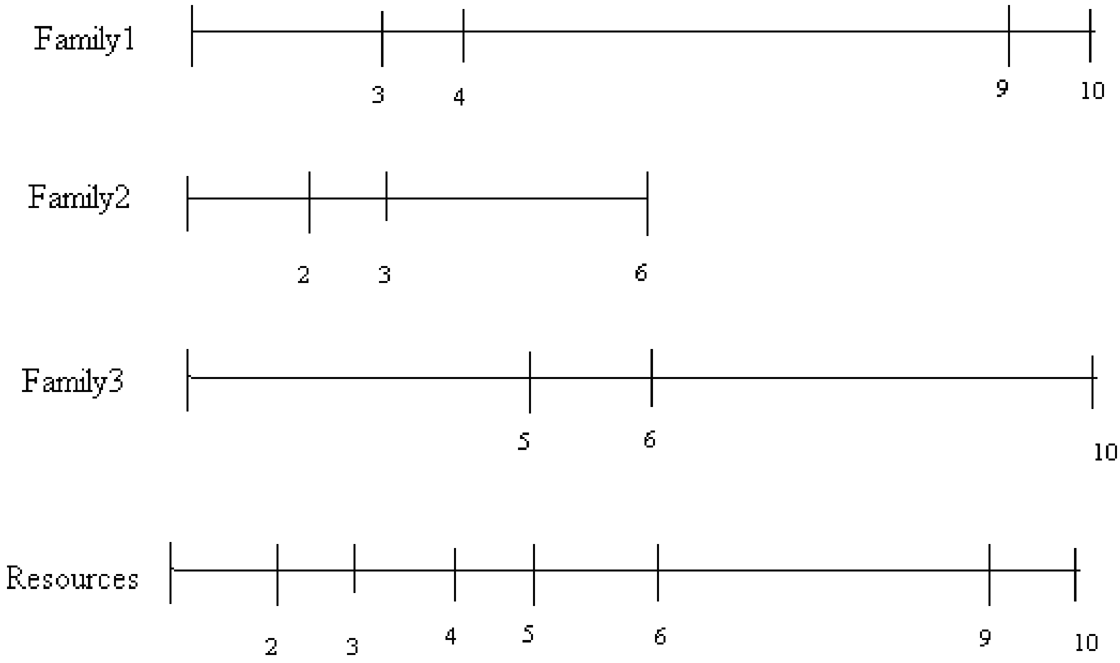

Figure 4.3. Family and resource time buckets of example 2

Tardiness values of family time buckets are zero so the optimal objective value is also zero. Using the results of the LP solved in Phase 2, we will generate a schedule in Phase 3 which is represented as a set of batches, including the information of start time, finish time, family and machine.

In Phase 3, batches are generated. In machine 1, a batch exists with size 1108. In machine 3, batches are 1727, 528, 673, 864, 100, 2900 and 1489 sized batches, in machine 4, batches with size 2304 and 507 are produced. These production quantities form a batch and the batches form a schedule with start times, and times, corresponding family and the machine information. After the formation of the batches, they are sequenced in ascending order according to their due dates starting from the earliest due date.

The sequence of batches in machine 2 is: first the one with 1727 size, then 528 sized, then 673, 864, 100, 2900 sized and finally the one with size 1489. Sequence of batches at machine 4 is: first 2304 sized and the 507 sized batch.

Assignment of these batches to jobs is done like this: 1108 sized batch on machine 1 produces jobs sized 1000, 600 and 900. 1727 sized batch on machine 3 produces jobs

1400 and 900, and 528 sized batch produces job sized 100. The batch with size 673 on machine 3 is for job 900 and job 100. The batch with size 864 is for job with size 900. Job with size 100 is produced by the batch with size 100 on machine 3 and jobs with sizes 400, 200 and 500 are produced by batch sized 2900 on machine 3. Batch sized 1489 produces job with size 1500 on machine 3. On machine 4, we have batch with size 2304 which is producing jobs with sizes 300, 700, 800 and 1000. Job with size 1000 and job with size 1500 are produced by batch with size 507 on machine 4.

Job with size 1400 finishes at time 1, job with size 300 is produced at time 1, job with size 700 is produced at time 2, job with size 800 is produced at time 5, job with size 900 is produced at time 3, job with size 400 is produced at time 6, job with size 600 is produced at time 2, job with size 100 is produced at time 4, , job with size 100 is produced at time 3, job with size 1000 is produced at time 2, job with size 2000 is produced at time 6, job with size 500 is produced at time 7, job with size 900 is produced at time 4, job with size 1000 is produced at time 5, job with size 1500 is produced at time 9. According to these finishing times, tardiness of all jobs is zero.

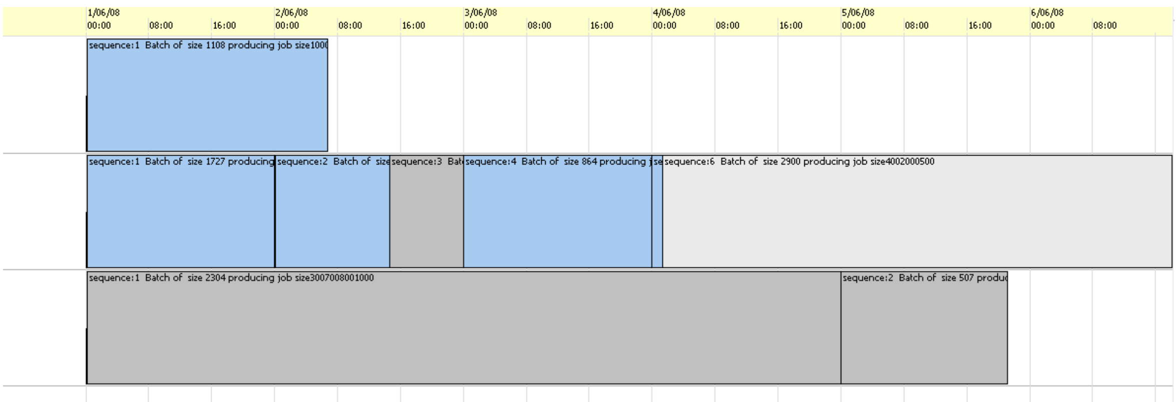

Figure 4.4. Gantt Chart of example 2

4.5. Modeling with ICRON

In this part, we introduce the methodology used in design and implementation phase of the study . In order to implement the heuristic that we propose, and also compare performance of our heuristic over Sansarcı (2007), we implement these heuristics in ICRON. For making the implementation, we have used object oriented methodology, object oriented mathematical programming and Graphical Scheduling Algorithm Modeling System (GSAMS) module of ICRON software.

In order to apply the proposed heuristics explained in the previous chapters, we have used ICRON software. ICRON is an optimization system developed in C++ to provide Advanced Planning and Scheduling (APS) and Capacity Planning (CP) solutions.

ICRON is based on object oriented data models which involves classes and their relationships. In object oriented methodology, real objects are represented in the by class definition and their particular instances, namely objects of these classes. In ICRON, user can make any class definition having attributes that may also refer to other objects in the system. Algorithms can be constructed based on these class definitions which are methods of the associated class in order to model the system and execute as a solution procedure.

Graphical scheduling algorithm generation process is based on visualization with node and link structures. User does not have to know any coding language and ICRON requires no experience in software development from the user. ICRON provides an environment for users to develop algorithmic modeling of variety of problems due to its generic system architecture and also modeling of mathematical programming problems. An example for demonstrating ICRON environment is given in Figure 4.5.

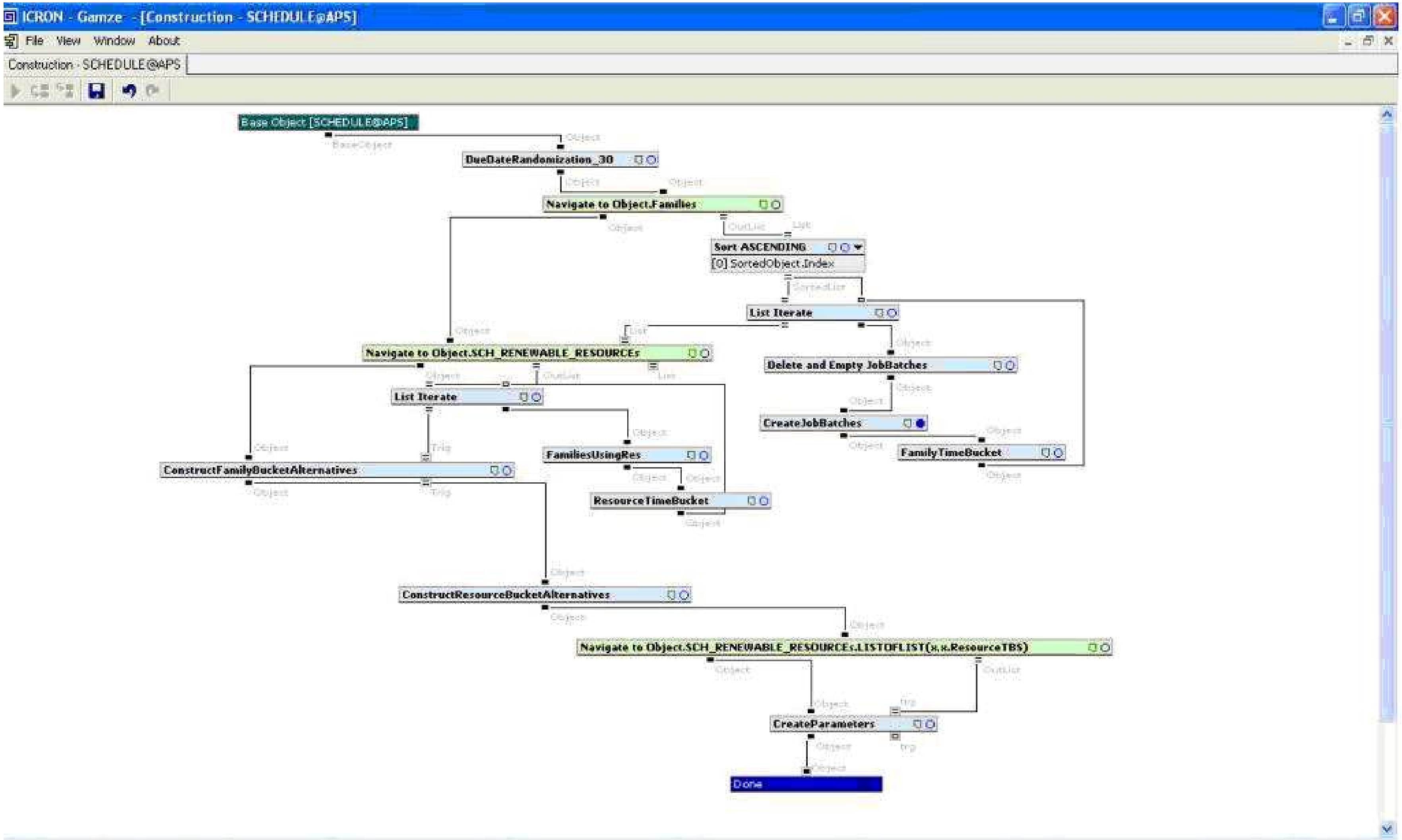

Figure 4.5. ICRON Environment

5. SIMULATED ANNEALING APPROACH

For finding good control parameters for each family, a simulated annealing approach is used in this study. The details of the simulated annealing procedure are provided in this section. Given an initial family control parameters sequence, a new sequence is created by neighborhood generation.

Performance of the schedules generated by each neighborhood is compared based on total tardiness values. The new control parameters sequence is accepted if its total tardiness is smaller than the previous sequence. If the new total tardiness value is equal to the previous total tardiness value, keeping the new tardiness value (abbreviate it as: equal) in mind, another sequence is found by neighborhood generation, if the total tardiness value found by that sequence (abbreviate it as: after_equal) is lower than equal value, than the control parameter sequence which generates after_equal value is accepted, otherwise the parameter sequence which generates equal value is accepted. If the new total tardiness value is bigger than the previous total tardiness value, the new control parameter sequence is accepted with some probability which decreases as the process evolves. This acceptance probability depends on a temperature value which is set to higher levels in initial iterations of the process and then this temperature value is lowered (cooled) in later iterations. As the stopping criterion is met, the smallest objective value found is selected as the solution of the simulated annealing approach.

5.1. Neighborhood Generation

The simulated annealing tries to find better total tardiness values by examining the solution space. In order to examine the solution space, neighborhood generation attempts are made. By changing the sequence of family control parameters, a new neighborhood is generated and then the total tardiness value due to this new neighborhood is calculated and the new tardiness value is compared with the old tardiness value.

Generation of the new sequence is accomplished in two ways. Either by swapping control parameter values of two families, or by changing (increasing or decreasing) control parameters of one family. While making change move, we decrease or increase control parameters of a randomly selected family by a constant percentage amount. This percentage is important since when it is low, then that will result with small changes in results between two neighbourhods. So the search space of simulated annealing will be bounded by that small percentage and will be a small search area. But when the percentage is at higher values, then difference between two neighbourhood results will be higher and which makes the search ineffective.

In this study, we select to use 5 percent as the percent change between two neighbourhood values. Also when making swap move, the family to be swapped is the family which includes the job with maximum tardiness value since it is wise to change the parameter values of this family in order to look if some adjustments can lower the total tardiness value. Other family to be swapped with the family which includes the job maximum tardiness value and the family whose control parameters are changed are selected randomly.

5.2. Acceptance Decision

In simulated annealing the procedure not only goes to better solutions, but it can also jump to worse solutions. The decision to jump to a worse solution is done based on an acceptance probability calculation. This acceptance probability is based on difference between the new solution and the old solution and also to temperature value. The temperature value is set to a higher level initially and decreased later.

Probability calculation is based on an exponential function where is the difference between the new tardiness value found by the neighborhood family control parameter sequence and the old tardiness value found by the previous family control parameter sequence.

$$\mathrm{P(acceptance)} = e^{-\Delta/T} \tag{5.1}$$

Temperature value is set initially to a higher value which is calculated as:

$$\left(\binom{\text{number of families}}{2}\right) * 4 * \text{number of families} * 10^{10}. \tag{5.2}$$

Temperature value is calculated like that since we can have $\binom{\text{number of families}}{2}$ different combinations of swap moves and for the other move. Since each family have two control parameters and we can decrease one parameter, increase the other; increase one parameter, decrease the other; decrease both or increase both. All these moves are also done randomly. The temperature value is divided into 10 after each 10 iterations.

Acceptance probability is calculated and is compared with a random number between 0 and 1. If acceptance probability is larger than the random number, the new sequence is accepted although the new total tardiness value is a worse one. If acceptance probability is smaller than the random variable, then the new sequence is not accepted and procedure goes back to previous family parameter sequence.

5.3. Stopping Criterion

The simulated annealing procedure terminates when the solution is equal to the lower bound or it reaches maximum number of non-improving solutions. Lower bound is calculated as the solution in which jobs are scheduled in ascending order according to their due dates and with zero setup. Maximum number of non-improving solutions is calculated $\left[\binom{\text{number of families}}{2} * 4 * \text{number of families}\right]$ in order to include a logical number of neighborhood generation moves.

6. EXPERIMENTAL STUDY

In this section, we will first justify that making control parameters different for each family gives better solutions than fixing the control parameters for all families. We make this justification by generating a set of problem instances. Then we will give an example presenting the simulated annealing approach. Moreover; we will compare solutions of the work of Sansarcı (2007) and our solution. Then we will investigate the effects of factors on the performance of the heuristic.

6.1. Need for Different Parameters

Sansarcı (2007) kept the control parameters the same for all families. But it is better to give different control parameters for each family since each family has different job combinations. In order to look whether changing values of the control parameters are effective on the performance of the heuristic, we generate 629 problem instances. In these instances the max earliness and production per setup values are set to different values and the heuristic is run for each instance yielding the total tardiness value of all jobs.

Assume we have 3 machines, 100 jobs and 2 families. The parameters are kept the same for family 1 and for family 2, the parameters are changed. Maximum difference value is 6 days for family 1 and production per setup value is 50. For family 2, maximum difference value can take on 17 different values. These are ranging between 1 day and 9 days. Production per setup value can take on 37 different values ranging from 10 to 360. The results according to changing values of the control parameters are shown in Table 6.1 as a matrix. Tardiness is given as hours in the results.

Table 6.1: Results of control parameters

	1	**1,5**	**2**	**2,5**	**3**	**3,5**	**4**	**4,5**	**5**	**5,5**	**6**	**6,5**	**7**	**7,5**	**8**	**8,5**	**9**
360	605	598	598	592	592	592	592	592	592	592	592	592	592	592	592	592	592
350	605	598	598	592	592	592	592	592	592	592	592	592	592	592	592	592	592
340	605	605	605	618	618	592	592	592	592	592	592	592	592	592	592	592	592
330	605	605	605	618	618	618	618	618	618	618	618	618	618	618	618	618	618
320	605	605	605	618	618	618	618	618	618	618	618	618	618	618	618	618	618
310	605	592	592	598	598	599	599	599	599	599	599	599	599	599	599	599	599
300	605	592	592	598	598	599	599	599	599	599	599	599	599	599	599	599	599
290	605	598	598	592	592	598	598	598	598	598	598	598	598	598	598	598	598
280	605	602	602	605	605	605	605	605	605	605	605	605	605	605	605	605	605
270	605	602	602	605	605	605	605	605	605	605	605	605	605	605	605	605	605
260	605	602	602	605	605	605	605	605	605	605	605	605	605	605	605	605	605
250	605	610	610	602	602	602	602	602	602	602	602	602	602	602	602	602	602
240	605	602	602	610	610	610	610	610	610	610	610	610	610	610	610	610	610
230	605	602	602	616	616	616	616	616	616	616	616	616	616	616	616	616	616
220	605	598	598	618	618	618	618	618	618	618	618	618	618	618	618	618	618
210	605	602	602	610	610	610	610	610	610	610	610	610	610	610	610	610	610
200	605	602	602	596	596	596	596	596	596	596	596	596	596	596	596	596	596
190	605	610	610	610	610	610	610	610	610	610	610	610	610	610	610	610	610
180	605	602	602	610	610	610	610	610	610	610	610	610	610	610	610	610	610
170	605	602	602	610	610	610	610	610	610	610	610	610	610	610	610	610	610
160	605	613	613	613	613	613	613	613	613	613	613	613	613	613	613	613	613
150	605	605	605	613	613	613	613	613	613	613	613	613	613	613	613	613	613
140	605	602	602	596	596	596	596	596	596	596	596	596	596	596	596	596	596
130	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605
120	605	592	592	592	592	592	592	592	592	592	592	592	592	592	592	592	592
110	605	605	605	613	613	613	613	613	613	613	613	613	613	613	613	613	613
100	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605
90	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605
80	605	618	618	618	618	618	618	618	618	618	618	618	618	618	618	618	618
70	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605
60	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605
50	605	614	614	614	614	614	614	614	614	614	614	614	614	614	614	614	614
40	605	614	614	614	614	614	614	614	614	614	614	614	614	614	614	614	614
30	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605
20	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605
10	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605	605

As we can see from the results matrix, the results are changing in different combinations of control parameters. Although the results are not the same for all parameter combinations, the results are not converging. We can't generalize the structure of the change in results according to change in parameter values. The changing results according to different parameter combinations are presented in Figure 6.1. From the figure,

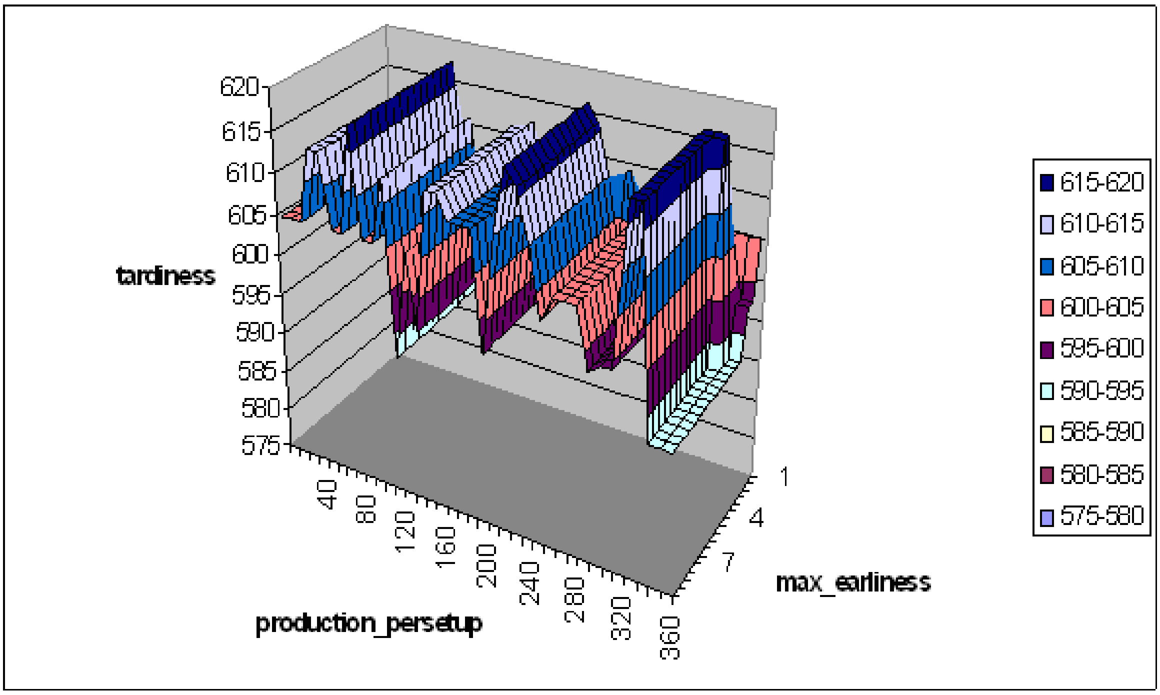

Figure 6.1. Solution space for changing parameters

we can say that results are changing over a non- linear space for different parameter combinations. Also we can't generalize the results for change in production per setup and maximum_difference values, like we can't say that it is decreasing or increasing when parameters are increased. But more importantly, we can say that it is changing for different parameter combinations, it can take on lower and higher values.

Also we can figure out that changing parameter values for families are a good approach since for the fixed parameter values 50 for production per setup and 6 for maximum_difference the result is not a better one among other results. Maximum value is 618 hours and the result when we fix the parameters for all families is 614 hours. Whereas we can get minimum result for example when production per setup value is 360 and maximum_difference is 6 days. This result justifies our claim that changing control parameter values for each family is an acceptable claim in which we can get lower objective values.

Since the solution space is a non-linear one, it is wise to use search techniques for control parameter values of families. In this study, we decide to use simulated annealing algorithm for searching the values of control parameters.

6.2. Simulated Annealing Procedure

For examining results of the simulated annealing used, an example is presented. Assume we have 4 machines, 100 jobs and 3 families. In family 1 we have 33 jobs. In family 2, we have 36 jobs. In family 3, we have 31 jobs. Jobs of family 1 can be produced on second and third machines where jobs of family 2 can be produced on first, second and third machines. Jobs of family 3 can be produced on first and fourth machines. Control parameter values for family 1 are: production per setup value: 100 and maximum_difference value: 172800 minutes (2 days), and control parameter values for family 2 are: production per setup value: 150 and maximum_difference value: 259200 minutes (3 days), and control parameter values for family 3 is: production per setup value: 260 and maximum_difference value: 345600 minutes (4 days).

So initial control parameter sequence is: "100-172800(2 days), 150-259200(3 days), 260-345600(4 days)". In Table A.1 of Appendix A, results of simulated annealing are shown. Family control parameters sequence, the resulting total tardiness and the neighborhood generation move used can be seen in that table. For example we start with sequence "100-172800, 150-259200, 260-345600 " in which the total tardiness value is 444. Then a new sequence which is a neighborhood sequence of previous sequence is generated by making a change move by changing the control parameters of family 3 which results with 456 total tardiness value, then a new sequence is generated also by making a change move resulting with an objective value 458. Then a swap move is done by changing the control parameters of family 2 and 3 resulting with a total tardiness value of 335, after that a change move is done by changing control parameters of family 2. The procedure goes on in this manner. After applying simulated annealing procedure, we generate different solutions in different family control parameter combinations. The best total tardiness value is the solution of simulated annealing. In this example the minimum value of total tardiness is found as 298. In Figure 6.3., results of simulated annealing are shown in a chart. As can be seen from the figure, procedure can jump to worse values as well as better total tardiness values.

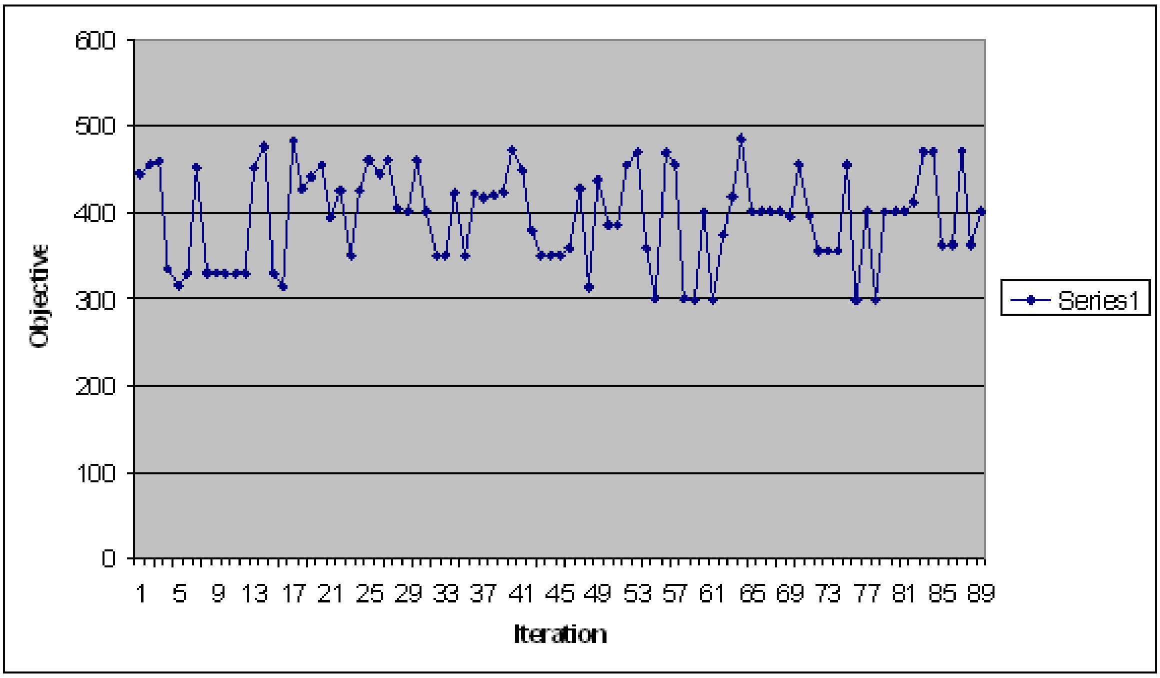

Figure 6.2. Results of simulated annealing

6.3. Problem Generation

Four factors are considered in order to generate problem instances. There are different levels of these factors. These factors are number of machines, number of families, due date structure and eligible machine structure.

Levels of the four factors considered are:

- Two levels are generated for number of machines. These are: 3 or 7
- Three levels are generated for number of families. These are: 3, 5 or 7
- Three levels are generated for due date structure: U (0, 10), U (0, 20), or U (0, 30)
- Two different levels are generated for number of eligible machines for each family: U (0.3, 0.7)*(number of machines) or U (0.1, 0.9)*(number of machines)

In order to see the performance of the proposed heuristic under different levels of the factors, an experimental design is prepared. Since there are two levels of number of machines, three levels of number of families, three levels of due date structure and two levels of number of eligible machines for each family, we have 36 different combinations of these factors.

For all these different combinations of levels, we generate problem instances with 100 jobs. These problem instances are solved by the heuristic proposed by Sansarcı (2007) and also the heuristic proposed in this study. Results of these two works are compared for these 36 factor combinations. Results of the two compared heuristics acording to each factor combination is in Table 6.2.

Also 15 problem instances are generated for each factor combination. So we generate 540 problems at all and solve each problem instance with these two heuristics. Detailed results can be seen in Table B.5. of Appendix B.

Table 6.2: Results in Different Factor Combinations

Number of machines	**Number of families**	**Number of eligible machines**	**Due date structure**	**Other Heuristic**	**Proposed Heuristic**	**Difference (%)**
3	3	U (0.3, 0.7)	U (0, 10)	1949	1851	5
3	3	U (0.3, 0.7)	U (0, 20)	1779	1548	13
3	3	U (0.3, 0.7)	U (0, 30)	1682	1494	11
3	3	U (0.1, 0.9)	U (0, 10)	1644	1582	4
3	3	U (0.1, 0.9)	U (0, 20)	1650	1584	4
3	3	U (0.1, 0.9)	U (0, 30)	1574	914	42
3	5	U (0.3, 0.7)	U (0, 10)	1727	1552	10
3	5	U (0.3, 0.7)	U (0, 20)	1418	1280	10
3	5	U (0.3, 0.7)	U (0, 30)	1402	1086	23
3	5	U (0.1, 0.9)	U (0, 10)	1750	1483	15
3	5	U (0.1, 0.9)	U (0, 20)	1538	1313	15
3	5	U (0.1, 0.9)	U (0, 30)	1440	896	38
3	7	U (0.3, 0.7)	U (0, 10)	2109	1842	13
3	7	U (0.3, 0.7)	U (0, 20)	2313	1716	26
3	7	U (0.3, 0.7)	U (0, 30)	1790	1574	12
3	7	U (0.1, 0.9)	U (0, 10)	1879	1961	-4
3	7	U (0.1, 0.9)	U (0, 20)	2130	1824	14
3	7	U (0.1, 0.9)	U (0, 30)	2022	1807	11
7	3	U (0.3, 0.7)	U (0, 10)	174	103	41
7	3	U (0.3, 0.7)	U (0, 20)	76	30	60
7	3	U (0.3, 0.7)	U (0, 30)	93	13	86
7	3	U (0.1, 0.9)	U (0, 10)	263	234	11
7	3	U (0.1, 0.9)	U (0, 20)	151	95	37
7	3	U (0.1, 0.9)	U (0, 30)	152	74	51
7	5	U (0.3, 0.7)	U (0, 10)	760	599	21
7	5	U (0.3, 0.7)	U (0, 20)	461	233	49

Continued on Next Page…

Number of machines	Number of families	Number of eligible machines	Due date structure	Other Heuristic	Proposed Heuristic	Difference (%)
7	5	U (0.3, 0.7)	U (0, 30)	263	105	60
7	5	U (0.1, 0.9)	U (0, 10)	812	617	24
7	5	U (0.1, 0.9)	U (0, 20)	684	513	25
7	5	U (0.1, 0.9)	U (0, 30)	378	359	5
7	7	U (0.3, 0.7)	U (0, 10)	790	738	7
7	7	U (0.3, 0.7)	U (0, 20)	602	379	37
7	7	U (0.3, 0.7)	U (0, 30)	461	226	51
7	7	U (0.1, 0.9)	U (0, 10)	896	725	19
7	7	U (0.1, 0.9)	U (0, 20)	743	656	12
7	7	U (0.1, 0.9)	U (0, 30)	460	401	13

6.4. Comparison of Solutions

There are 36 combinations of factors for each of which 15 problem instances are solved. When we look at results of experiments for these 36 combinations, we see that the heuristic that we suggest gives better solutions in 35 factor combinations. It only gives worse results in one factor combination which is sixteenth factor combination. Moreover; in the sixteenth combination, our heuristic gives 7 better results where the other heuristic gives 8 better results out of 15 results which is not an important difference.

For making this comparison statistically, we perform paired t-test. In order to compare whether the means of results of the two algorithms are equal or not; we will use paired t-test. The paired t test provides a hypothesis test of the difference between population means for a pair of random samples whose differences are approximately normally distributed. In this case; the first population is the results found by using the other heuristic and the second population is the results of our proposed heuristic.

In paired t-test; we use difference of the observations. We will take the null hypothesis H_0 that the difference of means of the first and second observations is 0; and the alternative hypothesis H_1 that the difference is not 0. The general paired t test formation is given in the following:

$$\text{Null Hypothesis: } H_0 : \mu_1 - \mu_2 = 0$$

$$\text{Test Statistic Value: } T = \frac{\overline{d}}{S_d/\sqrt{n}}$$

$(\overline{d})$ is the sample mean of differences and S_d is the standard deviation of the differences

$$S_d{}^2 = \sum_{i=1}^{n} \frac{[d_i - \overline{d}]^2}{n-1} \tag{6.1}$$

Alternative Hypothesis:	Rejection Level for H_0
$H_1 : \mu_1 - \mu_2 > 0$	$T \geq t_{\alpha,n-1}$
$H_1 : \mu_1 - \mu_2 < 0$	$T \leq -t_{\alpha,n-1}$
$H_1 : \mu_1 - \mu_2 \neq 0$	either $T \leq -t_{\alpha,n-1}$ or $T \geq t_{\alpha,n-1}$

After making necessary calculations, $\overline{d}$ is found as 183.5556 and S_d is found as 156.7118. Since we have 36 different factor combinations n is equal to 36. So resulting t statistic value is found as 7.027761

In the investigated case; we have a power value of 0, 90 and α level of 0.05. So, the t-value that we have to compare with the test statistic value we found is: $t_{\alpha/2,n-1} = t_{0.025,35}$ as 2.021.

Since $T > 2.021$; we reject H_0 which is $\mu_1 - \mu_2 = 0$. Now; we will determine which one of the two heuristics is better by looking at the alternative hypothesis.

Alternative Hypothesis:	Rejection Level for H_0
$H_1 : \mu_1 - \mu_2 > 0$	$T \geq t_{\alpha,n-1}$
$H_1 : \mu_1 - \mu_2 < 0$	$T \leq -t_{\alpha,n-1}$

For making this decision, $t_{\alpha,n-1} = t_{0.05,35}$ is 1.684 and we will compare this with our test statistic value.

Since $T > 1.684$; we will say $\mu_1 - \mu_2 > 0$ which means that difference between the result of other heuristic and our heuristic is greater than zero.

That result shows that our proposed heuristic gives smaller total tardiness values than the other heuristic. So; it is more logical to use our heuristic in our problem.

6.5. Effects of Factors on Suggested Heuristic

Problem set is generated considering different levels of four factors which are machine factor, family factor, eligibility factor and due date factor. In this section effects of these four factors will be studied by presenting results according to different levels of each factor. Also results based on statistical analysis will be given.

6.6. Effect of Machine Factor

For the number of machines factor, we have two levels which are 3 machines as the low level and the other is 7 machines as the high level. Results with machine number 3 are shown at the first column and results with 7 machines are shown at the second column in table B.1 of Appendix B.

Average result with 3 machines is 1533 where average result with 7 machines is 346. This results with a 78 percent difference between two levels. We can say from this result that machine number factor is affecting result of our suggested heuristic. This result is logical since when number of machines is increased, we can finish jobs earlier than when the number of machines is smaller. With increasing number of machines; we can produce a job on more machines and can finish job earlier. So; tardy jobs are produced earlier, resulting with a lower total tardiness value.

In order to investigate whether machine factor has an effect on our heuristic or not, we use statistical t-test again. The null hypothesis in the test is: difference between levels is zero. So, null hypothesis says that different levels of machine factor doesn't result in different results. Since we have 15 different problem instances, degrees of freedom is 14 and α level is 0.05. So resulting t value is $t_{0,025,14}$ which is 2.145. Since test statistic value found by calculations is 13.0749371 ; we reject the null hypothesis and conclude that machine factor is affecting our heuristic.

6.7. Effect of Family Factor

For the number of machines factor, we have three levels which are 3 families as the low level, 5 families as the medium level and the other is 7 families as the high level. Results according to changing levels of family number factor are shown in table B.2 of Appendix B.

Average result when there are 3 families is 804, average result with 5 families is 856 and average result when number of families is 7 comes as 1159. This results with a 30 percent difference between low and high levels and 26 percent diference between medium and high levels. As we can see from these results, total tardiness value is increasing when number of families is increased. This result is also making sense when we think the family dependent setup aspect of our problem. When number of families is increased, occurrence of family dependent setups between batches is increasing. With increasing time spend for family dependent setup; production of a job is shifted on later times. So; tardy jobs are produced later, resulting with a higher total tardiness value.

For searching the effect of family factor on our heuristic, the t-test is used. The null hypothesis in the test is: difference between levels is zero. The t-value that we have to compare is 2.145. Since test statistic value found by calculations is -5.915707393 ; we reject the null hypothesis which results in conclusion that family factor is affecting our heuristic.

6.8. Effect of Eligibility Factor

Eligibility factor for each family has two levels. The first level is U (0.3, 0.7) and the second one is U (0.1, 0.9). Results with eligibility factor U (0.3, 0.7) are shown at the first column and results with eligibility factor U (0.1, 0.9) are shown at the second column in table B.3 of Appendix B.

Average result with eligibility factor U (0.3, 0.7) is 908 where average result with eligibility factor U (0.1, 0.9) is 971. So there is a 6 percent difference between two levels. We can say from this result that eligibility factor has a slight effect on results of our suggested heuristic.

T-test is used for examining the effect of eligibility factor on our heuristic. The t-value that we have to compare which is $t_{0.025,14}$ is 2.145. Since test statistic value found by calculations is -0.967936931 which is lower than 2.145; we conclude that eligibility factor is not affecting our heuristic.

6.9. Effect of Due Date Factor

For the due date factor, we have three levels which are U (0, 10) as the low level, U (0, 20) as the medium level and the other is U (0, 30) as the high level. Results according to these three levels of family number factor are shown in table B.4 of Appendix B.

Average result when due date factor level is at its low level is 1112, average result when due date factor is U (0, 20) is 948 and average result when due date factor is at its high level is 757. So there is a 32 percent difference between low and high levels and 15 percent difference between medium and low levels.

From these results, we can conclude that when due date factor is affecting results of our heuristic. With increasing due date interval, maximum tardiness value is decreasing since due dates of jobs are more different and more far away from each other and also jobs have further due dates. More jobs can be produced with lower tardiness values since their due dates are on later times. So; maximum tardiness value is decreasing when due date factor is at higher levels.

For examining the effect of due date factor on our heuristic, statistical t-test is used again. The test statistic value found after calculations is compared with the t value that is corresponding to degrees of freedom and confidence interval. Since degress of freedom is 14 and confidence interval is 90, t-value is found as $t_{0,05,14}$. The test statistic value is compared with $t_{0,05,14}$ which is 2.145. Comparison showed that the test statistic value found by calculations which is 7.335507479 is larger than 2.145; so we can conclude that due date factor is affecting our heuristic.

7. SUMMARY AND CONCLUSIONS

In manufacturing facilities, one of the most important goals is to satisfy customer demand on time. Being tardy makes companies to lose money and more importantly to lose trust of their customers. So it is a significant issue not to produce jobs later than their due dates. Facilities have to produce products efficiently and use their capacity in a logical way. Also in facilities which produce different types of products, this need for effective capacity usage plays an important role. Capacity has to be shared effectively among these different product types.

In many situations where different types of products are produced, a setup time has to be realized in between these different product types. But this situation is opposed to the objective of being not tardy since end of production times of jobs are shifted to later times. Due to this production environment, scheduling plays an important role. Also, manufacturing systems which produce different products usually share machines for those products. This situation increases the importance of scheduling, too.

The problem we focus in this study is a very common problem that many manufacturing facilities encounter. In our problem, there are several parallel machines with different technological properties in order to produce different customer orders for different product families. Problem has family dependent type of setup structure which means there is not a setup between jobs of the same family but there is a setup time between productions of two families on the same machine consecutively.

Problem has also machine eligibility aspect in which jobs of a family can't be produced on all machines since machines have different technological properties. Only machines which have technology that is appropriate for a family can produce the family. This makes the problem more complex to solve. Job splitting is another property of the problem we concern. Production of a job can be split into small productions on different machines and/or different time buckets.

In scheduling literature, there are articles related to our problem but the only one considering our problem is Sansarcı (2007). Other articles handle parts of our problem like job splitting and eligibility. Since the problem we focus in this study is a complex problem with properties like machine eligibility, family dependent setup structure and job splitting, there is hard to find much related articles. In chapter two, these articles relating our problem are presented.

In section three of this work, the studied problem is described in detail and formal problem definition of the problem is given by mathematical model. For solving the problem defined, we suggest a heuristic consisting of three phases. Details of the proposed heuristic are given in section four of this study.

Heuristic consists of three phases. In the first phase, jobs of a particular family are aggregated according to control parameters. These control parameters are production per setup and maximum difference value. When combining jobs, these parameters decide whether to go on combining or to stop. Production per setup is a measure of time spend for production according to time spend for setup. It forces system to produce more so that it is worth to make setup for that production. It is logical that a control like this exists since making too much setup and producing in fewer amounts is not a well situation when we think from economical aspect of view. The other parameter, maximum difference determines if the differences of the due dates of jobs are acceptable for aggregation. This control parameter is also making sense since combing jobs with very different due dates is not acceptable in many cases. If total production of jobs that are aggregated is smaller than production per setup parameter multiplies by setup time and maximum difference between jobs is smaller than maximum difference, then aggregation of jobs is continued.

After completing first phase, created job batches are used for generating time buckets which are used as inputs of phase two. In phase two, production values in these time buckets are found by solving LP. These production quantities are inputs of phase three for generating job batches. Jobs are assigned to job batches generated in phase three so that production time and tardiness values for all jobs can be found.

In our study, we use simulated annealing approach for finding good control parameter values for families. Starting with an initial family control parameters sequence, simulated annealing procedure is applied by moving to neighborhoods. Solutions at each neighborhood are found by the suggested heuristic and according to simulated annealing logic that we use, appropriate moves are done.

In experimentation section of this study, solutions of Sansarcı (2007) and our study are compared. In other study, control parameters for all families are kept the same whereas control parameters of families are different in our study. Since job structures of all families are not the same, it makes sense that control parameters of each family should be different than each other. For making this comparison between two studies, problem instances are generated at different levels of factors which are number of families, number of machines, eligibility structure and due date structure.

In order to implement heuristic of Sansarcı (2007) and our heuristic, ICRON is used. Both heuristics are modeled in ICRON and problems are solved by each heuristic. There are 36 combinations for these factor levels and for each factor combination, 15 problems are generated resulting in 540 problem instances. Each problem instance is solved by heuristic of Sansarcı (2007) and our heuristic. Results of experimentations showed that in only one factor combination the other heuristic is better than our heuristic which is a very good performance.

Also from experimentations, effects of factors over the heuristic that we concern are also investigated. Results showed that number of families is affecting our heuristic since when number of families is increased, tardiness increases. Also number of machines is affecting our heuristic due to decrease in tardiness when number of families increase. Due date structure is another factor that affects our heuristic since when due date interval is increased, tardiness decreases.

As guide to further researches, different control parameters when aggregating jobs can be used in later studies like a control parameter considering demand of jobs that are aggregated.

Moreover for making comparison between the other heuristic and other heuristic, different performance evaluation criterion can be used like maximum completion time, value or total tardiness value.

Also different search methods can be used other than simulated annealing like genetic algorithm or tabu search. A different approach can be used in simulated annealing like other neighborhood generation methods.

Later studies may also try to solve problem optimally. Although it is a high possibility that it will take large amount of time for finding optimum solutions, it may worth to try for that. So that optimum results can be compared by result found by our heuristic.

APPENDIX A: Results of Simulated Annealing

Table A.1: Results of simulated annealing

OBJECTIVE	FAMILY PARAMETERS	NEIGHBOURHOOD
444	100-172800 150-259200 260-345600	
456	100-172800 150-259200 273-362880	change
458	100-172800 157.5-272160 273-362880	change
335	100-172800 273-362880 157.5-272160	swap
316	100-172800 286.7-344736 157.5-272160	change
330	100-172800 286.7-344736 149.6-285768	change
451	100-172800 149.6-285768 286.7-344736	swap
330	100-172800 286.7-344736 149.6-285768	swap
330	100-172800 301-327499.2 149.6-285768	change
330	100-172800 316.1-343874.2 149.6-285768	change
330	100-172800 300.3-326680.5 149.6-285768	change
330	100-172800 285.3-310346.5 149.6-285768	change
451	100-172800 149.6-285768 285.3-310346.5	swap
477	100-172800 149.6-285768 271-325863.8	change
330	100-172800 271-325863.8 149.6-285768	swap
314	105-181440 271-325863.8 149.6-285768	change
483	105-181440 149.6-285768 271-325863.8	swap
428	105-181440 149.6-285768 284.6-342157	change
441	105-181440 142.1-271479.6 284.6-342157	change
456	99.8-172368 142.1-271479.6 284.6-342157	change
394	99.8-172368 284.6-342157 142.1-271479.6	swap
425	284.6-342157 99.8-172368 142.1-271479.6	swap
351	284.6-342157 142.1-271479.6 99.8-172368	swap
425	284.6-342157 99.8-172368 142.1-271479.6	swap
461	142.1-271479.6 99.8-172368 284.6-342157	swap
445	142.1-271479.6 99.8-172368 270.4-325049.1	change
461	142.1-271479.6 99.8-172368 283.9-341301.6	change
404	142.1-271479.6 283.9-341301.6 99.8-172368	swap
401	149.2-257905.6 283.9-341301.6 99.8-172368	change
461	149.2-257905.6 99.8-172368 283.9-341301.6	swap
401	149.2-257905.6 283.9-341301.6 99.8-172368	swap

Continued on Next Page. . .

OBJECTIVE	FAMILY PARAMETERS	NEIGHBOURHOOD
351	283.9-341301.6 149.2-257905.6 99.8-172368	swap
351	283.9-341301.6 141.7-245010.3 99.8-172368	change
422	283.9-341301.6 99.8-172368 141.7-245010.3	swap
351	283.9-341301.6 141.7-245010.3 99.8-172368	swap
422	283.9-341301.6 99.8-172368 141.7-245010.3	swap
417	283.9-341301.6 99.8-172368 134.6-257260.8	change
420	283.9-341301.6 94.8-180986.4 134.6-257260.8	change
424	283.9-341301.6 94.8-180986.4 141.3-270123.8	change
473	141.3-270123.8 94.8-180986.4 283.9-341301.6	swap
449	141.3-270123.8 94.8-180986.4 269.7-324236.5	change
379	141.3-270123.8 269.7-324236.5 94.8-180986.4	swap
351	269.7-324236.5 141.3-270123.8 94.8-180986.4	swap
351	283.2-308024.7 141.3-270123.8 94.8-180986.4	change
351	297.4-323425.9 141.3-270123.8 94.8-180986.4	change
360	297.4-323425.9 148.4-283630 94.8-180986.4	change
428	94.8-180986.4 148.4-283630 297.4-323425.9	swap
314	94.8-180986.4 297.4-323425.9 148.4-283630	swap
438	297.4-323425.9 94.8-180986.4 148.4-283630	swap
385	297.4-323425.9 99.5-190035.7 148.4-283630	change
385	282.5-307254.6 99.5-190035.7 148.4-283630	change
455	282.5-307254.6 99.5-190035.7 155.8-269448.5	change
469	282.5-307254.6 94.5-199537.5 155.8-269448.5	change
360	282.5-307254.6 155.8-269448.5 94.5-199537.5	swap
300	282.5-307254.6 163.6-255976.1 94.5-199537.5	change
469	282.5-307254.6 94.5-199537.5 163.6-255976.1	swap
455	282.5-307254.6 99.2-209514.4 163.6-255976.1	change
300	282.5-307254.6 163.6-255976.1 99.2-209514.4	swap
298	282.5-307254.6 163.6-255976.1 104.2-219990.1	change
401	163.6-255976.1 282.5-307254.6 104.2-219990.1	swap
298	282.5-307254.6 163.6-255976.1 104.2-219990.1	swap
374	282.5-307254.6 171.8-243177.3 104.2-219990.1	change
418	282.5-307254.6 104.2-219990.1 171.8-243177.3	swap
485	171.8-243177.3 104.2-219990.1 282.5-307254.6	swap
401	171.8-243177.3 282.5-307254.6 104.2-219990.1	swap
401	180.4-231018.4 282.5-307254.6 104.2-219990.1	change

Continued on Next Page. . .

OBJECTIVE	FAMILY PARAMETERS	NEIGHBOURHOOD
401	171.4-242569.3 282.5-307254.6 104.2-219990.1	change
401	171.4-242569.3 282.5-307254.6 109.4-208990.6	change
396	109.4-208990.6 282.5-307254.6 171.4-242569.3	swap
456	109.4-208990.6 171.4-242569.3 282.5-307254.6	swap
396	109.4-208990.6 282.5-307254.6 171.4-242569.3	swap
356	109.4-208990.6 282.5-307254.6 162.8-230440.8	change
356	109.4-208990.6 296.6-322617.3 162.8-230440.8	change
356	109.4-208990.6 281.8-306486.4 162.8-230440.8	change
455	281.8-306486.4 109.4-208990.6 162.8-230440.8	swap
298	281.8-306486.4 162.8-230440.8 109.4-208990.6	swap
401	162.8-230440.8 281.8-306486.4 109.4-208990.6	swap
298	281.8-306486.4 162.8-230440.8 109.4-208990.6	swap
401	162.8-230440.8 281.8-306486.4 109.4-208990.6	swap
401	154.7-241962.8 281.8-306486.4 109.4-208990.6	change
401	154.7-241962.8 281.8-306486.4 114.9-219440.1	change
411	281.8-306486.4 154.7-241962.8 114.9-219440.1	swap
471	114.9-219440.1 154.7-241962.8 281.8-306486.4	swap
471	114.9-219440.1 147-254060.9 281.8-306486.4	change
362	114.9-219440.1 281.8-306486.4 147-254060.9	swap
362	109.2-208468.1 281.8-306486.4 147-254060.9	change
471	109.2-208468.1 147-254060.9 281.8-306486.4	swap
362	109.2-208468.1 281.8-306486.4 147-254060.9	swap
401	109.2-208468.1 281.8-306486.4 139.7-241357.9	change

APPENDIX B: Results of Experimentations

Table B.1. Effect of machine factor

MACHINE NUMBER: 3	**MACHINE NUMBER: 7**
1851	103
1548	30
1494	0,13
1644	234
1584	95
914	152
1552	599
1280	223
1086	105
1483	617
1538	513
896	378
1842	738
1716	379
1574	226
1961	725
1824	656
1807	460

Table B.2. Effect of family factor

FAMILY NUMBER: 3	**FAMILY NUMBER:5**	**FAMILY NUMBER:7**
1851	1552	1842
1548	1280	1716
1494	1086	1574
1644	1483	1961
1584	1538	1824
914	896	1807
103	599	738
30	223	379
0,13	105	226
234	617	725
95	513	656
152	378	460

Table B.3. Effect of eligibility factor

U (0.3, 0.7)	U (0.1, 0.9)
1851	1644
1548	1584
1494	914
1552	1483
1280	1538
1086	896
1842	1961
1716	1824
1574	1807
103	234
30	95
0,13	152
599	617
223	513
105	378
738	725
379	656
226	460

Table B.4. Effect of eligibility factor

U (0.3, 0.7)	U (0.1, 0.9)
1851	1644
1548	1584
1494	914
1552	1483
1280	1538
1086	896
1842	1961
1716	1824
1574	1807
103	234
30	95
0,13	152
599	617
223	513
105	378
738	725
379	656
226	460

Table B.5: Comparison of Heuristic Methods

Number of machines	Number of families	Number of eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
3	3	U (0.3, 0.7)	U (0, 10)	3022	2322
3	3	U (0.3, 0.7)	U (0, 10)	2649	1694
3	3	U (0.3, 0.7)	U (0, 10)	2605	1337
3	3	U (0.3, 0.7)	U (0, 10)	1399	2830
3	3	U (0.3, 0.7)	U (0, 10)	1416	2890
3	3	U (0.3, 0.7)	U (0, 10)	1875	2353
3	3	U (0.3, 0.7)	U (0, 10)	1432	1353
3	3	U (0.3, 0.7)	U (0, 10)	2649	1720
3	3	U (0.3, 0.7)	U (0, 10)	1473	1330
3	3	U (0.3, 0.7)	U (0, 10)	1629	1333
3	3	U (0.3, 0.7)	U (0, 10)	1403	1345
3	3	U (0.3, 0.7)	U (0, 10)	3040	1711
3	3	U (0.3, 0.7)	U (0, 10)	1416	2863
3	3	U (0.3, 0.7)	U (0, 10)	1473	1360
3	3	U (0.3, 0.7)	U (0, 10)	1759	1337
3	3	U (0.3, 0.7)	U (0, 20)	2636	1165
3	3	U (0.3, 0.7)	U (0, 20)	1487	2155
3	3	U (0.3, 0.7)	U (0, 20)	1191	1497
3	3	U (0.3, 0.7)	U (0, 20)	1176	1116
3	3	U (0.3, 0.7)	U (0, 20)	2255	2605
3	3	U (0.3, 0.7)	U (0, 20)	1436	1219
3	3	U (0.3, 0.7)	U (0, 20)	2814	1112
3	3	U (0.3, 0.7)	U (0, 20)	1597	2110
3	3	U (0.3, 0.7)	U (0, 20)	2674	1093
3	3	U (0.3, 0.7)	U (0, 20)	2654	1473
3	3	U (0.3, 0.7)	U (0, 20)	1452	1232

Continued on Next Page...

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
3	3	U (0.3, 0.7)	U (0, 20)	1447	1494
3	3	U (0.3, 0.7)	U (0, 20)	1452	1239
3	3	U (0.3, 0.7)	U (0, 20)	1176	1079
3	3	U (0.3, 0.7)	U (0, 20)	1245	2645
3	3	U (0.3, 0.7)	U (0, 30)	2547	1841
3	3	U (0.3, 0.7)	U (0, 30)	988	2396
3	3	U (0.3, 0.7)	U (0, 30)	1498	1849
3	3	U (0.3, 0.7)	U (0, 30)	2588	814
3	3	U (0.3, 0.7)	U (0, 30)	2525	846
3	3	U (0.3, 0.7)	U (0, 30)	1034	1207
3	3	U (0.3, 0.7)	U (0, 30)	925	862
3	3	U (0.3, 0.7)	U (0, 30)	2413	940
3	3	U (0.3, 0.7)	U (0, 30)	2477	829
3	3	U (0.3, 0.7)	U (0, 30)	1352	2351
3	3	U (0.3, 0.7)	U (0, 30)	933	1845
3	3	U (0.3, 0.7)	U (0, 30)	975	943
3	3	U (0.3, 0.7)	U (0, 30)	2610	918
3	3	U (0.3, 0.7)	U (0, 30)	1450	2367
3	3	U (0.3, 0.7)	U (0, 30)	927	2402
3	3	U (0.1, 0.9)	U (0, 10)	1694	2646
3	3	U (0.1, 0.9)	U (0, 10)	1494	1669
3	3	U (0.1, 0.9)	U (0, 10)	1266	903
3	3	U (0.1, 0.9)	U (0, 10)	1344	1358
3	3	U (0.1, 0.9)	U (0, 10)	1342	1540
3	3	U (0.1, 0.9)	U (0, 10)	1366	1403
3	3	U (0.1, 0.9)	U (0, 10)	1368	1351
3	3	U (0.1, 0.9)	U (0, 10)	1756	3061
3	3	U (0.1, 0.9)	U (0, 10)	2834	1416

Continued on Next Page...

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
3	3	U (0.1, 0.9)	U (0, 10)	1266	1519
3	3	U (0.1, 0.9)	U (0, 10)	1481	1461
3	3	U (0.1, 0.9)	U (0, 10)	1336	1571
3	3	U (0.1, 0.9)	U (0, 10)	2326	1115
3	3	U (0.1, 0.9)	U (0, 10)	1482	1597
3	3	U (0.1, 0.9)	U (0, 10)	2315	1115
3	3	U (0.1, 0.9)	U (0, 20)	2636	852
3	3	U (0.1, 0.9)	U (0, 20)	1321	2086
3	3	U (0.1, 0.9)	U (0, 20)	2186	569
3	3	U (0.1, 0.9)	U (0, 20)	2136	2595
3	3	U (0.1, 0.9)	U (0, 20)	1292	2089
3	3	U (0.1, 0.9)	U (0, 20)	2774	1097
3	3	U (0.1, 0.9)	U (0, 20)	1272	1472
3	3	U (0.1, 0.9)	U (0, 20)	1285	2091
3	3	U (0.1, 0.9)	U (0, 20)	2186	563
3	3	U (0.1, 0.9)	U (0, 20)	1273	979
3	3	U (0.1, 0.9)	U (0, 20)	1283	2086
3	3	U (0.1, 0.9)	U (0, 20)	1278	2605
3	3	U (0.1, 0.9)	U (0, 20)	1198	2642
3	3	U (0.1, 0.9)	U (0, 20)	1535	1066
3	3	U (0.1, 0.9)	U (0, 20)	1094	968
3	3	U (0.1, 0.9)	U (0, 30)	2547	1243
3	3	U (0.1, 0.9)	U (0, 30)	1921	733
3	3	U (0.1, 0.9)	U (0, 30)	1147	1257
3	3	U (0.1, 0.9)	U (0, 30)	1118	464
3	3	U (0.1, 0.9)	U (0, 30)	1005	383
3	3	U (0.1, 0.9)	U (0, 30)	1964	616
3	3	U (0.1, 0.9)	U (0, 30)	1092	1001

Continued on Next Page. . .

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
3	3	U (0.1, 0.9)	U (0, 30)	486	1870
3	3	U (0.1, 0.9)	U (0, 30)	2523	358
3	3	U (0.1, 0.9)	U (0, 30)	1098	389
3	3	U (0.1, 0.9)	U (0, 30)	1383	2347
3	3	U (0.1, 0.9)	U (0, 30)	2527	846
3	3	U (0.1, 0.9)	U (0, 30)	1922	766
3	3	U (0.1, 0.9)	U (0, 30)	895	567
3	3	U (0.1, 0.9)	U (0, 30)	1989	867
3	5	U (0.3, 0.7)	U (0, 10)	2482	2293
3	5	U (0.3, 0.7)	U (0, 10)	1414	1269
3	5	U (0.3, 0.7)	U (0, 10)	1477	1256
3	5	U (0.3, 0.7)	U (0, 10)	1617	2299
3	5	U (0.3, 0.7)	U (0, 10)	1485	1335
3	5	U (0.3, 0.7)	U (0, 10)	2473	1350
3	5	U (0.3, 0.7)	U (0, 10)	1639	2289
3	5	U (0.3, 0.7)	U (0, 10)	1580	1435
3	5	U (0.3, 0.7)	U (0, 10)	1520	1295
3	5	U (0.3, 0.7)	U (0, 10)	1535	1401
3	5	U (0.3, 0.7)	U (0, 10)	2500	1337
3	5	U (0.3, 0.7)	U (0, 10)	1383	1497
3	5	U (0.3, 0.7)	U (0, 10)	1502	1376
3	5	U (0.3, 0.7)	U (0, 10)	1928	1328
3	5	U (0.3, 0.7)	U (0, 10)	1367	1521
3	5	U (0.3, 0.7)	U (0, 20)	2154	1006
3	5	U (0.3, 0.7)	U (0, 20)	1181	976
3	5	U (0.3, 0.7)	U (0, 20)	1279	1136
3	5	U (0.3, 0.7)	U (0, 20)	1195	1216
3	5	U (0.3, 0.7)	U (0, 20)	1471	1056

Continued on Next Page...

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
3	5	U (0.3, 0.7)	U (0, 20)	1252	1643
3	5	U (0.3, 0.7)	U (0, 20)	1081	1228
3	5	U (0.3, 0.7)	U (0, 20)	1639	1643
3	5	U (0.3, 0.7)	U (0, 20)	1487	1341
3	5	U (0.3, 0.7)	U (0, 20)	1533	1056
3	5	U (0.3, 0.7)	U (0, 20)	1252	1643
3	5	U (0.3, 0.7)	U (0, 20)	1081	1228
3	5	U (0.3, 0.7)	U (0, 20)	1639	1643
3	5	U (0.3, 0.7)	U (0, 20)	1487	1341
3	5	U (0.3, 0.7)	U (0, 20)	1533	1053
3	5	U (0.3, 0.7)	U (0, 30)	1932	741
3	5	U (0.3, 0.7)	U (0, 30)	1135	1807
3	5	U (0.3, 0.7)	U (0, 30)	1094	896
3	5	U (0.3, 0.7)	U (0, 30)	877	1835
3	5	U (0.3, 0.7)	U (0, 30)	765	982
3	5	U (0.3, 0.7)	U (0, 30)	1912	924
3	5	U (0.3, 0.7)	U (0, 30)	1022	987
3	5	U (0.3, 0.7)	U (0, 30)	1608	771
3	5	U (0.3, 0.7)	U (0, 30)	2067	1418
3	5	U (0.3, 0.7)	U (0, 30)	1932	889
3	5	U (0.3, 0.7)	U (0, 30)	1496	930
3	5	U (0.3, 0.7)	U (0, 30)	1466	1340
3	5	U (0.3, 0.7)	U (0, 30)	827	764
3	5	U (0.3, 0.7)	U (0, 30)	992	974
3	5	U (0.3, 0.7)	U (0, 30)	1912	1034
3	5	U (0.1, 0.9)	U (0, 10)	2482	2293
3	5	U (0.1, 0.9)	U (0, 10)	1414	882
3	5	U (0.1, 0.9)	U (0, 10)	1634	2307

Continued on Next Page. . .

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
3	5	U (0.1, 0.9)	U (0, 10)	2434	1110
3	5	U (0.1, 0.9)	U (0, 10)	1293	1565
3	5	U (0.1, 0.9)	U (0, 10)	2482	2293
3	5	U (0.1, 0.9)	U (0, 10)	1414	882
3	5	U (0.1, 0.9)	U (0, 10)	1634	2307
3	5	U (0.1, 0.9)	U (0, 10)	2434	1110
3	5	U (0.1, 0.9)	U (0, 10)	1293	1565
3	5	U (0.1, 0.9)	U (0, 10)	2503	1106
3	5	U (0.1, 0.9)	U (0, 10)	1111	1233
3	5	U (0.1, 0.9)	U (0, 10)	1506	1106
3	5	U (0.1, 0.9)	U (0, 10)	1135	1420
3	5	U (0.1, 0.9)	U (0, 10)	1495	1074
3	5	U (0.1, 0.9)	U (0, 20)	737	2154
3	5	U (0.1, 0.9)	U (0, 20)	2075	1496
3	5	U (0.1, 0.9)	U (0, 20)	1136	1082
3	5	U (0.1, 0.9)	U (0, 20)	665	1304
3	5	U (0.1, 0.9)	U (0, 20)	1211	873
3	5	U (0.1, 0.9)	U (0, 20)	1050	2114
3	5	U (0.1, 0.9)	U (0, 20)	1702	2112
3	5	U (0.1, 0.9)	U (0, 20)	1223	843
3	5	U (0.1, 0.9)	U (0, 20)	2166	803
3	5	U (0.1, 0.9)	U (0, 20)	2116	1608
3	5	U (0.1, 0.9)	U (0, 20)	1620	1113
3	5	U (0.1, 0.9)	U (0, 20)	2067	718
3	5	U (0.1, 0.9)	U (0, 20)	1598	1495
3	5	U (0.1, 0.9)	U (0, 20)	2087	705
3	5	U (0.1, 0.9)	U (0, 20)	1624	1273
3	5	U (0.1, 0.9)	U (0, 30)	1932	766

Continued on Next Page. . .

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
3	5	U (0.1, 0.9)	U (0, 30)	705	921
3	5	U (0.1, 0.9)	U (0, 30)	1472	1394
3	5	U (0.1, 0.9)	U (0, 30)	1969	956
3	5	U (0.1, 0.9)	U (0, 30)	2021	738
3	5	U (0.1, 0.9)	U (0, 30)	1151	560
3	5	U (0.1, 0.9)	U (0, 30)	991	429
3	5	U (0.1, 0.9)	U (0, 30)	1369	963
3	5	U (0.1, 0.9)	U (0, 30)	1980	910
3	5	U (0.1, 0.9)	U (0, 30)	816	600
3	5	U (0.1, 0.9)	U (0, 30)	1041	975
3	5	U (0.1, 0.9)	U (0, 30)	1912	653
3	5	U (0.1, 0.9)	U (0, 30)	1090	830
3	5	U (0.1, 0.9)	U (0, 30)	1922	867
3	5	U (0.1, 0.9)	U (0, 30)	1228	1891
3	7	U (0.3, 0.7)	U (0, 10)	3097	1579
3	7	U (0.3, 0.7)	U (0, 10)	1751	1606
3	7	U (0.3, 0.7)	U (0, 10)	2531	1591
3	7	U (0.3, 0.7)	U (0, 10)	1818	2028
3	7	U (0.3, 0.7)	U (0, 10)	2400	1605
3	7	U (0.3, 0.7)	U (0, 10)	2170	1766
3	7	U (0.3, 0.7)	U (0, 10)	1685	1884
3	7	U (0.3, 0.7)	U (0, 10)	2050	2036
3	7	U (0.3, 0.7)	U (0, 10)	1829	2036
3	7	U (0.3, 0.7)	U (0, 10)	1788	2523
3	7	U (0.3, 0.7)	U (0, 10)	1698	1680
3	7	U (0.3, 0.7)	U (0, 10)	2796	1571
3	7	U (0.3, 0.7)	U (0, 10)	1807	1935
3	7	U (0.3, 0.7)	U (0, 10)	2451	2136

Continued on Next Page. . .

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
3	7	U (0.3, 0.7)	U (0, 10)	1765	1660
3	7	U (0.3, 0.7)	U (0, 20)	2728	2203
3	7	U (0.3, 0.7)	U (0, 20)	2764	1454
3	7	U (0.3, 0.7)	U (0, 20)	2011	1348
3	7	U (0.3, 0.7)	U (0, 20)	2493	1401
3	7	U (0.3, 0.7)	U (0, 20)	3033	1843
3	7	U (0.3, 0.7)	U (0, 20)	3191	1401
3	7	U (0.3, 0.7)	U (0, 20)	1912	1404
3	7	U (0.3, 0.7)	U (0, 20)	1742	1861
3	7	U (0.3, 0.7)	U (0, 20)	2593	2056
3	7	U (0.3, 0.7)	U (0, 20)	2644	2071
3	7	U (0.3, 0.7)	U (0, 20)	1514	2189
3	7	U (0.3, 0.7)	U (0, 20)	1392	1316
3	7	U (0.3, 0.7)	U (0, 20)	1560	1706
3	7	U (0.3, 0.7)	U (0, 20)	2426	2008
3	7	U (0.3, 0.7)	U (0, 20)	2692	1482
3	7	U (0.3, 0.7)	U (0, 30)	1929	1092
3	7	U (0.3, 0.7)	U (0, 30)	1676	1142
3	7	U (0.3, 0.7)	U (0, 30)	2227	1678
3	7	U (0.3, 0.7)	U (0, 30)	1251	2471
3	7	U (0.3, 0.7)	U (0, 30)	1821	1816
3	7	U (0.3, 0.7)	U (0, 30)	1251	2392
3	7	U (0.3, 0.7)	U (0, 30)	2257	1269
3	7	U (0.3, 0.7)	U (0, 30)	2070	1089
3	7	U (0.3, 0.7)	U (0, 30)	2329	1705
3	7	U (0.3, 0.7)	U (0, 30)	1428	1656
3	7	U (0.3, 0.7)	U (0, 30)	1652	1570
3	7	U (0.3, 0.7)	U (0, 30)	1197	1085

Continued on Next Page. . .

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
3	7	U (0.3, 0.7)	U (0, 30)	2032	1483
3	7	U (0.3, 0.7)	U (0, 30)	1179	1341
3	7	U (0.3, 0.7)	U (0, 30)	2552	1835
3	7	U (0.1, 0.9)	U (0, 10)	3121	1211
3	7	U (0.1, 0.9)	U (0, 10)	1807	1909
3	7	U (0.1, 0.9)	U (0, 10)	1906	2207
3	7	U (0.1, 0.9)	U (0, 10)	2647	1626
3	7	U (0.1, 0.9)	U (0, 10)	1440	1106
3	7	U (0.1, 0.9)	U (0, 10)	2101	1682
3	7	U (0.1, 0.9)	U (0, 10)	1154	2464
3	7	U (0.1, 0.9)	U (0, 10)	1423	2398
3	7	U (0.1, 0.9)	U (0, 10)	1423	2398
3	7	U (0.1, 0.9)	U (0, 10)	1040	1486
3	7	U (0.1, 0.9)	U (0, 10)	2758	2666
3	7	U (0.1, 0.9)	U (0, 10)	2050	2036
3	7	U (0.1, 0.9)	U (0, 10)	1829	2036
3	7	U (0.1, 0.9)	U (0, 10)	1788	2523
3	7	U (0.1, 0.9)	U (0, 10)	1698	1680
3	7	U (0.1, 0.9)	U (0, 20)	2684	1602
3	7	U (0.1, 0.9)	U (0, 20)	1871	1353
3	7	U (0.1, 0.9)	U (0, 20)	2002	2111
3	7	U (0.1, 0.9)	U (0, 20)	2026	2186
3	7	U (0.1, 0.9)	U (0, 20)	2147	1814
3	7	U (0.1, 0.9)	U (0, 20)	1896	2328
3	7	U (0.1, 0.9)	U (0, 20)	2629	1783
3	7	U (0.1, 0.9)	U (0, 20)	2040	2745
3	7	U (0.1, 0.9)	U (0, 20)	2463	870
3	7	U (0.1, 0.9)	U (0, 20)	1843	1630

Continued on Next Page. . .

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
3	7	U (0.1, 0.9)	U (0, 20)	2684	2869
3	7	U (0.1, 0.9)	U (0, 20)	1090	1022
3	7	U (0.1, 0.9)	U (0, 20)	2355	2193
3	7	U (0.1, 0.9)	U (0, 20)	1866	1146
3	7	U (0.1, 0.9)	U (0, 20)	2347	1713
3	7	U (0.1, 0.9)	U (0, 30)	2549	1817
3	7	U (0.1, 0.9)	U (0, 30)	2236	2071
3	7	U (0.1, 0.9)	U (0, 30)	2430	1829
3	7	U (0.1, 0.9)	U (0, 30)	2523	2174
3	7	U (0.1, 0.9)	U (0, 30)	1312	1393
3	7	U (0.1, 0.9)	U (0, 30)	2472	1675
3	7	U (0.1, 0.9)	U (0, 30)	1812	2328
3	7	U (0.1, 0.9)	U (0, 30)	2123	1978
3	7	U (0.1, 0.9)	U (0, 30)	871	1996
3	7	U (0.1, 0.9)	U (0, 30)	2159	2298
3	7	U (0.1, 0.9)	U (0, 30)	2552	1835
3	7	U (0.1, 0.9)	U (0, 30)	1642	1844
3	7	U (0.1, 0.9)	U (0, 30)	1701	1178
3	7	U (0.1, 0.9)	U (0, 30)	1963	1661
3	7	U (0.1, 0.9)	U (0, 30)	1991	1038
7	3	U (0.3, 0.7)	U (0, 10)	363	143
7	3	U (0.3, 0.7)	U (0, 10)	186	125
7	3	U (0.3, 0.7)	U (0, 10)	154	139
7	3	U (0.3, 0.7)	U (0, 10)	161	13
7	3	U (0.3, 0.7)	U (0, 10)	335	21
7	3	U (0.3, 0.7)	U (0, 10)	27	51
7	3	U (0.3, 0.7)	U (0, 10)	79	134
7	3	U (0.3, 0.7)	U (0, 10)	342	10

Continued on Next Page...

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
7	3	U (0.3, 0.7)	U (0, 10)	153	51
7	3	U (0.3, 0.7)	U (0, 10)	57	14
7	3	U (0.3, 0.7)	U (0, 10)	168	47
7	3	U (0.3, 0.7)	U (0, 10)	54	324
7	3	U (0.3, 0.7)	U (0, 10)	162	9
7	3	U (0.3, 0.7)	U (0, 10)	353	144
7	3	U (0.3, 0.7)	U (0, 10)	20	324
7	3	U (0.3, 0.7)	U (0, 20)	4	123
7	3	U (0.3, 0.7)	U (0, 20)	145	0
7	3	U (0.3, 0.7)	U (0, 20)	132	0
7	3	U (0.3, 0.7)	U (0, 20)	153	0
7	3	U (0.3, 0.7)	U (0, 20)	153	0
7	3	U (0.3, 0.7)	U (0, 20)	0	0
7	3	U (0.3, 0.7)	U (0, 20)	174	3
7	3	U (0.3, 0.7)	U (0, 20)	3	111
7	3	U (0.3, 0.7)	U (0, 20)	121	0
7	3	U (0.3, 0.7)	U (0, 20)	120	91
7	3	U (0.3, 0.7)	U (0, 20)	3	0
7	3	U (0.3, 0.7)	U (0, 20)	125	0
7	3	U (0.3, 0.7)	U (0, 20)	0	125
7	3	U (0.3, 0.7)	U (0, 20)	6	0
7	3	U (0.3, 0.7)	U (0, 20)	3	0
7	3	U (0.3, 0.7)	U (0, 30)	3	0
7	3	U (0.3, 0.7)	U (0, 30)	4	0
7	3	U (0.3, 0.7)	U (0, 30)	16	0
7	3	U (0.3, 0.7)	U (0, 30)	4	0
7	3	U (0.3, 0.7)	U (0, 30)	4	0
7	3	U (0.3, 0.7)	U (0, 30)	25	0

Continued on Next Page...

No. of Machines	**No. of Families**	**No. of Eligible machines**	**Due date structure**	**Other Heuristic**	**Proposed Heuristic**
7	3	U (0.3, 0.7)	U (0, 30)	0	0
7	3	U (0.3, 0.7)	U (0, 30)	3	0
7	3	U (0.3, 0.7)	U (0, 30)	25	0
7	3	U (0.3, 0.7)	U (0, 30)	4	0
7	3	U (0.3, 0.7)	U (0, 30)	16	2
7	3	U (0.3, 0.7)	U (0, 30)	3	0
7	3	U (0.3, 0.7)	U (0, 30)	4	0
7	3	U (0.3, 0.7)	U (0, 30)	4	0
7	3	U (0.3, 0.7)	U (0, 30)	4	0
7	3	U (0.1, 0.9)	U (0, 10)	3	3
7	3	U (0.1, 0.9)	U (0, 10)	188	139
7	3	U (0.1, 0.9)	U (0, 10)	16	128
7	3	U (0.1, 0.9)	U (0, 10)	3	131
7	3	U (0.1, 0.9)	U (0, 10)	20	3
7	3	U (0.1, 0.9)	U (0, 10)	4	3
7	3	U (0.1, 0.9)	U (0, 10)	185	918
7	3	U (0.1, 0.9)	U (0, 10)	204	934
7	3	U (0.1, 0.9)	U (0, 10)	961	3
7	3	U (0.1, 0.9)	U (0, 10)	924	43
7	3	U (0.1, 0.9)	U (0, 10)	193	13
7	3	U (0.1, 0.9)	U (0, 10)	882	3
7	3	U (0.1, 0.9)	U (0, 10)	3	913
7	3	U (0.1, 0.9)	U (0, 10)	184	140
7	3	U (0.1, 0.9)	U (0, 20)	728	0
7	3	U (0.1, 0.9)	U (0, 20)	3	0
7	3	U (0.1, 0.9)	U (0, 20)	4	0
7	3	U (0.1, 0.9)	U (0, 20)	4	0
7	3	U (0.1, 0.9)	U (0, 20)	4	716

Continued on Next Page...

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
7	3	U (0.1, 0.9)	U (0, 20)	27	713
7	3	U (0.1, 0.9)	U (0, 20)	6	0
7	3	U (0.1, 0.9)	U (0, 20)	0	0
7	3	U (0.1, 0.9)	U (0, 20)	0	0
7	3	U (0.1, 0.9)	U (0, 20)	23	0
7	3	U (0.1, 0.9)	U (0, 20)	3	0
7	3	U (0.1, 0.9)	U (0, 20)	740	0
7	3	U (0.1, 0.9)	U (0, 20)	704	0
7	3	U (0.1, 0.9)	U (0, 20)	3	0
7	3	U (0.1, 0.9)	U (0, 20)	23	0
7	3	U (0.1, 0.9)	U (0, 30)	0	4
7	3	U (0.1, 0.9)	U (0, 30)	0	4
7	3	U (0.1, 0.9)	U (0, 30)	0	3
7	3	U (0.1, 0.9)	U (0, 30)	0	0
7	3	U (0.1, 0.9)	U (0, 30)	0	2
7	3	U (0.1, 0.9)	U (0, 30)	466	3
7	3	U (0.1, 0.9)	U (0, 30)	453	3
7	3	U (0.1, 0.9)	U (0, 30)	439	4
7	3	U (0.1, 0.9)	U (0, 30)	466	508
7	3	U (0.1, 0.9)	U (0, 30)	0	12
7	3	U (0.1, 0.9)	U (0, 30)	0	548
7	3	U (0.1, 0.9)	U (0, 30)	0	17
7	3	U (0.1, 0.9)	U (0, 30)	0	3
7	3	U (0.1, 0.9)	U (0, 30)	466	3
7	3	U (0.1, 0.9)	U (0, 30)	0	4
7	5	U (0.3, 0.7)	U (0, 10)	864	1006
7	5	U (0.3, 0.7)	U (0, 10)	770	643
7	5	U (0.3, 0.7)	U (0, 10)	1119	401

Continued on Next Page. . .

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
7	5	U (0.3, 0.7)	U (0, 10)	778	440
7	5	U (0.3, 0.7)	U (0, 10)	749	597
7	5	U (0.3, 0.7)	U (0, 10)	487	337
7	5	U (0.3, 0.7)	U (0, 10)	537	448
7	5	U (0.3, 0.7)	U (0, 10)	728	830
7	5	U (0.3, 0.7)	U (0, 10)	549	845
7	5	U (0.3, 0.7)	U (0, 10)	1159	807
7	5	U (0.3, 0.7)	U (0, 10)	536	794
7	5	U (0.3, 0.7)	U (0, 10)	1159	761
7	5	U (0.3, 0.7)	U (0, 10)	845	312
7	5	U (0.3, 0.7)	U (0, 10)	712	372
7	5	U (0.3, 0.7)	U (0, 10)	418	406
7	5	U (0.3, 0.7)	U (0, 20)	566	255
7	5	U (0.3, 0.7)	U (0, 20)	406	217
7	5	U (0.3, 0.7)	U (0, 20)	654	278
7	5	U (0.3, 0.7)	U (0, 20)	668	303
7	5	U (0.3, 0.7)	U (0, 20)	390	268
7	5	U (0.3, 0.7)	U (0, 20)	228	98
7	5	U (0.3, 0.7)	U (0, 20)	285	334
7	5	U (0.3, 0.7)	U (0, 20)	409	80
7	5	U (0.3, 0.7)	U (0, 20)	507	279
7	5	U (0.3, 0.7)	U (0, 20)	162	238
7	5	U (0.3, 0.7)	U (0, 20)	433	195
7	5	U (0.3, 0.7)	U (0, 20)	484	311
7	5	U (0.3, 0.7)	U (0, 20)	502	259
7	5	U (0.3, 0.7)	U (0, 20)	295	69
7	5	U (0.3, 0.7)	U (0, 20)	928	161
7	5	U (0.3, 0.7)	U (0, 30)	578	301

Continued on Next Page...

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
7	5	U (0.3, 0.7)	U (0, 30)	142	27
7	5	U (0.3, 0.7)	U (0, 30)	264	33
7	5	U (0.3, 0.7)	U (0, 30)	61	231
7	5	U (0.3, 0.7)	U (0, 30)	142	25
7	5	U (0.3, 0.7)	U (0, 30)	244	12
7	5	U (0.3, 0.7)	U (0, 30)	230	43
7	5	U (0.3, 0.7)	U (0, 30)	420	357
7	5	U (0.3, 0.7)	U (0, 30)	222	71
7	5	U (0.3, 0.7)	U (0, 30)	220	36
7	5	U (0.3, 0.7)	U (0, 30)	120	24
7	5	U (0.3, 0.7)	U (0, 30)	65	46
7	5	U (0.3, 0.7)	U (0, 30)	144	51
7	5	U (0.3, 0.7)	U (0, 30)	795	285
7	5	U (0.3, 0.7)	U (0, 30)	312	43
7	5	U (0.1, 0.9)	U (0, 10)	1949	1162
7	5	U (0.1, 0.9)	U (0, 10)	792	483
7	5	U (0.1, 0.9)	U (0, 10)	339	262
7	5	U (0.1, 0.9)	U (0, 10)	289	276
7	5	U (0.1, 0.9)	U (0, 10)	761	241
7	5	U (0.1, 0.9)	U (0, 10)	407	1288
7	5	U (0.1, 0.9)	U (0, 10)	1026	279
7	5	U (0.1, 0.9)	U (0, 10)	365	326
7	5	U (0.1, 0.9)	U (0, 10)	2427	1400
7	5	U (0.1, 0.9)	U (0, 10)	369	700
7	5	U (0.1, 0.9)	U (0, 10)	379	430
7	5	U (0.1, 0.9)	U (0, 10)	321	1040
7	5	U (0.1, 0.9)	U (0, 10)	1071	764
7	5	U (0.1, 0.9)	U (0, 10)	1042	276

Continued on Next Page...

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
7	5	U (0.1, 0.9)	U (0, 10)	640	341
7	5	U (0.1, 0.9)	U (0, 20)	175	658
7	5	U (0.1, 0.9)	U (0, 20)	1619	737
7	5	U (0.1, 0.9)	U (0, 20)	473	533
7	5	U (0.1, 0.9)	U (0, 20)	645	414
7	5	U (0.1, 0.9)	U (0, 20)	334	1113
7	5	U (0.1, 0.9)	U (0, 20)	303	139
7	5	U (0.1, 0.9)	U (0, 20)	1174	119
7	5	U (0.1, 0.9)	U (0, 20)	585	235
7	5	U (0.1, 0.9)	U (0, 20)	136	998
7	5	U (0.1, 0.9)	U (0, 20)	417	630
7	5	U (0.1, 0.9)	U (0, 20)	1076	1094
7	5	U (0.1, 0.9)	U (0, 20)	522	533
7	5	U (0.1, 0.9)	U (0, 20)	600	97
7	5	U (0.1, 0.9)	U (0, 20)	557	68
7	5	U (0.1, 0.9)	U (0, 20)	1640	328
7	5	U (0.1, 0.9)	U (0, 30)	35	284
7	5	U (0.1, 0.9)	U (0, 30)	31	123
7	5	U (0.1, 0.9)	U (0, 30)	412	322
7	5	U (0.1, 0.9)	U (0, 30)	19	270
7	5	U (0.1, 0.9)	U (0, 30)	1374	42
7	5	U (0.1, 0.9)	U (0, 30)	1378	191
7	5	U (0.1, 0.9)	U (0, 30)	291	313
7	5	U (0.1, 0.9)	U (0, 30)	38	1391
7	5	U (0.1, 0.9)	U (0, 30)	32	249
7	5	U (0.1, 0.9)	U (0, 30)	294	443
7	5	U (0.1, 0.9)	U (0, 30)	294	114
7	5	U (0.1, 0.9)	U (0, 30)	292	353

Continued on Next Page. . .

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
7	5	U (0.1, 0.9)	U (0, 30)	850	341
7	5	U (0.1, 0.9)	U (0, 30)	291	902
7	5	U (0.1, 0.9)	U (0, 30)	49	41
7	7	U (0.3, 0.7)	U (0, 10)	802	656
7	7	U (0.3, 0.7)	U (0, 10)	587	488
7	7	U (0.3, 0.7)	U (0, 10)	696	490
7	7	U (0.3, 0.7)	U (0, 10)	603	491
7	7	U (0.3, 0.7)	U (0, 10)	825	483
7	7	U (0.3, 0.7)	U (0, 10)	968	547
7	7	U (0.3, 0.7)	U (0, 10)	715	766
7	7	U (0.3, 0.7)	U (0, 10)	614	937
7	7	U (0.3, 0.7)	U (0, 10)	642	569
7	7	U (0.3, 0.7)	U (0, 10)	580	956
7	7	U (0.3, 0.7)	U (0, 10)	723	519
7	7	U (0.3, 0.7)	U (0, 10)	820	1168
7	7	U (0.3, 0.7)	U (0, 10)	1259	1443
7	7	U (0.3, 0.7)	U (0, 10)	970	730
7	7	U (0.3, 0.7)	U (0, 10)	1042	826
7	7	U (0.3, 0.7)	U (0, 20)	678	666
7	7	U (0.3, 0.7)	U (0, 20)	703	341
7	7	U (0.3, 0.7)	U (0, 20)	424	330
7	7	U (0.3, 0.7)	U (0, 20)	454	444
7	7	U (0.3, 0.7)	U (0, 20)	679	318
7	7	U (0.3, 0.7)	U (0, 20)	515	350
7	7	U (0.3, 0.7)	U (0, 20)	481	319
7	7	U (0.3, 0.7)	U (0, 20)	461	373
7	7	U (0.3, 0.7)	U (0, 20)	553	322
7	7	U (0.3, 0.7)	U (0, 20)	537	373

Continued on Next Page. . .

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
7	7	U (0.3, 0.7)	U (0, 20)	624	342
7	7	U (0.3, 0.7)	U (0, 20)	701	310
7	7	U (0.3, 0.7)	U (0, 20)	777	336
7	7	U (0.3, 0.7)	U (0, 20)	569	328
7	7	U (0.3, 0.7)	U (0, 20)	867	538
7	7	U (0.3, 0.7)	U (0, 30)	435	103
7	7	U (0.3, 0.7)	U (0, 30)	586	87
7	7	U (0.3, 0.7)	U (0, 30)	862	89
7	7	U (0.3, 0.7)	U (0, 30)	541	314
7	7	U (0.3, 0.7)	U (0, 30)	479	399
7	7	U (0.3, 0.7)	U (0, 30)	515	385
7	7	U (0.3, 0.7)	U (0, 30)	588	90
7	7	U (0.3, 0.7)	U (0, 30)	246	328
7	7	U (0.3, 0.7)	U (0, 30)	393	355
7	7	U (0.3, 0.7)	U (0, 30)	283	290
7	7	U (0.3, 0.7)	U (0, 30)	549	136
7	7	U (0.3, 0.7)	U (0, 30)	482	351
7	7	U (0.3, 0.7)	U (0, 30)	211	68
7	7	U (0.3, 0.7)	U (0, 30)	224	79
7	7	U (0.3, 0.7)	U (0, 30)	526	326
7	7	U (0.1, 0.9)	U (0, 10)	650	978
7	7	U (0.1, 0.9)	U (0, 10)	633	1260
7	7	U (0.1, 0.9)	U (0, 10)	452	495
7	7	U (0.1, 0.9)	U (0, 10)	580	493
7	7	U (0.1, 0.9)	U (0, 10)	429	419
7	7	U (0.1, 0.9)	U (0, 10)	2017	403
7	7	U (0.1, 0.9)	U (0, 10)	531	420
7	7	U (0.1, 0.9)	U (0, 10)	625	696

Continued on Next Page. . .

No. of Machines	No. of Families	No. of Eligible machines	Due date structure	Other Heuristic	Proposed Heuristic
7	7	U (0.1, 0.9)	U (0, 10)	2102	500
7	7	U (0.1, 0.9)	U (0, 10)	571	370
7	7	U (0.1, 0.9)	U (0, 10)	445	1609
7	7	U (0.1, 0.9)	U (0, 10)	465	1160
7	7	U (0.1, 0.9)	U (0, 10)	1559	633
7	7	U (0.1, 0.9)	U (0, 10)	1108	951
7	7	U (0.1, 0.9)	U (0, 10)	1283	488
7	7	U (0.1, 0.9)	U (0, 20)	993	151
7	7	U (0.1, 0.9)	U (0, 20)	812	305
7	7	U (0.1, 0.9)	U (0, 20)	811	590
7	7	U (0.1, 0.9)	U (0, 20)	567	578
7	7	U (0.1, 0.9)	U (0, 20)	807	363
7	7	U (0.1, 0.9)	U (0, 20)	473	1002
7	7	U (0.1, 0.9)	U (0, 20)	665	1679
7	7	U (0.1, 0.9)	U (0, 20)	832	857
7	7	U (0.1, 0.9)	U (0, 20)	993	163
7	7	U (0.1, 0.9)	U (0, 20)	800	189
7	7	U (0.1, 0.9)	U (0, 20)	1139	858
7	7	U (0.1, 0.9)	U (0, 20)	386	515
7	7	U (0.1, 0.9)	U (0, 20)	498	1190
7	7	U (0.1, 0.9)	U (0, 20)	580	540
7	7	U (0.1, 0.9)	U (0, 20)	782	866
7	7	U (0.1, 0.9)	U (0, 30)	1421	350
7	7	U (0.1, 0.9)	U (0, 30)	53	86
7	7	U (0.1, 0.9)	U (0, 30)	43	957
7	7	U (0.1, 0.9)	U (0, 30)	994	123
7	7	U (0.1, 0.9)	U (0, 30)	370	306
7	7	U (0.1, 0.9)	U (0, 30)	454	848

Continued on Next Page...

No. of Machines	**No. of Families**	**No. of Eligible machines**	**Due date structure**	**Other Heuristic**	**Proposed Heuristic**
7	7	U (0.1, 0.9)	U (0, 30)	257	164
7	7	U (0.1, 0.9)	U (0, 30)	345	401
7	7	U (0.1, 0.9)	U (0, 30)	39	780
7	7	U (0.1, 0.9)	U (0, 30)	475	860
7	7	U (0.1, 0.9)	U (0, 30)	1167	588
7	7	U (0.1, 0.9)	U (0, 30)	469	75
7	7	U (0.1, 0.9)	U (0, 30)	41	185
7	7	U (0.1, 0.9)	U (0, 30)	734	71
7	7	U (0.1, 0.9)	U (0, 30)	43	218

REFERENCES

Balasubramanian, H., L. Monch, J. Fowler and M. Pfund, 2004 , "A Genetic Algorithm Based Scheduling of Parallel Batch Machines with Incompatible Job Families to Minimize Total Weighted Tardiness", *International Journal of Production Research*, Vol. 42, pp. 1621 - 1638.

Bank, J. and F. Werner, 2001, "Heuristic Algorithms for Unrelated Parallel Machine Scheduling with a Common Due Date, Release Dates, and Linear Earliness and Tardiness Penalties", *Mathematical and Computer Modelling*, Vol. 33, pp. 363-383.

Bekkia, O. B. and M. Azizoglu, 2007, "Operational Fixed Interval Scheduling Problem on Uniform Parallel Machines", *International Journal of Production Economics.*

Brucker, P., M. Y. Kovalyov, Y. M. Shafransky and F. Werner, 1998 , "Batch Scheduling with Deadlines on Parallel Machine", *Annals of Operations Research*, Vol. 83, pp. 23-40.

Centeno, G. and R. L. Armacost, 1997, "Parallel Machine Scheduling with Release Time and Machine Eligibility Restrictions", *Computers ind. Engng*,Vol. 33, Nos 1-2. , pp. 273-276, 1997.

Chen, Z. L., W. B. Powell, 2003, "Exact Algorithms for Scheduling Multiple Families of Jobs on Parallel Machines", *Naval Research Logistics*,Vol. 50, pp. 823-840, 2003.

Chen, Z. L., 1997, "Scheduling with Batch Setup Times and Earliness-Tardiness Penalties", *European Journal of Operational Research*,Vol. 96, pp. 518-537, 1997.

Chen, J. F., 2006, "Minimization of Maximum Tardiness on Unrelated Parallel Machines with Process Restrictions and Setups", *Int J Adv Manuf Technol*,Vol. 29, pp. 557-563, 2006.

Dunstalla, S., A. Wirth, 2005, "Heuristic methods for The Identical Parallel Machine Flowtime Problem with Set-up Times", *Computers and Operations Research*,Vol. 32, pp. 2479-2491, 2005.

Ghirardi, M. and C. N. Potts, 2005, "Make span Minimization for Scheduling Unrelated Parallel Machines: A Recovering Beam Search Approach", *European Journal of Operational Research*,Vol. 165, pp. 457-467, 2005.

Kim Y. D., S. O. Shim, S. B. Kim, Y. C. Choi and H. M. Yoon, 2004 , "Parallel Machine Scheduling Considering a Job-Splitting Property", *International Journal of Production Research*, Vol. 42, pp. 4531 - 4546, 2004.

Liao, L. W. and G. J. Sheen, 2007, "Parallel Machine Scheduling with Machine Availability and Eligibility Constraints", *European Journal of Operational Research*,Vol. 184, pp. 458-467, 2007.

Liaw, C. F., Y. D. Lin, C. Y. Cheng and M. Chen, 2003 , "Scheduling Unrelated Parallel Machines to Minimize Total Weighted Tardiness", *Computers and Operations Research*, Vol. 30, pp. 1777-1789, 2003.

Logendran, R. and F. Subur, 2004, "Unrelated Parallel Machine Scheduling with Job Splitting", *IIE Transactions*,Vol. 36, pp. 359 - 372, 2004.

Logendrana, R., B. McDonell and B. Smuckera, 2007 , "Scheduling Unrelated Parallel Machines with Sequence-Dependent Setups", *Computers and Operations Research*, Vol. 34, pp. 3420 - 3438, 2007.

Omar, M. K., S. C. Teo, 2006, "Minimizing The Sum of Earliness/Tardiness in Identical Parallel Machines Schedule with Incompatible Job Families: An Improved MIP Approach", *Applied Mathematics and Computation*, Vol. 181, pp. 1008-1017, 2006.

Rabadi, G., R. J. Moraga and A. A. Salem, 2006 , "Heuristics for the Unrelated Parallel Machine Scheduling Problem with Setup Times", *Journal of Intelligent*

Manufacturing, Vol. 17, pp. 85-97, 2006.

Rojanasoonthon S., 2004, “Parallel Machine Scheduling with Time Windows”, *Ph.D. Thesis ,University of Texas*, 2004.

Salem, A. H., 1999, “Unrelated Parallel Machine Scheduling with Sequence Dependent Setup Times and Machine Eligibility Restrictions for Minimizing Makespan”, *Ph.D. Thesis, Central Florida University*, 1999.

Sansarcı, E., 2007, “A Heuristic To Minimize Total Tardiness on Parallel Machines: An Aggregate Planning Approach”, *M.S. Thesis, Bogazici University*, 2007.

Santillan, E. D., 2002, “Analysis of parallel machines with splitting jobs and sequence dependent set up times, *Ph.D. Thesis, Texas A & M University*, 2002.

Senniappan K., 2001, “Parallel Machine Scheduling with Load Balancing and Sequence Dependent Setups”, *Thesis, Wichita State University*, 2001.

Sheen, G. J., L. W. Liao and C.F. Lin, 2006 , “Optimal Parallel Machines Scheduling with Machine Availability and Eligibility Constraints”, *Int J Adv Manuf Technol*, 2006.

Omar, Shim S. O., Y. D. Kim, 2006, “A Branch and Bound Algorithm for an Identical Parallel Machine Scheduling Problem with a Job Splitting Property”, *Computers and Operations Research*, Vol. 35, pp. 863 - 875, 2006.

Sung, S. C. and M. Vlach, 2005, “Maximizing Weighted Number of Just-In-Time Jobs on Unrelated Parallel Machines”, *Journal of Scheduling*, Vol. 8, pp. 453-460, 2005.

Tahar D. N., F. Yalaoui, C. Chu and L. Amode, 2006 , “A Linear Programming Approach for Identical Parallel Machine Scheduling with Job Splitting and Sequence-Dependent Setup Times”, *Int. J. Production Economics*, Vol. 99, pp. 63-73, 2006.

Webster, S. T., 1997, “The Complexity of Scheduling Job Families About a Common

Due Date", *Operations Research Letters*, Vol. 20, pp. 65-74, 1997.

Weng, M. X., J. Lu and H. Ren, 2001 , "Unrelated Parallel Machine Scheduling with Setup Consideration and a Total Weighted Completion Time Objective", *Int. J. Production Economics*, Vol. 70, pp. 215-226, 2001.

Yalaoui F., C. Chu, 2002, "Parallel Machine Scheduling to Minimize Total Tardiness", *Int. J. Production Economics*, Vol. 76, pp. 265-279, 2002.