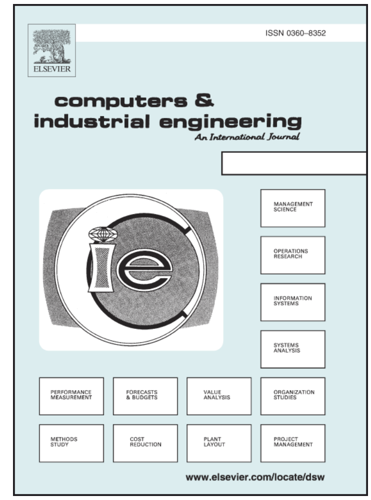
Accepted Manuscript



Robust scheduling of parallel machines considering total flow time

S. Alimoradi, M. Hematian, G. Moslehi

PII: S0360-8352(15)00504-5
DOI: http://dx.doi.org/10.1016/j.cie.2015.12.024
Reference: CAIE 4218

To appear in: *Computers & Industrial Engineering*

Received Date: 24 July 2014
Revised Date: 17 September 2015
Accepted Date: 25 December 2015

Please cite this article as: Alimoradi, S., Hematian, M., Moslehi, G., Robust scheduling of parallel machines considering total flow time, *Computers & Industrial Engineering* (2016), doi: http://dx.doi.org/10.1016/j.cie.2015.12.024

Title of the manuscript:

Robust scheduling of parallel machines considering total flow time

S. Alimoradi[1], M. Hematian[2], G. Moslehi[21]

First author: Soroush Alimoradi, Assistant Professor

Department of Mathematical Sciences, Isfahan University of Technology, 84156-83111, Isfahan, Iran, salimora@cc.iut.ac.ir

Biography:

- Bsc: Department of Statistics, Shiraz University, Iran, 1985 .
- Msc: Department of Statistics, Shiraz University, Iran, 1988 .
- PhD: Department of Statistics, Carleton University, 1995.

Second author: Milad Hematian, Master of Industrial Engineering

Department of Industrial and Systems Engineering, Isfahan University of Technology, 84156-83111, Isfahan, Iran, milad.hemmatian@in.iut.ac.ir

Third and Correspond author: Ghasem Moslehi, Professor
Department of Industrial and Systems Engineering, Isfahan University of Technology, 84156-83111, Isfahan, Iran, moslehi@cc.iut.ac.ir

Biography: I joined the department of industrial Engineering in 1989. My interesting research area is sequencing and scheduling, operations research and engineering economy.

[1] **Corresponding Author's Information:** Department of Industrial and Systems Engineering, Isfahan University of Technology, 84156-83111, Isfahan, Iran
Tel: +98 0311 3915522
Fax: +98 0311 3915526
Email: moslehi@cc.iut.ac.ir

Robust scheduling of parallel machines considering total flow time

Abstract

An effective approach to structuring uncertainty and making decisions under uncertain conditions is the robustness approach. The robust scheduling approach tries to create a schedule that minimizes the effect of disturbances caused by uncertainty during the operation on the objective function of initial scheduling. The present research was an attempt to study the problem of maximizing the probability that the total flow time does not exceed a predetermined limit in identical parallel machines while the processing time of each job is stochastic. In order to find an optimal solution to this problem, several theorems were proposed. The proposed theorems considerably reduced the search space and resulted in a branch and bound method to the problem with a specific branching scheme. Moreover, development of theorems to determine the dominance set along with introduction of dominance rules, an upper bound, and a lower bound helped find optimal solutions to the problems of 45 jobs and 5 machines. In addition, it was found that the method proposed in this paper for several machines is more effective than the methods developed for a single machine.

Keywords: Uncertainty; β-Robust Scheduling; Identical Parallel Machines; Total Flow Time; Branch and Bound.

1. Introduction

In the past half century, numerous studies have been conducted on scheduling problems. The majority of studies on scheduling problems are limited to deterministic scheduling. In deterministic problems, all job characteristics are specified with certainty in advance, with some parameters (e.g., processing time) known from the beginning and remaining unchanged. However, in the real world, because of probabilistic and uncertain phenomena such as machine breakdown, elongated processing time, sick leaves, delays in delivery of materials, and new jobs emerging at the time of implementation, the scheduling methods for solving certain scheduling problems lose their efficiency and even become infeasible.

The processing time of each job depends on the operation conditions, especially the workforce, equipment, and raw materials. Hence, processing times are uncertain regardless of random factors that may cause disturbance to the processes.

The robustness approach is one of the newest approaches to deal with the uncertainty of scheduling problems. It allows for the initial scheduling to be set in such a way that changes in data during the implementation of the schedule will have a minimal impact on the primary scheduling. Among robust scheduling approaches, the β-robust is regarded most suitable for the purpose.

Under certain conditions, total flow time is among the most important criteria that are always discussed (Li & Yang, 2009). Since this criterion is more interesting under uncertain conditions when using the robust scheduling approach, it can be used as the objective function for this problem. A decrease in this criterion means a reduction in the inventory of raw materials or work-in-process and thus lessening the order lead time (Webster, 1993). On the other hand, in the real world, it seems logical to use several machines in the manufacturing sector. In most production and assembly lines, several machines are often used in parallel, especially in bottlenecks, to prevent line halts (Cheng & Sin, 1990). In the service sector, parallel computing is yet another such example (Cheng & Sin, 1990). The same method is also used to schedule ship loadings and to assign patients to medical groups (Moslehi & Mahnam, 2010).

In this paper, we consider the problem of maximizing the probability that the total flow time will not exceed a predetermined limit in identical parallel machines while also having probabilistic processing time.

This paper is organized as follows: In section 2, a literature review of scheduling problems under uncertain conditions is presented; section 3 provides the problem formulation and symbols associated with β-robust scheduling. Theorems and lemmas developed for this problem are presented in section 4; section 5 describes the branch and bound method as well as definitions of upper bound, lower bound, dominance rules and branching scheme. Computational results are reported in section 6 and finally the conclusions and suggestions are included in section 7.

2. Literature Review

A careful scrutiny of the articles published on robust scheduling revealed that relevant studies can be classified by their assessment measures. Accordingly, five categories were distinguished by the following five measures: robustness measures based on the realized performance, robustness measures based on regret, surrogate measures, stability measures, and the β-robust approach.

In the following part, studies on robust scheduling, which have used robustness measure on the basis of the realized performance or regret, are introduced. As can be understood from the literature, it was Daniels and Kouvelis (1995) who introduced the robust optimization approach to scheduling problems for the first time. They assumed uncertain processing time for jobs in a single machine environment and developed a robust scheduling solution to single machine problem using the total flow time as the objective function. They used scenario-based planning to create uncertain processing times. They also used the worst case minimization as the robustness measure. They developed a branch and bound algorithm and two heuristic algorithms with $O(n \log n)$ time complexity to achieve robust schedule.

Kouvelis and Yu (1997) defined three robustness measures for the scheduling problem and developed a mathematical model for the problem of single machine scheduling and a two-machine flowshop.

In a single machine environment and total completion time as their objective function, Yang and Yu (2002) used the scenario-based planning with uncertain processing time to achieve robust scheduling. They also proposed a dynamic programming algorithm of $O(2^n)$ complexity and two heuristic algorithms, one with a complexity of $O(n^4|S|)$ and the other with polynomial complexity of $O(n\log n+|S|n)$, where $|S|$ refers to the number of scenarios.

Lu et al. (2012) proposed a robust scheduling model for the single-machine scheduling problem with the total flow time as its objective function. In that problem, the processing time was uncertain and the sequence-dependent family setup time was considered. They solved the problem using the robust-constrained shortest path method and developed a simulated annealing algorithm. They also studied the effect of robustness on the optimal value. Lu et al. (2014) also considered the scenario-based planning for the same problem. They used the worst case scenario to solve the problem. They also developed a mixed integer mathematical programming model for the problem and introduced two heuristic algorithms to find a more efficient solution to the problem. Among many other studies on this subject, the studies by Kasperski (2005) and Montemanni (2007) can be mentioned.

A large number of studies on robust scheduling are based on surrogate measures (Goren and Sabuncuoglu (2008); Goren and Sabuncuoglu (2010); Kasperski. et al. (2012); Wu et al. (1999)). Xia et al. (2008) studied the due date assignment problem and a sequence of jobs in a single-machine environment based on surrogate measures. In this problem, the processing time was uncertain and its mean and variance values were known. A heuristic method for achievement of a sequence of jobs and due date assignment was also developed to minimize the linear combination of three penalties including penalty on job earliness, penalty on job tardiness and penalty associated with long due date assignment. The result indicated that when the mean and variance of processing time were uncorrelated or negatively correlated, the system performance was enhanced and the scheduling was more robust.

The β-robust approach is one of the various approaches of robust scheduling that has recently attracted increasing attention. Daniels and Carrillo (1997) were the first researchers to introduce the notion of β-robust scheduling for a single machine scheduling problem with total flow time as its objective function and uncertain processing times. They also used the shortest processing time rule with the complexity of $O(n\log n)$ in their research. Results of the study revealed that the β-robust scheduling problem was NP-hard. They tried to find a sequence to maximize the probability so that the total flow time would not exceed a predetermined limit. They managed to develop a branch and bound algorithm capable of solving problems up to 20 jobs.

Wu. et al. (2009) proposed a β-robust scheduling model with constraint programming for the problem of single machine scheduling that had the total flow time as its objective function and assumed an uncertain processing time with a normal distribution. Three constraint programming models were developed for this problem. The method developed by these researchers was capable of solving problems up to 20 jobs with respect to time limit of 12 minutes.

Recently, Ranjbar et al. (2012) have used the β-robust scheduling approach to solve the scheduling problem of identical parallel machines with the maximum completion time as its objective function and uncertain processing times. The present study was an attempt to find a robust scheduling that maximizes the customer service level. That is to say, the scheduling should maximize the probability so that the maximum completion time does not exceed the due date. Two branch and bound algorithms that differ mainly in their branching schemes were developed for finding an optimal solution. The proposed algorithm managed to solve problems of 20 jobs and 3, 4, and 5 machines.

3. Problem Formulation

In this section, the probability distribution function of a specific scheduling objective function, known as the β-robust scheduling problem (β-RSP), is maximized (Daniels & Carrillo, 1997). This study tries to solve the problem of maximizing the probability that the total flow time will not exceed a predetermined limit (i.e., customer service level) with identical parallel machines. The three-field notation method introduced by Graham et al. (1979) was used in this study (Jarboui et al., 2009). Therefore, the problem was written as $P_m \mid \beta - RSP \mid TFT \leq \delta$. The objective here is to find the sequence with the highest probability without having the total flow time exceed a predetermined limit.

Further, it is assumed that all the jobs are available and can be started at time zero, and the setup time for each job on each machine is independent of the sequence of jobs. At any one moment, a job can be processed only on one machine, and similarly, each machine can process only one job at a time. Moreover, the processing times for jobs are independent and uncertain. The processing time of job *i* has a normal distribution function with a mean of μ_i and variance of σ_i^2. Symbols used in this paper are as follows:

n : The total number of jobs;

m : Number of machines; $m < n$

n_j : Number of jobs on machine *j* $j = 1, 2, ..., m;$

n'_j : Number of jobs in the partial sequence on machine *j*. $j = 1, 2, ..., m;$

M_j : Machine number *j* $j = 1, 2, ..., m;$

$J_{[i]j}$: The *i*th job on machine *j* $j = 1, 2, ..., m,$ $i = 1, 2, ..., n_j$

$P_{[i]j}$: Processing time of the *i*th job on machine *j* $j = 1, 2, ..., m,$ $i = 1, 2, ..., n_j$

TFT_j : Total flow time of jobs on machine j $\quad j = 1, 2, ..., m;$

TFT : Total flow time of jobs on all machines

$\mu_{[i]j}$: The mean processing time of the ith job on machine j $\quad j = 1, 2, ..., m,\ i = 1, 2, ..., n_j$ and

$\sigma^2_{[i]j}$: Variance of the processing time of the ith job on machine j $\quad j = 1, 2, ..., m,$ $i = 1, 2, ..., n_j$;

The total flow time for all jobs on all machines is obtained through relation 1.

$$TFT = \sum_{j=1}^{m} \sum_{i=1}^{n_j} (n_j - i + 1) P_{[i]j} \tag{1}$$

Wu. et al. (2009) proved that the TFT for one machine, which is obtained through relation 1, has a normal distribution. In the following, it is indicated that the total flow time on m machines has a normal distribution.

Theorem 1: In problem $P_m \mid \beta - RSP \mid TFT \le \delta$, the total flow time has a normal distribution.

As TFT is distributed normally in a single-machine mode, the TFT distribution of any machine among the m machines is also normal. Consequently, total TFT, obtained from summation of every machine's TFT is also normally distributed, which is shown in relation 2.

$$TFT \sim N\left(\sum_{j=1}^{m} \sum_{i=1}^{n_j} (n_j - i + 1) \mu_{[i]j}, \sum_{j=1}^{m} \sum_{i=1}^{n_j} (n_j - i + 1)^2 \sigma^2_{[i]j}\right) \tag{2}$$

Moreover, the probability that the total flow time on m machines for a sequence does not exceed δ is obtained by relation 3.

$$p_r(TFT \le \delta) = \phi\left(\frac{\delta - \sum_{j=1}^{m} \sum_{i=1}^{n_j} (n_j - i + 1) \mu_{[i]j}}{\sqrt{\sum_{j=1}^{m} \sum_{i=1}^{n_j} (n_j - i + 1)^2 \sigma^2_{[i]j}}}\right) \tag{3}$$

Where, ϕ is the cumulative distribution function of the standard normal distribution.

4. Problem Properties

In this section, at first the problem of minimizing the expected total flow time on m machines with probabilistic processing times (i.e., the $P_m \| TFT$ problem) is discussed. Next, the problem of maximizing the probability that the total flow time on identical parallel machines does not exceed a predetermined limit ($P_m \mid \beta - RSP \mid TFT \le \delta$) is discussed.

Theorem 2: In an optimal solution to $P_m \| TFT$ with stochastic processing time, the number of jobs assigned to each machine follows relation 4.

$$|n_j - n_k| \le 1 \quad j,k = 1,2,...,m \tag{4}$$

Since in this theorem only the mean processing time is of interest, it follows the rules of a deterministic case and is therefore clearly evident (Baker & Trietsch, 2009).

Theorem 3: In an optimal solution to $P_m \,|\, \beta - RSP \,|\, TFT \le \delta$, the number of jobs assigned to each machine follows relation 5.

$$|n_j - n_k| \le 1 \quad j,k = 1,2,...,m \tag{5}$$

Proof: Figure 1 shows the assignment of jobs to two arbitrary machines *l* and *k* in sequence S. Let us assume that in this sequence, the number of jobs assigned to machines *l* and *k* do not follow relation 5, which means the difference between the numbers of jobs assigned to the two machines is more than one. Relation 6 assumes that the number of jobs assigned to machine *l* is more than that to machine *k*.

$$n_l - n_k \ge 2 \tag{6}$$

Figure 1. About here

As seen in Figure 2, in the S' , the first job is omitted from machine *l* and is added to the beginning of the jobs for machine *k* so that difference between the numbers of jobs assigned to both machines will decrease.

Figure 2. About here

It is assumed that the numbers of jobs assigned to other machines are identical in the S and S' sequences. This indicates that the variance of total flow time on *m* machine in sequence S' is less than the variance of total flow time in sequence S. According to theorem 2, relation 7 holds true.

$$mean(TFT(S)) - mean(TFT(S')) > 0 \tag{7}$$

The variance of total flow time on *m* machine in sequences S and S' is obtained from relations 8 and 9, respectively.

$$\begin{aligned}\mathrm{var}(TFT(S)) = \textstyle\sum_{j \ne l,k} \mathrm{var}\left(TFT(S)_j\right) &+ \left((n_l)^2 \sigma^2_{[1]l} + (n_l - 1)^2 \sigma^2_{[2]l} + ... + (2)^2 \sigma^2_{[n_l - 1]l} + \sigma^2_{[n_l]l}\right) \\ &+ \left((n_k)^2 \sigma^2_{[1]k} + (n_k - 1)^2 \sigma^2_{[2]k} + ... + (2)^2 \sigma^2_{[n_k - 1]k} + \sigma^2_{[n_k]k}\right)\end{aligned} \tag{8}$$

$$\begin{aligned}\mathrm{var}(TFT(S')) = \textstyle\sum_{j \ne l,k} \mathrm{var}\left(TFT(S')_j\right) &+ \left((n_l - 1)^2 \sigma^2_{[2]l} + (n_l - 2)^2 \sigma^2_{[3]l} + ... + (2)^2 \sigma^2_{[n_l - 1]l} + \sigma^2_{[n_l]l}\right) \\ &+ \left((n_k + 1)^2 \sigma^2_{[1]k} + (n_k)^2 \sigma^2_{[2]k} + ... + (2)^2 \sigma^2_{[n_k - 1]k} + \sigma^2_{[n_k]k}\right)\end{aligned} \tag{9}$$

Relation 10 is obtained through the substitutions and the fact that the first term of the right sides of relations 8 and 9 are identical.

$$\mathrm{var}(TFT\,(S)) - \mathrm{var}(TFT\,(S')) = ((n_l)^2 \sigma^2_{[1]l} - (n_k + 1)^2 \sigma^2_{[1]l} \tag{10}$$

As relation 10 is always positive, the total variance of flow time on *m* machine in sequence S' is less than that in sequence S.

Since the mean total flow time and its variance in sequence S' are less than those in sequence S, the probability that the total flow time will not exceed δ is always higher in sequence S' than in sequence S.

$$\phi\left(\frac{\delta - mean(TFT(S'))}{\sqrt{\mathrm{var}(TFT(S'))}}\right) - \phi\left(\frac{\delta - mean(TFT(S))}{\sqrt{\mathrm{var}(TFT(S))}}\right) > 0 \tag{11}$$

Therefore, with a one-unit decrease in the difference between the numbers of jobs assigned to machines *l* and *k* better results can be obtained. This process continues until the difference between the jobs of machines *l* and *k* reaches one and follows relation 5. Hence, the theorem is proved as this process continues for machines *l* and *k* and also for any machine whose number of jobs does not follow relation 5.

□

If the dominant set is defined as a set that certainly contains an optimal solution, the following results can be put forward about the set.

Corollary 1: The dominance set for $P_m\,|\,\beta - RSP\,|\,TFT \le \delta$ includes all the sequences in which no machine is idle.

Therefore, each machine's program can be executed when a sequence exists.

Corollary 2: The dominance set includes the sequences in which the minimum and maximum number of jobs on each machine are $\left\lfloor n/m \right\rfloor$ and $\left\lceil n/m \right\rceil$, respectively.

It should be mentioned that solutions without either condition of results 1 and 2 are not optimal.

In the following, it will be demonstrated that if two jobs with identical positions, as counted from the end of the respective sequence, are exchanged, the sum of mean total flow time and its variance will remain unchanged.

Lemma 1: If in problem $P_m\,|\,\beta - RSP\,|\,TFT \le \delta$ the two jobs named $J_{[k]j}$ and $J_{[l]i}$ are exchanged so that relation 12 is followed, no changes occur in the probability that the total flow time does not exceed δ in a sequence.

$$n_i - l = n_j - k \tag{12}$$

According to relation 2, the position of each job acts as a coefficient and affects the mean, variance, and probability values in the objective function of the total flow time. Therefore, jobs in identical positions hold equal significance.

Corollary 3 (Dominance Rule): The dominance set only includes sequences in which the job indexes for similar positions at the end of each machine are in an ascending order.

According to Lemma 1, for every sequence S in Figure 3, jobs with identical positions are of equal importance. Hence, these jobs can be classified into the same set. According to Figure 3, jobs $J_{[4]1}$, $J_{[4]2}$ and $J_{[3]3}$ are put in Set 1 because their positions are equal to 1. Similarly, jobs $J_{[3]1}$, $J_{[3]2}$ and $J_{[2]3}$ are put in Set 2 as their positions are equal to 2. This process continues until all the jobs are classified into their appropriate sets. Meanwhile, each set includes different and numerous permutations of jobs that are of equal importance. Therefore, it is possible to only consider one that shows an ascending order of the jobs' index in each set. As a result, a large number of jobs' orders are eliminated and the ascending order of the jobs' index forms the dominance set.

Figure 3. About here

Here is an example to further elaborate on Corollary 3. Let us assume there are 7 jobs and 3 machines. A sequence called S is shown on the left-hand side of Figure 4. Set 1 includes jobs 5, 2 and 7. The possible permutations of these three jobs are identical when they are at the end of their machine's sequence. According to Lemma 1, only one permutation which shows the jobs in the set in an ascending order is selected. Therefore, in set 1, job 2 is assigned to machine 1, job 5 is assigned to machine 2, and job 7 is assigned to machine 3. This procedure is repeated for all the sets. On the right-hand side of Figure 4, the dominance set of this sequence is shown, in which jobs in each set are arranged in an ascending order.

Figure 4. About here

It should be noted that there are optimal solutions which do not apply to result 3.

5. Branch and Bound Procedure

Each node in the proposed branch and bound algorithm represents a partial sequence of arranged jobs, which sits at the beginning of the complete sequence. The sub-branch of each node is called an offspring and the node is known as its parent. Prior to the search process, a lower bound is calculated by a heuristic method. The tree search in B&B is carried out using the depth first method. During the tree search process, if the upper bound of a node is lower than the lower bound or at least one dominance rule applies to it, the node is fathomed.

Here, at first the lower bound, upper bounds, and dominance rules for the $P_m \,|\, \beta - RSP \,|TFT \le \delta$ problem are described and then the branching procedure which is specific to this problem is explained.

5.1. Lower Bound

In this section, a lower bound is developed for a fixed node of the branch and bound tree based on the characteristics of the $P_m \mid \beta - RSP \mid TFT \le \delta$ problem. The process is described below. This lower bound is used to find a good, feasible solution.

In order to obtain the aforementioned lower bound, the Shortest Expected Processing Time is employed (Baker & Trietsch, 2009). At each stage, the job with the shortest mean processing time is assigned to the machine with the lowest mean flow time. According to Lemma 1, jobs in identical positions are put in one set. Thereafter, each job in each set is temporarily replaced with another job in another set and if the probability value improves, the exchange is made permanent. The details of the proposed algorithm are presented in the appendix

5.2. Upper Bounds

In this section, three upper bounds are developed for each node of the proposed branch and bound algorithm based on the characteristics of problem $P_m \mid \beta - RSP \mid TFT \le \delta$. The resulting bounds are described as follows.

Assume that π denotes a partial sequence of the assigned jobs so that the current step includes jobs $J_{[1]j}$ and $J_{[n'_j]j}$ on machine *j*. The set of the unassigned jobs is also shown by π'. It will be proved that partial sequence π which is shown by π_{U1} is an upper bound for $P_m \mid \beta - RSP \mid TFT \le \delta$.

Theorem 4: The partial sequence π_{U1} is an upper bound for the $P_m \mid \beta - RSP \mid TFT \le \delta$ problem.

According to the nature of the problem, adding a job to a partial sequence leads to an increase in the mean total flow time and its variance which in turn leads to a decrease in the probability. □

By adding each new job to the partial sequence π_{U1}, probability declines while the mean total flow time and its variance increase.

In order to calculate the second upper bound, a virtual job named d2 is assumed. The mean processing time for the job is μ' which is equal to the minimum mean processing time for the unassigned jobs π' and its variance is σ'^2 which is equal to the minimum variance of the unassigned jobs π'.

Each machine is assigned a number of d2 jobs so that the number of jobs on each machine equals $\left\lfloor {n}/{m} \right\rfloor$ after assigning the d2 jobs. As a result, a sequence of jobs is created which is shown by π_{U2}.

Theorem 5: The partial sequence π_{U2} is an upper bound for $P_m \mid \beta - RSP \mid TFT \le \delta$ problem.

Proof: According to Corollary 2, the number of jobs assigned to each machine in an optimal solution is at least equal to $\lfloor n/m \rfloor$. Hence, a number of d2 jobs are assigned to the partial sequence π so that the number of jobs on each machine equals $\lfloor n/m \rfloor$. In this case, machine j is assigned n''_j jobs.

The mean total flow time in this partial sequence is obtained through relation 13.

$$mean(TFT(\pi_{U2})) = mean(TFT(\pi_{U1})) + \sum_{j=1}^{m}\sum_{i=1}^{n'_j} n''_j \mu_{[i]j} + \sum_{j=1}^{m}\sum_{i=1}^{n''_j}(n''_j - i + 1)\mu'_{[i]j} \tag{13}$$

In relation 13, $\mu'_{[i]j}$ shows the mean processing time for job d2 on the ith position of machine j. Relation 13 can easily be transformed into relation 14.

$$mean(TFT(\pi_{U2})) = mean(TFT(\pi_{U1})) + \sum_{j=1}^{m}\sum_{i=1}^{n'_j} n''_j \mu_{[i]j} + \sum_{j=1}^{m}\left(\frac{(n''_j)(n''_j + 1)}{2}\right)\mu' \tag{14}$$

As can be seen, the first term on the right side of relation 14 is for the π_{U1} partial sequence and therefore is constant. The second part of the right side of relation 14 shows the sum of the product of position coefficients of virtual jobs at the end of each machine by the mean processing time of jobs. Since the coefficient value is the same for all the sequences, and the mean total processing time for all the jobs is equal to the minimum mean of total processing time in set π', the mean total flow time in the partial sequence π_{U2}, (i.e., $mean(TFT(\pi_{U2}))$), is at its lowest. Similarly, the variance of total flow time in the partial sequence π_{U2} is obtained through relation 15. Since the variance of total processing time is equal to the minimum variance of total processing time in set π', then $\mathrm{var}(TFT(\pi_{U2}))$ is also at its lowest.

$$\mathrm{var}(TFT(\pi_{U2})) = \mathrm{var}(TFT(\pi_{U1})) + \Sigma_{j=1}^{m}\Sigma_{i=1}^{n'_j}(2n'_j n''_j + n''^2_j)\sigma'^2_{[i]j} + \Sigma_{j=1}^{m}\Sigma_{i=1}^{n''_j}(n''_j - i + 1)\sigma'^2_{[i]j} \tag{15}$$

In relation 15, $\sigma'^2_{[i]j}$ shows the mean processing time for d2 which is assigned to position ith of machine j. Relation 16 can be obtained easily from relation 15.

$$\mathrm{var}(TFT(\pi_{U2})) = \mathrm{var}(TFT(\pi_{U1})) + \Sigma_{j=1}^{m}\Sigma_{i=1}^{n'_j}(2n'_j n''_j + n''^2_j)\sigma'^2_{[i]j} + \sum_{j=1}^{m}\left(\frac{(n''_j)(n''_j + 1)}{2}\right)\sigma' \tag{16}$$

It is, therefore, concluded that the mean total flow time and its variance are at their lowest in the partial sequence π_{U2}. Therefore, the probability is at its highest and consequently the partial sequence π_{U2} is considered an upper bound for $P_m \mid \beta - RSP \mid TFT \le \delta$ problem. □

To calculate the third upper bound, at first the jobs in set π', which amount to n'', are arranged in an ascending order and form a new set which is called the π'_μ set. Moreover, jobs in set π' are once again arranged in an ascending order based on the variance of processing time and the resulting set is called π'_{σ^2}. Hence, relations 17 and 18 apply to set π'_{μ} and set π'_{σ^2}, respectively.

$$\mu'_i \leq \mu'_j \quad \forall i < j, i, j \in \pi'_{\mu} \tag{17}$$

$$\sigma'^2_i \leq \sigma'^2_j \quad \forall i < j, i, j \in \pi'_{\sigma^2} \tag{18}$$

Now, for each member of set π', a virtual job is created. The mean processing time and variance for the first virtual job are μ'_1 and σ'^2_1, respectively. The second virtual job has the second mean processing time from set π'_{μ} and the second variance from set π'_{σ^2}. Next, virtual jobs one, two, etc. are assigned to each machine so that the number of jobs assigned to each machine equals $\left\lfloor n/m \right\rfloor$. The new sequence is called π_{U3}. In the following part, it is proved that this sequence is an upper bound for $P_m \,|\, \beta\!-\!RSP \,|\, TFT \leq \delta$.

Theorem 6: The partial sequence π_{U3} is an upper bound for $P_m \,|\, \beta\!-\!RSP \,|\, TFT \leq \delta$ problem.

Proof: Similar to theorem 5, in order to prove this theorem it is assumed that a number of n''_j jobs have to be assigned so that the number of jobs on machine *j* equals $\left\lfloor n/m \right\rfloor$. Jobs are assigned to the machines in a specific order and the new sequence is called π_{U3}. The mean total flow time in this partial sequence is obtained by relation 13. Since the jobs assigned to each machine to increase the number of jobs on that machine to $\left\lfloor n/m \right\rfloor$ are virtual, the mean total flow time is at its lowest. Similarly, the variance of total flow time, which is obtained using relation 15, is at its lowest. On the other hand, job coefficients are in descending order while the mean and variance of total flow times for jobs are in an ascending order. Hence, the result of multiplication of these two terms is minimum.

Since the mean and variance of total flow time in the partial sequence π_{U3} are at their lowest, the probability obtained from this partial sequence is at its highest. As a result, the partial sequence π_{U3} is an upper bound for $P_m \,|\, \beta\!-\!RSP \,|\, TFT \leq \delta$. □

5.3. Dominance Rules

In this section, two dominance rules used in the proposed branch and bound algorithm are explained.

Theorem 7: In an optimal solution to $P_m \,|\, \beta\!-\!RSP \,|\, TFT \leq \delta$, if $\mu_p \leq \mu_q$ and $\sigma^2_p \leq \sigma^2_q$, then there is an optimal solution where $J_p = J_{[s]l}$ and $J_q = J_{[t]k}$. In that case, similar to relation 19, the distance between job *p* and the last job on the machine of concern is higher than the distance between job q and the last job on that machine.

$$n_l - s + 1 \geq n_k - t + 1 \tag{19}$$

Proof: The theorem will be proven by contradiction. Assume an optimal sequence of $P_m \mid \beta - RSP \mid TFT \leq \delta$ includes $\mu_p \leq \mu_q$, $\sigma_p^2 \leq \sigma_q^2$ and $n_l - s + 1 < n_k - t + 1$.

According to Lemma 1, there is a job like r that follows relation 20 as $J_r = J_{[w]k}$.

$$n_l - s = n_k - w \tag{20}$$

According to relation 20 and Lemma 1, by replacing jobs p and r no change occurs in the total flow time and the probability value. Hence, these two jobs are replaced. Consequently, jobs p and q belong to machine k while job p precedes q.

For the single machine problem, Wu. et al. (2009) indicate that for the sequence with $\mu_p \leq \mu_q$ and $\sigma_p^2 \leq \sigma_q^2$ an optimal solution will be found where job p precedes job q. According to the proof presented by Wu. et al. (2009), the sequence including $n_l - s + 1 \geq n_k - t + 1$ is an optimal sequence. Therefore, the opposite assumption is rejected and the theorem is approved. □

A corollary of the former dominance rule can be a particularly effective tool. Firstly, parameter e_{ij} is defined as follows.

$$e_{ij} = \begin{cases} 1 & \mu_p \leq \mu_q, \sigma_p^2 \leq \sigma_q^2 \ \forall i \neq j \\ 0 & o.w. \end{cases}$$

Based on parameter e_{ij} for each job, it is possible to determine the earliest and latest position the job can have.

5.4. Branching Procedure

The branching procedure explained in this paper is based on assigning a set of jobs to the existing machines at each stage. Job sets are formed based on the position of jobs from the end of each machine. Jobs placed in the first position at the end of each machine are put in one set. Jobs placed in the second position at the end of each machine are put in a separate set and this process continues until all the jobs are categorized into appropriate sets. In this form of branching, Lemma 1 and Corollary 3 are used to structure the branching. According to Lemma 1, the order of jobs in a set on all machines does not influence the result. Therefore, based on Corollary 3 only one order of jobs is selected as a representative.

In this form of branching, the jobs in each set are determined at each level. With *n* jobs and *m* machines, $\left\lceil {}^{n}\!/_{m} \right\rceil$ yields the number of sets which is equal to the number of the tree levels. The result of $\left\lfloor {}^{n}\!/_{m} \right\rfloor$ is also used to determine the number of sets with equal number of jobs. According to Theorem 3, the maximum difference between the results is 1, which shows that the set of jobs sits at the beginning of the sequence at level 1.

For example, assume the scheduling problem $P_m \mid \beta - RSP \mid TFT \leq \delta$ with 5 jobs and 2 machines. According to the above description, the number of sets is equal to $\left\lceil {}^{5}\!/_{2} \right\rceil = 3$. Therefore, the branching procedure for this problem involves three levels. Moreover, the number of sets with equal jobs is $\left\lfloor {}^{5}\!/_{2} \right\rfloor = 2$. Hence, one single-job set is developed on machine 1 while two 2-job sets are developed on machines 1 and 2.

Figure 5 depicts the branching procedure in which at first the single job in set 3 is determined at level 1. Next, at the second level the jobs in set 2, which have second position at the end of the two machines, are selected such that the jobs in the set are arranged in an ascending order based on Corollary 3. Finally, the remaining 2 jobs in set 1 are assigned to the two machines in an ascending order.

Figure 5. About here

6. Computational Results

In this section, the performance and effectiveness of the proposed branch and bound algorithm are assessed. All of the computations were carried out in version 2010 of Visual Studio C++ running on a computer with the following specifications: 4GB RAM, a 2.40 GHz, Intel Core i5-2430M CPU.

6.1. Instance Problems Generation

The instance problems presented in the study were derived from the studies by Daniels and Carrillo (1997), Wu. et al. (2009), and Ranjbar et al. (2012). Processing time follows a normal distribution. The mean processing time has a uniform distribution and is randomly selected from the discrete interval $[10, 50\delta_1]$ and variance of processing time is randomly selected from the $[0, \frac{1}{9}\mu_i^2\delta_2]$ discrete interval. Parameters δ_1 and δ_2 control the variations in the average mean and variance values of the processing times. That is to say, they determine the intervals for changes in the mean and variance of processing time. Values of 0.4, 0.7 and 1 were determined for parameters δ_1 and δ_2 (Daniels & Carrillo, 1997; Wu. et al., 2009). Symbol δ, which shows the service level, was used to determine the maximum acceptable TFT via relation 21.

$$\delta = n \times \delta' \tag{21}$$

According to Wu. et al. (2009), the δ' was calculated using the mean, variance, and z_α values through relation 22. In this regard, α was assumed to be equal to 0.85, 0.95 and 0.99. In this study, it is tried to find sequences in which the average flow time is less than δ with the probabilities of 0.85, 0.95 and 0.99. The z_α values corresponding to probabilities 0.85, 0.95, and 0.99 are α = 1.04, 1.65, and 2.33, respectively.

$$\delta' = \frac{mean(TFT_{SEPT})}{n} + z_\alpha \times \sqrt{\frac{\mathrm{var}(TFT_{SEPT})}{n^2}} \tag{22}$$

Hence, based on the values assigned to δ_1 , δ_2 and α, a total of 27 (3*3*3) sets of problems are obtained and named as G1 to G27.

In this study, the numbers of jobs are 10, 15, 20, 25, 30, 35, 40 and 45, and for each job 3, 4 and 5 machines are available. From each problem set, 10 sample problems were generated and solved. Hence, 6480 (27*8*3*10) instance problems were totally generated and solved.

6.2. Experimental results

The results of solving the instance problems are presented in Table 1. In this table, the column headed "number of optimal solutions" displays the number of instances for which the B&B method could obtain an optimal solution from 10 samples. Results indicated that out of the 27 sets, 11 sets were fully solved using the B&B method. Moreover, sets G21, G24 and G27 had the minimum number of solved instances.

Table 1. About here

As can be seen, the first 11 sets could find optimal solutions to all of their 240 instances. These 11 sets are called the *Easy* sets. *G2*, *G5*, and *G8* are presented as *Easy sets*. The set of *moderate* problems includes 10 problem sets, most of which have α=0.95. In the *moderate* set at most 5 samples could not be solved optimally. The time required for solving the instance problems in these sets varies between 32 to 120 seconds. The *Hard* sets are those problem sets that have α=0.99. In these sets, a fewer number of instances found optimal solutions and they required a higher solving time on average.

A 3600-second constraint was considered for solving the instance problems, and those that were not solved during the period lacked a mean value. Comparison of the mean time required for solving the samples in Table 1 indicates that G21 had the highest solving time average while 20 sets also required less than 2 minutes to be solved.

The higher the percentage of fathomed nodes, the better the performance of the B&B method. In other words, the problem has been solved with a lower proportion of the problem tree explored.

In order to assess the efficiency of the sequence obtained from the β-robust method, a comparison was made between the probabilities obtained from the SEPT and β-robust schedules (Wu. et al., 2009). In order to compare the two sequences, the aforementioned researchers used relation 23, which compares the risks of two sequences. In this relation, R denotes the risk while TFT_{SEPT} and TFT_{β} show the total flow time in the SEPT sequence and the β-robust sequence, respectively. Risk of a sequence indicates that the TFT obtained from the sequence exceeds the predetermined δ.

$$R = \frac{(1-p_r(TFT_{SEPT} \leq \delta) - (1-p_r(TFT_{\beta} \leq \delta)}{(1-p_r(TFT_{\beta} \leq \delta)} \times 100 \tag{23}$$

For example, if the result of relation 23 is equal to 5, it means that the likelihood that the TFT exceeds δ is 5% more with the SEPT sequence than in β-robust sequence. In order to demonstrate the performance of the β-robust sequence, two columns headed "average risk" and "maximum risk" are reported in Table 2. Each row in the "average risk" column shows the average risk (%) in some of the instances from the set of 10 instance problems that had optimal solutions. Moreover, each row in column "maximum risk" shows the maximum risk (%) in some of the instances from the set of 10 instance problems for which optimal solutions have been found.

Table 2. About here

Relation 23 can be simplified and turned into relation 24.

$$R = \frac{(1-\alpha)}{(1-p_r(TFT_{\beta} \leq \delta)} - 1 \tag{24}$$

According to the computational results presented in Table 2 and relation 24, when α is constant, an increase in the number of jobs *n* leads to a decrease in the risk. In other words, the changes in variance in the β-robust sequence increase more drastically with a larger *n* as compared to changes in the mean flow time. Hence, $(1-p_r(TFT_{\beta} \leq \delta)$ value becomes larger in sets with a larger *n*, and leads to a decline in relation 24. Therefore, the risk of sets with larger *n* values is less than the risk of sets with smaller *n* values.

According to the computational results reported in Table 1, changes in α do not considerably affect the percentage of nodes fathomed using dominance rules 1 and 2 and the upper bound. It is concluded that the dominance rule 2 is always executed better than dominance rule 1 and the upper bound. Table 1 suggests that the most difficult problems are in sets G21, G24 and G27. On average, instance problems in these sets require a higher solving time and yield fewer optimal solutions. The reason might be that the interval available for the maximum probability is lengthier given that α=0.99 and $\delta_2 = 1$. As a consequence, a larger search space needs to be explored and, due to the time constraint, either the time required for solving should increase or the instance will not be solved.

As seen in Table 1, for each *m*, with an increase in the number of jobs the average instance problems' solving time increases. The difference between the times required for solving instance problems for *m* machines up to 35 jobs is negligible but when the number of jobs exceeds 35, the difference escalates and intensifies. The reason for the increased problem solving time is that the number of jobs assigned to each machine is higher in m=3 than in m=4 and m=5. This happens because with an increase in the number of jobs assigned to each machine more possible states have to be examined.

Figure 6 shows the time required for solving instance problems with different values of the α parameter. Values of (0.85, 0.95 and 0.99) and (0.4, 0.7 and 1) are considered for parameters α and δ_1, respectively. As can be seen, the instance solving time grows with an increase in α.

Figure 6. About here

With an increase in α, the instance problem solving time is also expected to rise because the interval for the maximum probability will be extended and thus a larger search space will need to be examined, leading to a longer solving time. Moreover, this diagram also shows the effect of different values of δ_1 on the instance problem solving time for each α. According to Figure 6, with α =0.85 and α =0.99 the instance solving time is longer for $\delta_1 = 0.4$ as compared to the other two values. Moreover, with α =0.95 the instance solving time is less than the other two values.

As mentioned before, the B&B algorithm was developed for the problems including parallel machines. However, in order to examine its efficiency and capability, this algorithm was solved for a single-machine problem and the results were compared with the results from the latest research by Wu. et al. (2009). Instance problems were generated using a similar method in both studies and a time constraint of 720 seconds was determined for solving the instances. In the study by Wu. et al. (2009), three primary, secondary and combined models were developed for solving the instance problems, with the combined model demonstrating the best performance. Table 3 presents the comparison between the computational results of the combined model proposed by Wu. et al. (2009) and the B&B method.

As shown in Table 3, the algorithm proposed by Wu. et al. (2009) is capable of solving problems of up to 20 jobs within 720 seconds, but the B&B algorithm solves problems up to 30 jobs. Moreover, the mean solving time required by B&B for the jobs up to 20 was a lot lower than the mean solving time consumed by the algorithm proposed by Wu. et al. (2009).

Table 3. About here

7. Conclusions and Suggestions

In this paper, we studied the problem of maximizing the probability that the total flow time would not exceed a predetermined limit in identical parallel machines when the processing time is stochastic (i.e. $P_m \mid \beta{-}RSP \mid TFT \le \delta$). A branch and bound B&B algorithm was proposed to solve the problem. Moreover, a lower bound, three upper bounds, and two dominance rules were also developed for the branch and bound algorithm. Computational results revealed the satisfactory performance of the B&B algorithm and indicated that it is capable of solving problems up to 45 jobs and 3, 4 and 5 machines. The B&B algorithm was also solved for the single-machine problem and the results were compared with those of the latest study on this problem. The comparison indicated that the proposed algorithm (B&B) is more capable than the combined model developed by Wu. et al. (2009).

For future work, it is recommended to solve this problem in other environments such as the flow shop environment, job shop environment, etc. In this research, probability was maximized based on the total flow time. Moreover, other measures such as lateness, earliness, and tardiness can also be put to test. In order to add to the complexity of the problem, it is possible to define precedence constraints, setup time, and so on. Finally, it is recommended to examine distribution functions other than the normal distribution function for this problem.

Appendix

Steps of the lower bound algorithm are as follows:

Step 0: Set *n* as the number of jobs and *m* as the number of machines. Also, set *t* to $\lceil n/m \rceil$. Call S_t the set of jobs with the position *t* from the end of the each machine. Moreover, consider $N_t = |S_t|$.

Step 1: Arrange jobs based on the ascending order of mean processing time.

Step 2: Assign jobs to the machine with the lowest total mean processing time and call the resulting sequence π.

Step 3: For the sequence obtained from the second step, calculate the mean and variance of total flow time on machines as well as the probability that the total flow time becomes less than δ. Assign $p_r(\pi)$ to the resulting probability.

Step 4: Set $i = 1, j = 1, l = \lceil n/m \rceil$ and $k = l - 1$.

Step 5: The sequence π' is generated from the temporary exchange of the *i*th job in set J_l with the *j*th job in set J_k. Calculate the probability of the sequence π' which is shown by $p_r(\pi')$. If $p_r(\pi') < p_r(\pi)$, go to Step 6; otherwise, go to Step 7.

Step 6: Undo the temporary replacement of the two jobs and go to Step 8.

Step 7: Make the temporary replacement permanent, set π = π́ and go to Step 8.

Step 8: Set $j = j + 1$. If $j \le N_k$, then go to Step 5; otherwise, go to Step 9.

Step 9: Set $k = k - 1$. If $k \ge 1$, then set $j = 1$ and go to Step 5; otherwise, go to Step 10.

Step 10: Set $i = i + 1$. If $i \le N_l$, then set $j = 1$ and $k = l - 1$ and go to Step 5; otherwise, go to Step 11.

Step 11: Set $l = l - 1$. If $l > 1$, then go to Step 4; otherwise, finish.

The computational complexity of the proposed algorithm is equal to $O(n^2/m + n\log n)$. Hence, if $m < \frac{n}{\log n}$, then the computational complexity is equal to n^2/m; otherwise the computational complexity is equal to $n\log n$.

References

Baker, K.R. & Trietsch, D. (2009). Principles of Sequencing and Scheduling John Wiley & Sons.

Cheng, T.C.E. & Sin, C.C.S. (1990). A state-of-the-art review ofparallel-machine scheduling research European Journal of Operational Research, 47, 271-292.

Daniels, R.L. & Carrillo, J.E. (1997). β-Robust scheduling for single-machine systems with uncertain processing times. IIE Transactions, 29, 977-985.

Daniels, R.L. & Kouvelis, P. (1995). Robust scheduling to hedge against processing time uncertainty in single-stage production. Management Science, 41(2), 363–376.

Goren, S. & Sabuncuoglu, I. (2008). Robustness and stability measures for scheduling: single-machine environment. IIE Transactions, 40(1), 66-83.

Goren, S. & Sabuncuoglu, I. (2010). Optimization of schedule robustness and stability under random machine breakdowns and processing time variability. IIE Transactions, 42(3), 203-220.

Graham, R.L., Lawler, E.L. & Lenstra, J.K. (1979). Optimization and approximation in deterministic sequencing and scheduling: A survey. Annuals of Discrete Math, 5, 287-326.

Jarboui, B., Eddaly, M. & Siarry, P. (2009). An estimation of distribution algorithm for minimizing the total flowtime in permutation flowshop scheduling problems. Computers & Operations Research, 36, 2638-2646.

Kasperski, A. (2005). Minimizing maximal regret in the single machine sequencing problem with maximum lateness criterion. Operations research Letters, 33, 431-436.

Kasperski., A., Kurpisz, A. & Zielin´ ski., P. (2012). Approximating a two-machine flow shop scheduling under discrete scenario uncertainty. European Journal of Operational Research 217, 36–43.

Kouvelis, P. & Yu, G. (1997). Robust Discrete Optimization and Its Applications. Boston: Kluwer.

Li, K. & Yang, S.-l. (2009). Non-identical parallel-machine scheduling research with minimizing total weighted completion times Models, relaxations and algorithms. Applied Mathematical Modelling, 33, 2145-2158.

Lu, C.C., Lin, S.W. & Ying, K.C. (2012). Robust scheduling on a single machine to minimize total flowtime. Computers & Operations Research, 39, 1682–1691.

Lu, C.C., Ying, K.C. & Lin, S.W. (2014). Robust single machine scheduling for minimizing total flow time in the presence of uncertain processing times. Computers & Industrial Engineering, 74, 102-110.

Montemanni, R. (2007). A Mixed Integer Programming Formulation for the Total Flow Time Single Machine Robust Scheduling Problem with Interval Data. Journal of Mathematic Model Algorithms 6, 287–296.

Moslehi, G. & Mahnam, M. (2010). Minimizing the sum of maximum earliness and tardiness on identical parallel machines scheduling problem. Journal of Industrial Engineering (Engineering Faculty of Tehran University), (In Persian), 44(2), 229-242.

Ranjbar, M., Davari, M. & Leus, R. (2012). Two branch-and-bound algorithms for the robust parallel machine scheduling problem. Computers & Operations Research, 39, 1652–1660.

Webster, S.T. (1993). A priority rule for minimizing weighted flow time in a class of parallel machine scheduling problems. European Journal of Operational Research, 70, 327-334.

Wu, S.D., Byeon, E. & Storer, R.H. (1999). A graph-theoretic decomposition of the job shop scheduling problem to achieve scheduling robustness. Operations Research, 47(1), 113–124.

Wu., C.W., Brown, K.N. & Beck, J.C. (2009). Scheduling with uncertain durations: modeling beta-robust scheduling with constraints. Computers & Operations Research, 36(8), 2348-2356.

Xia, Y., Chen B. & Yue J. (2008). Job sequencing and due date assignment in a single machine shop with uncertain processing times. European Journal of Operational Research 184, 63–75.

Yang, J. & Yu, G. (2002). On the robust single machine scheduling problem. Journal of Combinatorial Optimization, 6, 17-33.

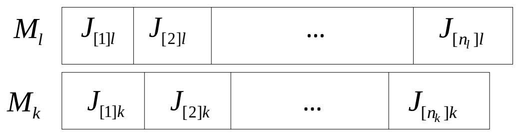


Figure 1: Jobs assigned to machines *l* and *k* in the sequence S

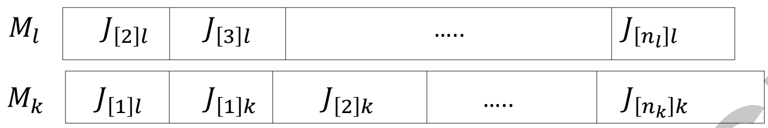


Figure 2: Jobs assigned to machines *l* and *k* in the sequence Ś

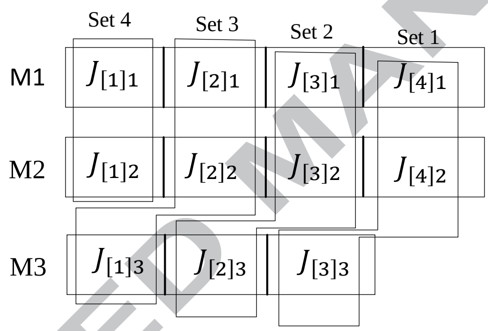


Figure 3: The sequence *S*

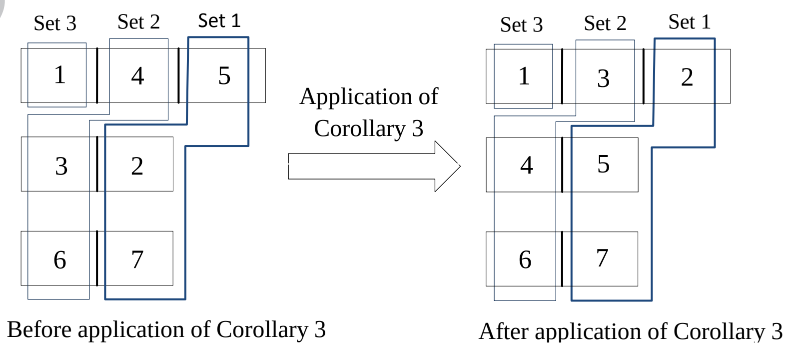


Figure 4: Sequence S before and after application of Corollary 3

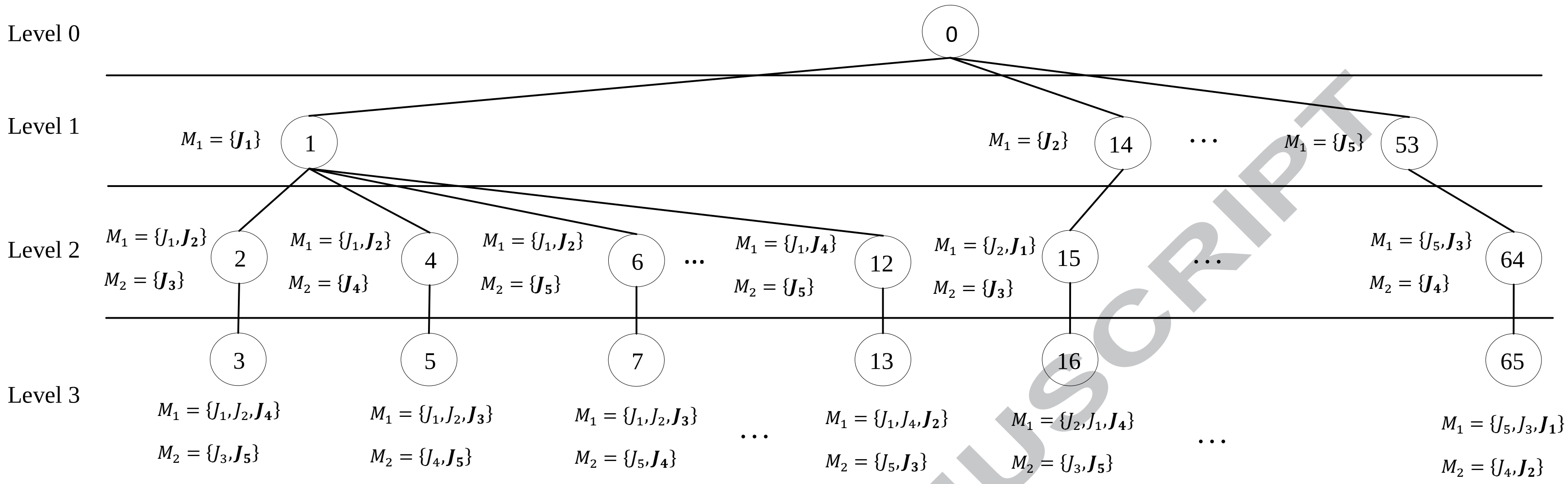


Figure 5: An example of the branching procedure

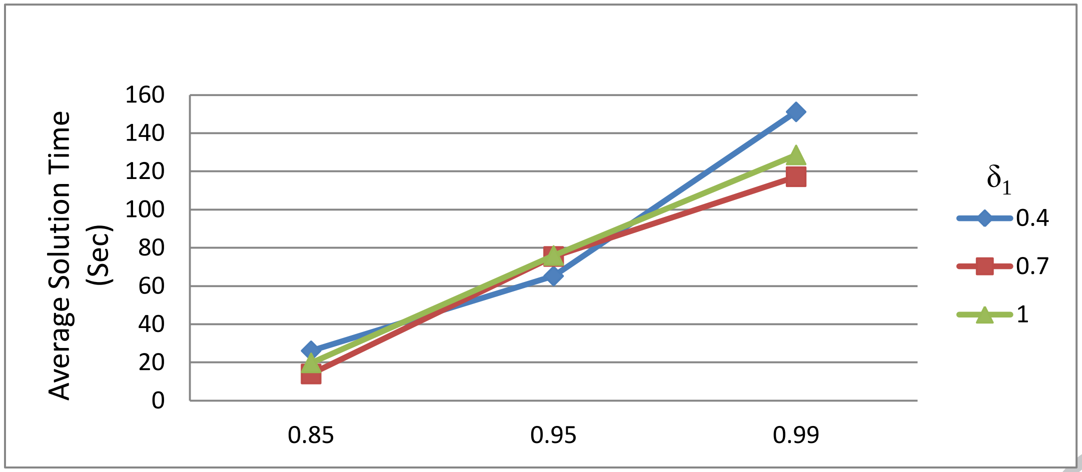


Figure 6: The problem-solving time based on different α values and variations of δ_1

Table1: Results of solving sample problems using B&B

Group	n	m	N. of Optimal Solution	Time Avg. (s)	Fathomed Nodes (%)	Fathomed Nodes Avg. DR1	DR2	DR3
G2 (δ_1=0.4, δ_2=0.7, α=0.85)	30	3	10	0.30	74.08	32.56	64.65	2.79
		4	10	0.16	78.06	27.13	71.79	1.07
		5	10	0.13	76.88	29.66	69.91	0.43
	35	3	10	5.06	69.62	41.96	56.10	1.95
		4	10	1.39	79.09	25.49	73.47	1.03
		5	10	0.52	78.70	26.72	72.92	0.36
	40	3	10	7.21	77.50	27.68	70.77	1.54
		4	10	4.40	75.67	31.32	67.78	0.89
		5	10	4.03	77.71	28.44	71.30	0.26
	45	3	10	283.40	73.06	35.42	62.97	1.61
		4	10	94.69	77.91	27.90	71.61	0.49
		5	10	32.86	78.63	27.00	72.81	0.19
G11 (δ_1=0.4, δ_2=0.7, α=0.95)	30	3	10	1.06	73.27	33.55	63.08	3.37
		4	10	0.49	76.36	29.84	68.93	1.23
		5	10	0.31	75.55	31.90	67.63	0.47
	35	3	10	33.85	67.13	46.91	50.72	2.37
		4	10	4.93	77.34	28.22	70.59	1.19
		5	10	2.10	77.81	28.11	71.46	0.43
	40	3	10	58.79	74.69	32.19	65.88	1.93
		4	10	25.77	74.02	34.08	64.83	1.09
		5	10	16.79	75.80	31.61	68.05	0.34
	45	3	10	561.56	76.73	29.04	69.48	1.48
		4	10	662.12	77.47	28.49	70.88	0.63
		5	10	157.53	77.13	29.39	70.34	0.27
G21 (δ_1=0.4, δ_2=7, α=0.99)	30	3	10	14.85	67.32	43.60	51.11	5.29
		4	10	2.49	73.05	35.27	62.99	1.75
		5	10	0.93	72.43	37.30	62.01	0.70
	35	3	10	71.53	74.89	30.73	66.13	3.14
		4	10	47.17	70.91	39.37	58.89	1.74
		5	10	7.81	74.37	33.89	65.54	0.58
	40	3	10	1048.56	74.45	32.12	65.36	2.52
		4	10	384.49	71.51	38.55	60.08	1.38
		5	10	42.44	80.19	24.25	75.27	0.49
	45	3	10	893.12	83.40	18.28	79.86	1.86
		4	10	1330.13	76.62	29.73	69.40	0.87
		5	10	529.02	80.43	23.98	75.65	0.37

Group	n	m	N. of Optimal Solution	Time Avg. (s)	Fathomed Nodes (%)	Fathomed Nodes Avg. DR1	DR2	DR3
G5 (δ_1=0.7, δ_2=0.7, α=0.85)	30	3	10	0.11	78.70	23.55	72.59	3.85
		4	10	0.14	74.14	33.34	65.01	1.65
		5	10	0.10	77.82	27.86	71.49	0.65
	35	3	10	1.93	74.45	31.36	65.42	3.23
		4	10	0.62	80.39	23.20	75.51	1.29
		5	10	0.24	81.10	22.81	76.68	0.51
	40	3	10	24.79	75.05	30.95	66.53	2.52
		4	10	3.62	78.83	25.59	73.06	1.35
		5	10	1.73	79.80	24.89	74.67	0.44
	45	3	10	109.49	78.93	24.43	73.11	2.46
		4	10	62.02	76.21	30.54	68.72	0.74
		5	10	19.35	81.02	23.16	76.57	0.27
G15 (δ_1=0.7, δ_2=7, α=0.95)	30	3	10	1.41	70.78	36.92	58.36	4.72
		4	10	0.49	74.93	31.96	66.45	1.59
		5	10	0.30	76.41	30.10	69.29	0.61
	35	3	10	20.11	75.10	30.01	66.50	3.49
		4	10	7.40	74.47	32.81	65.61	1.58
		5	10	3.99	73.85	34.75	64.63	0.62
	40	3	10	133.60	78.34	25.43	72.11	2.47
		4	10	32.57	77.95	27.48	71.63	0.89
		5	10	12.82	77.75	28.06	71.37	0.56
	45	3	10	1315.46	75.36	29.69	67.07	3.24
		4	10	967.20	76.65	29.32	69.47	1.21
		5	10	659.25	73.25	36.00	63.46	0.54
G24 (δ_1=0.7, δ_2=7, α=0.99)	30	3	10	3.80	70.21	38.01	57.18	4.82
		4	10	1.02	74.08	33.37	64.92	1.72
		5	10	0.55	75.76	31.12	68.26	0.62
	35	3	10	80.59	73.68	32.16	63.89	3.96
		4	10	21.39	73.02	35.39	62.94	1.67
		5	10	9.52	72.35	37.39	61.86	0.76
	40	3	10	536.55	77.27	26.97	70.30	2.73
		4	10	97.20	76.87	29.12	69.82	1.06
		5	10	26.09	76.81	29.57	69.80	0.64
	45	3	3	945.52	78.81	25.15	72.84	2.01
		4	5	801.41	77.81	27.25	71.41	1.34
		5	8	1520.87	74.79	33.18	66.26	0.56

Group	n	m	N. of Optimal Solution	Time Avg. (s)	Fathomed Nodes (%)	Fathomed Nodes Avg. DR1	DR2	DR3
G8 (δ_1=1, δ_2=0.7, α=0.85)	30	3	10	0.29	76.64	26.77	69.17	4.06
		4	10	0.07	80.24	23.24	75.25	1.51
		5	10	0.09	76.13	30.51	68.63	0.86
	35	3	10	1.95	75.49	29.78	67.23	2.98
		4	10	1.13	77.30	27.98	70.54	1.48
		5	10	0.48	78.07	27.48	71.90	0.63
	40	3	10	30.12	73.17	32.93	63.11	3.96
		4	10	2.83	77.54	27.78	70.96	1.26
		5	10	2.33	75.33	32.13	67.23	0.64
	45	3	10	271.06	78.39	24.69	72.22	3.09
		4	10	66.04	79.26	25.33	73.78	0.89
		5	10	79.05	77.25	29.03	70.53	0.44
G18 (δ_1=1, δ_2=7, α=0.95)	30	3	10	1.56	73.18	31.72	62.88	5.40
		4	10	0.32	75.60	30.31	67.81	1.88
		5	10	0.31	75.50	31.15	68.02	0.82
	35	3	10	33.69	70.11	39.59	57.01	3.39
		4	10	8.73	70.83	39.27	58.71	2.01
		5	10	3.74	75.25	32.15	67.10	0.74
	40	3	10	127.90	71.10	37.66	59.06	3.28
		4	10	71.07	73.82	34.01	64.44	1.55
		5	10	20.26	73.06	36.14	63.16	0.70
	45	3	10	1553.95	73.66	32.56	63.97	3.47
		4	10	487.81	77.26	28.25	70.47	1.28
		5	10	381.01	69.85	42.64	56.82	0.54
G27 (δ_1=1, δ_2=7, α=0.99)	30	3	10	3.38	72.70	32.28	61.96	5.77
		4	10	0.64	74.10	32.87	65.22	1.92
		5	10	0.49	75.32	31.39	67.74	0.87
	35	3	10	129.06	69.09	41.27	54.86	3.87
		4	10	25.35	68.27	44.11	53.45	2.44
		5	10	5.65	74.22	33.90	65.28	0.82
	40	3	10	439.22	69.92	39.39	56.65	3.97
		4	10	224.44	71.43	38.29	59.88	1.82
		5	10	41.61	72.07	37.91	61.30	0.79
	45	3	3	1972.20	79.83	22.25	74.45	3.30
		4	8	1279.79	75.21	31.51	66.92	1.57
		5	9	790.94	70.71	40.76	58.56	0.68

Table2: Risk of instance problem using B&B over SEPT schedule

Group	*n*	*m*	Risk		Group	*n*	*m*	Risk		Group	*n*	*m*	Risk	
			Avg.	Max.				Avg.	Max.				Avg.	Max.
G2	30	3	0.16	1.57	G5	30	3	0.31	1.57	G8	30	3	0.16	1.57
δ_1=0.4, δ_2=0.7, α=0.85		4	0.00	0.00	δ_1=0.7, δ_2=0.7, α=0.85		4	0.16	1.57	δ_1=1, δ_2=0.7, α=0.85		4	0.95	3.18
		5	0.16	1.57			5	0.16	1.57			5	0.31	1.57
	35	3	0.00	0.00		35	3	0.16	1.57		35	3	0.31	1.57
		4	0.00	0.00			4	0.31	1.57			4	0.16	1.57
		5	0.00	0.00			5	0.16	1.57			5	0.47	3.18
	40	3	0.00	0.00		40	3	0.00	0.00		40	3	0.16	1.57
		4	0.00	0.00			4	0.00	0.00			4	0.16	1.57
		5	0.00	0.00			5	0.16	1.57			5	0.31	1.57
	45	3	0.00	0.00		45	3	0.16	1.57		45	3	0.00	0.00
		4	0.00	0.00			4	0.00	0.00			4	0.31	1.57
		5	0.00	0.00			5	0.00	0.00			5	0.47	1.57
G11	30	3	0.62	2.06	G15	30	3	2.53	6.45	G18	30	3	4.72	8.79
δ_1=0.4, δ_2=0.7, α=0.95		4	1.24	2.06	δ_1=0.7, δ_2=7, α=0.95		4	3.61	8.79	δ_1=1, δ_2=7, α=0.95		4	3.40	6.45
		5	1.68	4.21			5	2.75	6.45			5	6.40	15.93
	35	3	1.26	6.45		35	3	3.60	8.79		35	3	3.82	10.99
		4	0.83	4.21			4	3.19	8.79			4	2.31	6.45
		5	0.62	2.06			5	2.33	6.45			5	3.18	6.45
	40	3	0.41	2.06		40	3	1.68	6.45		40	3	2.96	6.45
		4	0.21	2.06			4	1.86	4.21			4	2.74	6.45
		5	0.62	2.06			5	1.25	4.21			5	2.52	6.45
	45	3	0.52	2.06		45	3	1.29	2.06		45	3	2.45	6.45
		4	0.21	2.06			4	1.45	4.21			4	1.86	6.45
		5	0.21	2.06			5	1.46	4.21			5	1.67	4.21
G21	30	3	10.65	17.86	G24	30	3	9.27	20.73	G27	30	3	14.55	23.75
δ_1=0.4, δ_2=7, α=0.99		4	7.24	20.73	δ_1=0.7, δ_2=7, α=0.99		4	11.35	23.75	δ_1=1, δ_2=7, α=0.99		4	10.25	20.73
		5	17.71	50.00			5	8.21	20.73			5	19.39	45.59
	35	3	6.94	11.24		35	3	10.29	20.73		35	3	11.92	32.00
		4	8.83	17.86			4	10.47	23.75			4	7.36	17.86
		5	6.80	13.79			5	7.90	20.73			5	11.00	20.73
	40	3	8.93	17.86		40	3	7.52	20.73		40	3	8.37	17.86
		4	8.39	13.79			4	6.45	11.24			4	9.91	20.73
		5	4.44	11.24			5	7.01	13.79			5	8.11	20.73
	45	3	4.22	5.32		45	3	6.48	8.79		45	3	7.63	8.79
		4	3.67	5.32			4	5.51	13.79			4	4.93	8.79
		5	4.27	8.79			5	7.39	11.24			5	6.03	8.79

Table 3: Comparison between the results obtained from B&B and the method proposed by Wu et al. in a single-machine problem

Number of Jobs (n)	α	Number of Instance Problems	Number of Optimal Solutions	
			***B**&B*	**Wu & et. al.**
10	0.85	90	90	90
	0.95	90	90	90
	0.99	90	90	90
15	0.85	90	90	90
	0.95	90	90	90
	0.99	90	90	90
20	0.85	90	90	90
	0.95	90	90	90
	0.99	90	90	90
25	0.85	90	90	-
	0.95	90	90	-
	0.99	90	90	-
30	0.85	90	90	-
	0.95	90	80	-
	0.99	90	60	-

Highlights

- We consider β-robust approach on parallel machine with Total Flow Time for the first time.
- We propose a branch and bound algorithm with particular branching procedure.
- For the first time we optimally solve β-robust scheduling problem up to 45 jobs.
- Our algorithm for a single machine is better and more efficient than previous works.