Block Models for Scheduling Jobs on Two Parallel Machines with a Single Server

Keramat Hasani
Islamic Azad University, Malayer Branch,
Malayer, Iran

Svetlana A. Kravchenko
United Institute of Informatics Problems,
Surganova St. 6, 220012 Minsk, Belarus

Frank Werner [*]
Fakultät für Mathematik,
Otto-von-Guericke-Universität Magdeburg,
Postfach 4120, 39016 Magdeburg, Germany

January 16, 2013

Abstract

We consider the problem of scheduling a set of non-preemptable jobs on two identical parallel machines such that the makespan is minimized. Before processing, each job must be loaded on a machine, which takes a given setup time. All these setups have to be done by a single server which can handle at most one job at a time. For this problem, we propose a mixed integer linear programming formulation based on the idea of decomposing a schedule into a set of blocks. We compare the results obtained by the model suggested with known heuristics from the literature.

Keywords: scheduling, parallel machines, single server

[*]Corresponding author. Tel.:+0391 6712025; fax:+0391 6711171.
E-mail address: frank.werner@mathematik.uni-magdeburg.de (F. Werner).

1 Introduction

The problem considered in this paper can be described as follows. There are n independent jobs and two identical parallel machines. For each job J_j, $j = 1, \ldots, n$, there is given its processing time p_j. Before processing, a job must be loaded on the machine M_q, $q = 1, 2$, where it is processed which requires a setup time s_j. During such a setup, the machine M_q is also involved into this process for s_j time units, i.e., no other job can be processed on this machine during this setup. All setups have to be done by a single server which can handle at most one job at a time. The goal is to determine a feasible schedule which minimizes the makespan. So, using the common notation, we consider the problem $P2, S1 \mid\mid C_{max}$. This problem is strongly NP-hard since problem $P2, S1 \mid s_j = s \mid C_{max}$ is unary NP-hard [5]. The interested reader is referred to [3] and [6] for additional information on server scheduling models.

Several heuristics were developed for the problem $P2, S1 \mid\mid C_{max}$ under consideration so far. In Abdekhodaee et al. [2], two versions of a greedy heuristic, a genetic algorithm and a version of the Gilmory-Gomory algorithm were proposed and tested. The analysis started in [2] was extended in Gan et al. [4], where two mixed integer linear programming formulations and two variants of a branch-and-price scheme were developed. Computational experiments have shown that for small instances with $n \in \{8, 20\}$, one of the mixed integer linear programming formulations was the best whereas for the larger instances with $n \in \{50, 100\}$, the branch-and-price scheme worked better, see [4].

In this paper, we propose a mixed integer linear programming formulation for the problem $P2, S1 \mid\mid C_{max}$, based on the structure of an optimal schedule. The proposed models use the simple idea of a possible decomposition of any schedule into a set of blocks, which significantly contributes to a reduction of the number of jobs. We compare the performance of this model with the heuristics proposed in [4].

The remainder of the paper is organized as follows. In Section 2, we introduce two block models for the problem under consideration and give the resulting mixed integer programming formulations. In Section 3, we present computational results and perform a comparison with existing heuristics. Finally, we give some concluding remarks in Section 4.

2 Block Models

It is easy to see that any schedule for the problem $P2, S1 \mid\mid C_{max}$ can be considered as a unit of blocks $B_1, \ldots, B_z$, where $z \leq n$. Each block B_k can be completely defined by the first level job J_a and a set of second level jobs $\{J_{a1}, \ldots, J_{ak}\}$, where inequality $p_a \geq s_{a1} + \ldots + s_{ak} + p_{a1} + \ldots + p_{ak}$ holds, see Figure 1.

For example, for the schedule given in Figure 2, we can define the four blocks B_1, B_2, B_3, and B_4. The block B_1 is defined by the first level job J_1 and by the second level job

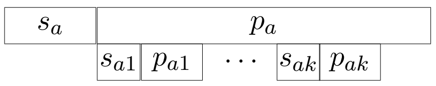

Figure 1: Each block can be completely defined by the first level job J_a and a set of the second level jobs $\{J_{a1}, \ldots, J_{ak}\}$.

$\{J_2\}$; the block B_2 is defined by the first level job J_3 and by the second level job $\{J_4\}$; the block B_3 is defined by the first level job J_5 and an empty set of second level jobs; the block B_4 is defined by the first level job J_6 and by the second level jobs $\{J_7, J_8\}$.

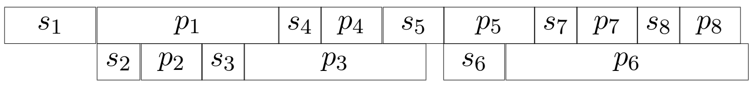

Figure 2: A schedule with four blocks.

Thus, the model that we suggest is based on the fact that any schedule can be decomposed into a set of blocks. The variable $B_{k,f,j}$ is used for a block. We have $B_{k,f,j} = 1$ if job J_j is scheduled in level f in the k-th block, otherwise $B_{k,f,j} = 0$. The index $k = 1, \ldots, n$ indicates the serial number of the block. The index $f \in \{1, 2\}$ indicates the level, i.e., we have $f = 1$ if the level is the first one, and $f = 2$ if the level is the second one. The index $j = 1, \ldots, n$ indicates the job.

Each job belongs to some block, i.e., for any $j = 1, \ldots, n$, the equality

$$\sum_{k=1}^{n} \sum_{y=1}^{2} B_{k,y,j} = 1 \tag{2.1}$$

holds. There is only one job of the first level for each block, i.e., for each $y = 1$ and for any $k = 1, \ldots, n$, the inequality

$$\sum_{j=1}^{n} B_{k,1,j} \leq 1 \tag{2.2}$$

holds.

Since all blocks are given, we define the following data for each block B_k, where $k = 1, \ldots, n$:

- The loading part of the block B_k has the length $ST_k \geq 0$, formally inequality

$$ST_k \geq \sum_{j=1}^{n} s_j B_{k,1,j} \tag{2.3}$$

holds.

- The objective part of the block B_k has the length $\sum_{j=1}^{n}(s_j + p_j)B_{k,2,j}$.
- The processing part of the block B_k has the length $PT_k \geq 0$, formally inequality

$$PT_k \geq \sum_{j=1}^{n} p_j B_{k,1,j} - \sum_{j=1}^{n}(s_j + p_j)B_{k,2,j} \tag{2.4}$$

holds.

Thus, each block is composed into three parts: *loading*, *objective*, and *processing.*

We add the objective part to the objective function and delete it from the block. After deleting the objective part from each block, the schedule can be considered as a set of modified jobs J'_k with the setup time ST_k and the processing time PT_k. The jobs J'_k, $k = 1, \ldots, n$, are scheduled in staggered order, i.e., job J'_1 is scheduled on the first machine, job J'_2 is scheduled on the second machine, job J'_3 is scheduled on the first machine, job J'_4 is scheduled on the second machine, and so on.

In Figure 2, we have a schedule consisting of four blocks.

- For the first block we have $B_{1,1,1} = 1$, i.e., J_1 is the first level job, and $B_{1,2,2} = 1$, i.e., J_2 is the second level job. The modified job J'_1 has the loading part $ST_1 = s_1$ and the processing part $PT_1 = p_1 - s_2 - p_2$.
- For the second block we have $B_{2,1,3} = 1$, i.e., J_3 is the first level job, and $B_{2,2,4} = 1$, i.e., J_4 is the second level job. The modified job J'_2 has the loading part $ST_2 = s_3$ and the processing part $PT_2 = p_3 - s_4 - p_4$.
- For the third block we have $B_{3,1,5} = 1$, i.e., J_5 is the first level job, and there are no second level jobs in this block. The modified job J'_3 has the loading part $ST_3 = s_5$ and the processing part $PT_3 = p_5$.
- For the fourth block we have $B_{4,1,6} = 1$, i.e., J_6 is the first level job, and there are two jobs J_7 and J_8 of the second level, i.e., $B_{4,2,7} = 1$ and $B_{4,2,8} = 1$. The modified job J'_4 has the loading part $ST_4 = s_6$ and the processing part $PT_4 = p_6 - s_7 - p_7 - s_8 - p_8$.

The jobs J'_1, J'_2, J'_3, J'_4 are processed alternately on the two machines.

Formally, if we denote by st_j the starting time of each modified job J'_j, then
$st_1 + ST_1 \leq st_2$, $st_2 + ST_2 \leq st_3$, and so on, i.e., inequality

$$st_j + ST_j \leq st_{j+1} \tag{2.5}$$

holds for each $j = 1, \ldots, n - 1$;
$st_1 + ST_1 + PT_1 \leq st_3$, $st_2 + ST_2 + PT_2 \leq st_4$, and so on, i.e., inequality

$$st_j + ST_j + PT_j \leq st_{j+2} \tag{2.6}$$

holds for each $j = 1, \ldots, n-2$.
We denote by F the total length of the modified schedule, i.e., inequality

$$F \geq st_n + ST_n + PT_n \tag{2.7}$$

holds,
and inequality

$$F \geq st_{n-1} + ST_{n-1} + PT_{n-1} \tag{2.8}$$

holds.
For each job J_j, the integer number $ch[j]$ is introduced with the following meaning. If J_j is the first level job for some block B_x, then $ch[j]$ denotes the maximal number of second level jobs for the same block. Formally, one can write

$$B_{x,2,1} + \ldots + B_{x,2,n} \leq ch[1]B_{x,1,1} + \ldots + ch[n]B_{x,1,n} \tag{2.9}$$

The objective function is

$$F + \sum_{x=1}^{n} \sum_{j=1}^{n} (s_j + p_j) B_{x,2,j}. \tag{2.10}$$

Since any schedule can be decomposed into a set of blocks, the following theorem holds.

Theorem 1 *Any schedule s can be described as a feasible solution of system (2.1) - (2.8) and as a feasible solution of system (2.1) - (2.9), respectively. In both cases, equality*

$$C_{max}(s) = F + \sum_{x=1}^{n} \sum_{j=1}^{n} (s_j + p_j) B_{x,2,j}$$

holds.

Now, to prove the equivalence between the scheduling problem $P2, S1||C_{max}$ and the models (2.1) - (2.8) and (2.1) - (2.9), respectively, one has to prove the following theorem.

Theorem 2 *Any feasible solution of system (2.1) - (2.8) and any feasible solution of system (2.1) - (2.9), respectively, can be described as a feasible schedule for the problem $P2, S1||C_{max}$. In both cases, equality*

$$F + \sum_{x=1}^{n} \sum_{j=1}^{n} (s_j + p_j) B_{x,2,j} = C_{max}(s)$$

holds.

Proof: Suppose that we have some feasible solution of system (2.1) - (2.8). Using the values st_j, ST_j and PT_j, one can reconstruct the schedule s' for the set of modified jobs J'_1, J'_2, $\ldots$, J'_n. Since all these jobs are scheduled in staggered order, it is sufficient to consider only the following three cases for the possible scheduling of two adjacent jobs, say J'_j and J'_{j+1}.

Case 1: Equality $st_j + ST_j = st_{j+1}$ holds. In this case, one can divide s' into two parts at the time point $st_j + ST_j$. The objective part of job J'_j is scheduled between the two parts of the divided schedule s'.

Case 2: Inequalities $st_j + ST_j < st_{j+1}$ and $st_j + ST_j + PT_j \geq st_{j+1}$ hold. In this case, one can divide s' into two parts at the time point st_{j+1}. The objective part of job J'_j is scheduled between the two parts of the divided schedule s'.

Case 3: Inequalities $st_j + ST_j < st_{j+1}$ and $st_j + ST_j + PT_j < st_{j+1}$ hold. In this case, one can shift the job J'_j to the right in such a way that the completion time of J'_j is equal to st_{j+1}. Now, one can divide the obtained schedule into two parts at the time point st_{j+1}. The objective part of job J'_j is scheduled between the two parts of the divided schedule.

In such a way, one can reconstruct a feasible schedule using the obtained solution of system (2.1) - (2.8).

In the same way one can reconstruct a feasible schedule for any feasible solution of system (2.1) - (2.9). It is sufficient to note that the set of all feasible solutions of (2.1) - (2.9) is a subset for the set of all feasible solutions of (2.1) - (2.8). □

In fact, our models divide the problem into two subproblems. The first one consists in assigning the jobs to blocks. This step can considerably reduce the number of jobs, since any second level job can be omitted. The second one consists in arranging the blocks in a list.

Thus, we consider two models in the following.
Model BM1: Minimize (2.10) subject to the constraints (2.1) - (2.9), and
model BM2: Minimize (2.10) subject to the constraints (2.1) - (2.8).

To evaluate the obtained results, we use the known lower bound

$$LB = \max\{LB_1, LB_2, LB_3\},$$

where

$$LB_1 = \frac{1}{2}\left(\sum_{i \in J}(s_i + p_i) + \min_{i \in J}\{s_i\}\right),$$

$$LB_2 = \sum_{i \in J} s_i + \min_{i \in J}\{p_i\},$$

$$LB_3 = \max_{i \in J}\{s_i + p_i\}$$

(see [4]). Next, we compare the models BM1 and BM2 with the heuristics developed in [4].

3 Computational Results

The performance of the models BM1 and BM2 was tested on the data generated in the same way as it is described in [1] and [4].

For $n \in \{8, 20\}$, the data sets were generated for server load values ranging between 0.1 and 2 with 0.1 increments, i.e., for each $L \in \{0.1, 0.2, \dots, 2\}$ the value s_j is uniformly distributed in $(0, 100L)$. For each value L, 10 instances were randomly generated with $p_j \overset{d}{=} U(0, 100)$, i.e., p_j is uniformly distributed in the interval $(0, 100)$.

For $n \in \{50, 100, 200, 250\}$, 5 instances were generated for each $L \in \{0.1,\ 0.5,\ 0.8,\ 1,\ 1.5,\ 1.8,\ 2.0\}$ with $p_j \overset{d}{=} U(0, 100)$, and $s_j \overset{d}{=} U(0, 100L)$.

The test instances have been solved using CPLEX 10.1 with 2GB of memory available for working storage, running on a personal computer Intel(R) Core(TM)i5-2430M CPU @2.4GHz.

The model BM1 was used for $n \in \{8, 20, 50\}$ since the model BM2 works slowly in these cases. For $n = 100$, the model BM1 was unable to find any solution in 10% of the instances and for this reason, the model BM2 was used for the instances with $n \in \{100, 200, 250\}$.

Some computational results for $n \in \{8, 20, 50\}$ and $L \in \{0.1,\ 0.5,\ 0.8,\ 1,\ 1.5,\ 1.8,\ 2.0\}$ are given in Table 1.

Time limit		Load	0.1	0.5	0.8	1.0	1.5	1.8	2.0
50 sec	$n = 8$	Average $\frac{C_{max}}{LB}$	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		Maximal $\frac{C_{max}}{LB}$	1.00	1.00	1.00	1.00	1.00	1.00	1.00
750 sec	$n = 20$	Average $\frac{C_{max}}{LB}$	1.00	1.00	1.01	1.02	1.01	1.00	1.00
		Maximal $\frac{C_{max}}{LB}$	1.00	1.00	1.05	1.05	1.05	1.01	1.02
1875 sec	$n = 50$	Average $\frac{C_{max}}{LB}$	1.00	1.02	1.04	1.03	1.01	1.00	1.00
		Maximal $\frac{C_{max}}{LB}$	1.00	1.02	1.05	1.04	1.02	1.00	1.00

Table 1: Detailed statistics for the model BM1

Comparing the obtained results for the model BM1 with the results from [4], we can claim the following.

1. For $n = 8$, we were able to find optimal solutions for all generated instances within 50 seconds. In [4], for the same instances with $n = 8$, optimal solutions for all generated instances were only found within 300 seconds.

2. For $n = 20$, we used a run time limit of 750 seconds.

- The maximal value for the relation $\frac{C_{max}}{LB}$ was 1.05, and the average value for the relation $\frac{C_{max}}{LB}$ was 1.01
- while in [4], the maximal value for the relation $\frac{C_{max}}{LB}$ was 1.08 and the average value for the relation $\frac{C_{max}}{LB}$ was 1.01.

3. For $n = 50$, we used a run time limit of 1875 seconds.

   - The maximal value for the relation $\frac{C_{max}}{LB}$ was 1.05, and the average value for the relation $\frac{C_{max}}{LB}$ was 1.01
   - while in [4], the best makespan was compared not with LB but with the worst makespan among the heuristics developed. The relation "Worst makespan/Best makespan" was ranging from 1.02 to 1.09.

For the model BM2, detailed results are given in Table 2.

Time limit		Load	0.1	0.5	0.8	1.0	1.5	1.8	2.0
3750 sec	$n = 100$	Average $\frac{C_{max}}{LB}$	1.02	1.01	1.04	1.05	1.01	1.00	1.01
		Maximal $\frac{C_{max}}{LB}$	1.03	1.02	1.04	1.08	1.02	1.01	1.02
3600 sec	$n = 200$	Average $\frac{C_{max}}{LB}$	1.01	1.04	1.07	1.09	1.01	1.00	1.00
		Maximal $\frac{C_{max}}{LB}$	1.01	1.08	1.10	1.12	1.02	1.01	1.01
3600 sec	$n = 250$	Average $\frac{C_{max}}{LB}$	1.02	1.07	1.10	1.10	1.02	1.00	1.00
		Maximal $\frac{C_{max}}{LB}$	1.03	1.09	1.11	1.12	1.04	1.00	1.01

Table 2: Detailed statistics for the model BM2

Comparing the obtained results for the model BM2 with [4], we can summarize the following.

1. For $n = 100$, we used a run time limit of 3750 seconds.

   - The maximal value for the relation $\frac{C_{max}}{LB}$ was 1.08, and the average value for the relation $\frac{C_{max}}{LB}$ was 1.02
   - while in [4], the relation "Worst makespan/Best makespan" was ranging from 1.01 to 1.07.

2. For $n = 200$ and for $n = 250$, we used a run time limit of 3600 seconds.

   - The maximal values and the average values of C_{max}/LB for each load value are presented in Table 2,
   - while in [4], tests have been made only for $n \leq 100$.

In Figure 3, we show the dependence of the average value of $\frac{C_{max}}{LB}$ on the time limit for the instances with $n = 200$ and $L \in \{0.1, 1.0, 2.0\}$.

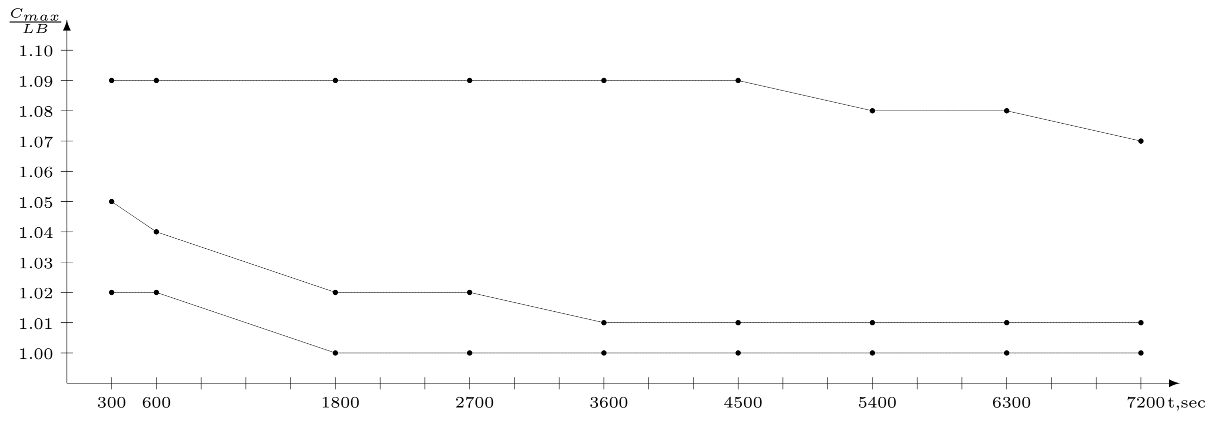

Figure 3: The dependence of the solution quality on the time limit for the instances with $n = 200$. The upper curve refers to the instances with $L = 1.0$, the middle curve refers to the instances with $L = 0.1$, and the lower curve refers to the instances with $L = 2.0$.

4 Concluding Remarks

We developed a mixed integer linear programming formulation for the problem of scheduling a set of jobs on two parallel machines with a single server. Two models were tested and the performance was compared with that of the heuristics developed in [4]. The computational results show that both models BM1 and BM2 clearly outperform all heuristics proposed in [4].

References

[1] Abdekhodaee A., Wirth A., 2002. Scheduling parallel machines with a single server: some solvable cases and heuristics. Computers & Operations Research 29, 295–315.

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

[3] Brucker P., Dhaenens-Flipo C., Knust S., Kravchenko S.A., Werner F., 2002. Complexity results for parallel machine problems with a single server. Journal of Scheduling 5, 429–457.

[4] Gan H. S., Wirth A., Abdekhodaee A., 2012. A branch-and-price algorithm for the general case of scheduling parallel machines with a single server. Computers & Operations Research 39, 2242–2247.

[5] Hall N., Potts C., Sriskandarajah C., 2000. Parallel machine scheduling with a common server. Discrete Applied Mathematics 102, 223–243.

[6] Werner F., Kravchenko S.A., 2010. Scheduling with multiple servers. Automation and Remote Control 71, 2109–2121.