A mixed integer programming formulation for parallel machines scheduling with a single server

Mi-Yi Kim
Young Hoon Lee

Department of Information and Industrial Engineering
Yonsei University
252 Sungsanno Seodaemoon-gu Seoul, Korea
youngh@yonsei.ac.kr

Abstract
This paper considers the identical parallel machines scheduling problem (PMSP) with a single server, which is in charge of job setups. A job can be processed with a precedent setup by a server on one of the machines. The setup can be processed at only one machine at any time because of the single server constraint. In this paper, the Problem P,S1|sj|Cmax with a general job set is formulated in a mixed integer programming, which is developed by taking the characteristic of the single server problem into account and modifying the description of server waiting time identified by Adbeckhodaei and Wirth (2002).

Keywords: Scheduling, MIP, Parallel machines, Single server

1. Introduction

In this paper, identical parallel machines scheduling problem (PMSP) with a single server, who is in charge of the setup for loading the job or handling tools on the machines, is considered. This problem is denoted by P,S1|sj|Cmax, the makespan (maximum completion time), Cmax, to be minimized under the arbitrary number of identical parallel machines, and the arbitrary number of jobs, where a setup has to be processed by a single server. The setup can be processed at any time before the job processing in the general PMSP. In the problem environment suggested in this paper, the setup can be processed by a single server, hence has
to be performed at only one machine at any time. It is assumed that the preemption of the setup and the job processing is not allowed.

Kravchenko and Werner(1997) showed that Problem P,S1|sj=1|Cmax is unary NP-Hard and presented a pseudopolynomial algorithm for Problem P2,S1|sj=1|Cmax. The optimal schedule can be obtained by transforming schedule obtained from Problem P2||Cmax into a feasible one for Problem P2,S1|sj=1|Cmax. Exact solutions and dominant rules for special cases such as equal length of job and regular job are suggested by Abdekhodaee and Wirth(2002). The integer programming formulation by describing the waiting time of server and idle time of machines is suggested for Problem P2,S1|sj=Cmax with regular job environment. Let sj and pj denote the setup time and the processing time of job j. Equal length of jobs are defined such that the length sj + pj of job j is equal to a (constant value) for all jobs. Jobs are defined as ‘regular’ such that the length of job j is greater than or equal to the processing time of any job h for all jobs j, h. Adbekhodaee et al.(2004) has investigated special cases of Problem P2,S1|sj|Cmax and proved that an optimal sequence exists in cases of equal processing time and equal setup time. The exact algorithm for the equal length job case using a sorting is suggested in Adbekhodaee et al.(2006). They have proposed also a greedy heuristics and a metaheuristic such as Genetic Algorithm or the Gilmore-Gomory algorithm to solve the general case of Problem P2,S1|sj|Cmax.

In this paper, a mixed integer programming formulation for general case of PMSP with a single server(Problem P,S1|sj|Cmax) and dominant conditions for the efficient search are suggested. The computational experiments are performed to discuss the characteristics of Problem P,S1|sj|Cmax.

2. MIP model
Let Si and Pi be setup time and processing time of the i-th scheduled job for i = 1, ..., n, regardless of the machine. Notice that sj and pj are given for the job j, while Si and Pi are determined for the job at the i-th sequence.

In the parallel machines scheduling problem, the job can be processed when the machine is available. But, in the parallel machines scheduling problem with a single server, the (i+1)th scheduled job may not be started its setup until the setup of the ith scheduled job is completed due to the single server constraint as shown in Figure 1.
The characteristic of the single server problem is that job sequence of PMSP with a single server is the same as setup sequence of the single server, regardless of the number of machines.

An integer programming formulation for Problem P2,S1| sj | Cmax with a regular job set was suggested by Abdekhodaee and Wirth(2002). In the regular job environment, the processing of jobs has to be completed in the same order of the start of job processing (Figure 2(a)), hence the jobs are processed alternately on the machines (Figure 2(a)). But, in case of PMSP with a general job set, the i th scheduled job may be completed earlier than the (i-1) th scheduled job as Figure 2(b).

In this paper, a mixed integer programming formulation for Problem P,S1| sj | Cmax with a general job set is developed using the waiting time of the server and the characteristic of the server problem. Let W_i denote the i th waiting time of the server. ST_i indicates the starting time of the setup for the i th scheduled job to be performed by the single server. CT_i denotes the completion time of the i th scheduled job.

The i th waiting time of server means the time elapsed from the completion of the i th setup to the start of the (i+1) th setup, W_i = ST_{i+1} - (ST_i + S_i) for i = 1, ..., n - 1. The n th waiting time of server is defined as the difference between maximum completion time of machines and the end of the last setup, W_n = max_{i=1}^n (CT_i) - (ST_n + S_n). Completion time of the i th
scheduled job is equal to the sum of starting time of the setup for the \( i \) th scheduled job, setup time and processing of the \( i \) th scheduled job, \( CT_i = ST_i + S_i + P_i \).

By definitions, the makespan is equal to the sum of setup time of all jobs and waiting time in the sequence. Assume that the makespan is determined by the \( h \) th scheduled job.

\[
\text{makespan} = \sum_{i=1}^{n} S_i + \sum_{i=1}^{n} W_i \quad \text{since} \quad CT_h = \max_{i \in [n]} \{CT_i\}, \quad W_n = \max_{i \in [n]} \{CT_i\} - ST_n - S_n \quad \text{and} \quad ST_{i+1} - ST_i = S_i + W_i.
\]

The starting time of the \((i+1)\) th setup is influenced by the available time of server and the available time of machine on which the \((i+1)\) th scheduled job is assigned. The available time of server is equal to the completion time of the \( i \) th setup and the available time of that machine is equal to the completion time of the \( g \) th scheduled job which is processed right before the \((i+1)\) th scheduled job on the same machine. If the completion time of the machine is bigger than the available time of server then the \( i \) th waiting time of server is occurred. It can be expressed by:

\[
W_i = (ST_{i+1} - (ST_i + S_i))^+ = (P_g - \sum_{h=i+1}^{n} S_h - \sum_{h=i}^{n-1} W_h)^+.
\]

There are \( n \) jobs for \( j = 1, \ldots, n \) and \( m \) identical parallel machines for \( k = 1, \ldots, m \). The setup time \( s_j \) and the processing time \( p_j \) for the job \( j \) are given. Decision variables are: \( x_{ij} \) is 1 if the job \( j \) is assigned in the \( i \) th sequence, 0, otherwise, and \( y_{ik} \) is 1 if the \( i \) th scheduled job is processed on the machine \( k \), 0, otherwise. \( W_i \) is the \( i \) th waiting time of the server.

Formulation)

Minimize \( \sum_{i=1}^{n} W_i \) \hspace{1cm} (1)

Subject to

\[
\sum_{j=1}^{n} x_{ij} = 1 \quad \forall i
\]

\[\sum_{i=1}^{n} x_{ij} = 1 \quad \forall j \]

\[
\sum_{k=1}^{m} y_{ik} = 1 \quad \forall i
\]

\[
W_i \geq \sum_{j=1}^{n} x_{ij} p_j - \sum_{h=g+1}^{i} \sum_{j=1}^{n} x_{hh} s_j - \sum_{h=g}^{i-1} W_h - M(2 - y_{i+1,k} - y_{g,k}) \quad \forall i \in [1, n-1], g \leq i, k
\]

\[
W_n \geq \sum_{j=1}^{n} x_{nj} p_j - \sum_{h=i+1}^{n} \sum_{j=1}^{n} x_{hh} s_j - \sum_{h=i}^{n-1} W_h \quad \forall j
\]
\[ x_{ij} \in \{0,1\} \quad \forall i, j \]  
\[ y_{ik} \in \{0,1\} \quad \forall i, k \]  
\[ W_i \geq 0 \quad \forall i \]  

The objective function implies that minimizing the makespan is equivalent to minimizing the waiting time. Constraints (2)-(4) are the scheduling requirements. Constraints (5)-(6) mean the \( i \) th waiting time of the server. Constraints (7)-(8) are integrality requirements and constraints (9) are non-negativity restriction.

The formulation and the dominant conditions are computationally impractical to be implemented. For a practical use, the dominant conditions are suggested as below.

Makespan is minimized by assigning the \( i \) th scheduled job on the machine with the shortest available time. If not, the completion time of machines and the available time of server will be increased by more than zero and maximum difference of machines’ available time. Makespan is minimized by sequencing such that \( P_i \leq P_{i+1} \) for all sequences where the \( i \) th and \( (i+1) \) th scheduled jobs are assigned on the same machine. The reason for this, the completion time of the \( (i+1) \) th scheduled job is not changed when the jobs are interchanged in schedule. But the time of server available for the \( (i+2) \) th setup can be increased by maximum difference of jobs’ processing time. Makespan is minimized by sequencing such that \( P_i \leq P_{i+1} \) for all sequences where the length of the \( (i+1) \) th scheduled job is equal to one of the \( (i+1) \) th scheduled job and is assigned on the different machine. When the jobs are interchanged in schedule, the completion time of machines are not changed since the jobs have the same length. But the time of server available for the \( (i+2) \) th setup can be increased by maximum difference of jobs’ setup time.

These dominant conditions are applied in some special conditions. The heuristics to get the near-optimal solution are strongly encouraged to be developed.

4. Computational experiments

Data sets used in the experiment are generated randomly using the combination of four parameters \( (n, m, \alpha, \rho) \): \( n \) is the number of jobs, \( m \) is the number of machines, \( \alpha \) is the diversity factor, \( \rho \) is the setup time severity factor. Setup time data and the processing time data are generated in the range of \( \alpha \) times diversity from their mean values prespecified. In our experiment, \( \alpha \) and \( \rho \) are set in \([0.1, 0.5]\) and \([0.5, 1.0]\) respectively. For each combination...
of factors, 10 instances are generated with a different random number seed to have an average result.

\[ s_j = [\bar{s} - \alpha \bar{s}, \bar{s} + \alpha \bar{s}] \forall j, \quad p_j = [\bar{p} - \alpha \bar{p}, \bar{p} + \alpha \bar{p}] \forall j, \quad \rho = m \sum_{j=1}^{n} s_j / \sum_{j=1}^{n} p_j \]

Computational experiment with small instances was performed to get the character of PMSP with a single server. The optimal solution is obtained by using ILOG OPL Development Studio 5.5, on Pentium IV PC with 2.5 GHz CPU and 2GB RAM.

<table>
<thead>
<tr>
<th>$n$</th>
<th>$m$</th>
<th>$\alpha$</th>
<th>( \rho )</th>
<th>( 0.5 )</th>
<th>( 0.7 )</th>
<th>( 1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2</td>
<td>0.1</td>
<td>3.31</td>
<td>0.07</td>
<td>0.044</td>
<td>0.07</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>50.84</td>
<td>3.43</td>
<td>3.05</td>
<td>0.28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>47.87</td>
<td>3.14</td>
<td>3.18</td>
<td>0.93</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>0.125</td>
<td>94</td>
<td>59.21</td>
<td>138.409</td>
<td>813.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3218</td>
<td>5</td>
<td>72.126</td>
<td>155.295</td>
<td>769.44</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5256</td>
<td>52</td>
<td>45.506</td>
<td>176.944</td>
<td>168.87</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>0.1</td>
<td>2.792</td>
<td>2.33</td>
<td>2.96</td>
<td>2.82</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>2.242</td>
<td>2.46</td>
<td>2.68</td>
<td>2.82</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>2.282</td>
<td>2.37</td>
<td>2.82</td>
<td>2.82</td>
</tr>
<tr>
<td>20</td>
<td>6</td>
<td>0.1433</td>
<td>7</td>
<td>824.56</td>
<td>2524.40</td>
<td>369.91</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3273</td>
<td>9</td>
<td>865.13</td>
<td>2122.90</td>
<td>369.91</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5217</td>
<td>0.24</td>
<td>54.05</td>
<td>2554.40</td>
<td>157.6</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>0.1588</td>
<td></td>
<td>6.05</td>
<td>15.76</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3762</td>
<td></td>
<td>8.80</td>
<td>369.91</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5174</td>
<td></td>
<td>5.10</td>
<td>47.05</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>6</td>
<td>0.1</td>
<td>128.39</td>
<td>1599.93</td>
<td>2518.99</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>1082.23</td>
<td>1340.22</td>
<td>2254.41</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>1590.77</td>
<td>1343.85</td>
<td>2570.29</td>
<td>29</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>0.1488</td>
<td>44</td>
<td>482.38</td>
<td>1931.53</td>
<td>95</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3596</td>
<td>0.06</td>
<td>751.96</td>
<td>2233.19</td>
<td>95</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>861.52</td>
<td>1.262</td>
<td>2559.24</td>
<td>95</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>0.1381</td>
<td>2</td>
<td>47.27</td>
<td>141.84</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>38.57</td>
<td>46.4</td>
<td>198.1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>54.42</td>
<td></td>
<td>216.44</td>
<td></td>
</tr>
</tbody>
</table>

In result, computational time to acquire optimal solution is geometrically increased as the number of jobs has increased. But computational time is sharply decreased by increasing the number of machines.

5. References