Polynomial Lower Bounds for the Two-Machine Flowshop Problem with Sequence-Independent Setup Times

Anis Gharbi\textsuperscript{1,2}, Talel Ladhari\textsuperscript{1,3}

Princess Fatimah Alnijris’s Research Chair for AMT
Department of Industrial Engineering
College of Engineering, King Saud University, Riyadh, Saudi Arabia

Mohamed Kais Msakni\textsuperscript{4}

Combinatorial Optimization Research Group – ROI
Ecole polytechnique de Tunisie, BP 743, 2078 La Marsa, Tunisia

Mehdi Serairi\textsuperscript{5}

Heudiasyc UMR 6599, Centre de Recherches de Royallieu
Université de Technologie de Compiègne
France

Abstract

In this paper, we address the problem of two-machine flowshop scheduling problem with sequence independent setup times to minimize the total completion time. We propose five new polynomial lower bounds. Computational results based on randomly generated data show that our proposed lower bounds consistently outperform those of the literature.

Keywords: Flowshop, setup times, total completion time, lower bounds
1 Introduction

We consider the strongly $\mathcal{NP}$-hard scheduling problem of minimizing the total completion time on a two-machine permutation flowshop with setup times (denoted by $F_2 \mid ST_{si} \mid \sum C_j$). As soon as a machine $M_i (i = 1, 2)$ becomes available, a job $j (j = 1, \ldots, n)$ requires a sequence-independent setup time $s_{i,j}$ before being processed for $p_{i,j}$ units of time on that machine. An interesting database Internet connectivity application of the $F_2 \mid ST_{si} \mid \sum C_j$ can be found in [1].

Bagga and Khurana [3] were the first to address the $F_2 \mid ST_{si} \mid \sum C_j$ problem. They proposed two lower bounds, a dominance rule and a branch-and-bound algorithm which could only solve small-sized instances with up to 9 jobs. Allahverdi [2] implemented the lower bounds of [3], together with two new dominance rules in a branch-and-bound algorithm which was able to solve instances with up to 35 jobs. Moreover, he proposed three constructive heuristics which have been combined with local search procedures by Al-Anzi and Allahverdi [1]. These heuristics have been recently outperformed by those of Msakni et al. [5] who also proposed a genetic local search algorithm that provides near-optimal solutions in reasonable CPU time for large-sized instances with up to 500 jobs.

In the sequel, we briefly describe the lower bounds of Bagga and Khurana [3]. Then, we introduce new lower bounds for the $F_2 \mid ST_{si} \mid \sum C_j$ and provide computational results which show the good performance of the proposed lower bounds.

2 Lower bounds of Bagga and Khurana [3]

In this section, we describe the two lower bounds of Bagga and Khurana [3], denoted hereafter by $LB_1$ and $LB_2$. To the best of our knowledge, these are the only available lower bounds for $F_2 \mid ST_{si} \mid \sum C_j$.

If we relax the capacity of the second machine, then we get a single machine problem $1 \mid s_j, q_j \mid \sum C_j$ with setup times $s_j = s_{1,j}$ and tails $q_j = p_{2,j}$ ($j = 1, \ldots, n$). The optimal total completion time of the obtained problem is

---

1 Dr. Anis Gharbi and Dr. Talel Ladhari would like to thank Princess Fatimah Alnijris’s Research Chair for Advanced Manufacturing Technology for the provided financial support.
2 Email: a.gharbi@ksu.edu.sa
3 Email: talel_ladhari2004@yahoo.fr
4 Email: msakni_kais@yahoo.fr
5 Email: mehdi.serairi@hds.utc.fr
a valid lower bound for $F_2 | ST_{si} | \sum C_j$. Let $C_j^1$ denote the completion time of job $j$ in the SPT (optimal) sequence of the $1 || \sum C_j$ problem which is obtained after merging the setup time and the processing time of each job. We have $LB_1 = \sum_{j=1}^{n} C_j^1 + \sum_{j=1}^{n} q_j$.

In the computation of $LB_2$, it is assumed that $s_{1,j} = p_{1,j} = 0$. Consequently, a single machine problem $1 | s_j | \sum C_j$ with setup times $s_j = s_{2,j} (j = 1, ..., n)$ is obtained. Then, $LB_2$ is equal to the corresponding optimal total completion time which is obtained using the SPT rule after merging the setup time and the processing time of each job.

$LB_1$ and $LB_2$ can be computed in $O(n \log n)$ time.

3 New waiting time-based lower bounds

Let $\delta_j (j = 1, ..., n)$ denote the minimum amount of time that a job $j$ has to wait between its exit from $M_1$ and its starting time on $M_2$. Clearly, a valid lower bound for $F_2 | ST_{si} | \sum C_j$ is $LB_{WT} = \sum_{j=1}^{n} C_j^1 + \sum_{j=1}^{n} p_{2,j} + \sum_{j=1}^{n} \delta_j$.

In this section, we focus on deriving effective lower bounds on the total waiting time $\Delta = \sum_{j=1}^{n} \delta_j$. Note that trivially setting $\delta_j = 0$ for all $j = 1, ..., n$ yields the lower bound $LB_1$ of Bagga and Khurana [3]. Therefore, all the lower bounds which are proposed in this section dominate $LB_1$.

3.1 A zero predecessor-based lower bound

It is worth noting that the setup operation of each job $j$ on $M_2$ starts at least at the same time as its setup operation on $M_1$. Two cases are to be considered (see figure 1):

(i) $s_{2,j} \leq s_{1,j} + p_{1,j}$ : job $j$ will start processing on $M_2$ immmmediately after its completion on $M_1$. That is, $\delta_j = 0$.

(ii) $s_{2,j} > s_{1,j} + p_{1,j}$ : job $j$ has to wait until $M_2$ is prepared to process it. In this case, we have $\delta_j = s_{2,j} - s_{1,j} - p_{1,j}$.

![Fig. 1. Illustration of $\delta_j^1$ computation](image-url)
Consequently, a lower bound on \( \delta_j \) is \( \delta_j^1 = \max(s_{2,j} - s_{1,j} - p_{1,j}, 0) \). Therefore, a valid lower bound for the \( F_2 \mid ST_{si} \mid \sum C_j \) is \( LB_{WT}^1 = \sum_{j=1}^n C_j^1 + \sum_{j=1}^n p_{2,j} + \sum_{j=1}^n \delta_j^1 \).

\( LB_{WT}^1 \) can be computed in \( O(n \log n) \) time.

### 3.2 A one predecessor-based lower bound

In the computation of \( LB_{WT}^1 \), it is implicitly assumed that a job is not preceded by any job. However, this is only valid for the first scheduled job. For any other job \( j \), the setup operation on \( M_2 \) has to wait until the predecessor of job \( j \) finishes its processing. Consequently, if a job \( j \) is not the first scheduled job, then a better lower bound on \( \delta_j \) is

\[
\delta_j^2 = \max(0, s_{2,j} - s_{1,j} - p_{1,j} + \min_{h \neq j}(\delta_h^1 + p_{2,h}))
\]

(see figure 2). Therefore, if a particular job \( j_0 \) is the first one to be processed, then a lower bound on the total waiting time is \( \delta_{j_0}^1 + \sum_{j \neq j_0} \delta_j^2 \). Hence, a valid lower bound for the \( F_2 \mid ST_{si} \mid \sum C_j \) is \( LB_{WT}^2 = \sum_{j=1}^n C_j^1 + \sum_{j=1}^n p_{2,j} + \Delta' \), where \( \Delta' = \min_{j_0=1,\ldots,n}(\delta_{j_0}^1 + \sum_{j \neq j_0} \delta_j^2) \).

The computation of \( \Delta' \) requires \( O(n) \) time. Thus, \( LB_{WT}^2 \) can be computed in \( O(n \log n) \) time. Clearly, we have \( LB_{WT}^2 \geq LB_{WT}^1 \).

![Fig. 2. Illustration of \( \delta_j^2 \) computation](image_url)

### 3.3 A multiple predecessor-based lower bound

Let \( C_{i,j} \) denote the completion time of job \( j \) on machine \( M_i \) \((i = 1, 2)\). Actually, we have \( \delta_j = C_{2,j} - p_{2,j} - C_{1,j} \). Let \([k]\) denote the job in position \( k \), and \( \delta_j^{[k]} \) denote a lower bound on \( \delta_j \) if job \( j \) is scheduled at position \( k \). Obviously, \( \delta_j^{[1]} = \delta_j^1 \). Now, if job \( j \) is scheduled at position \( k \geq 2 \), then we have \( C_j^1 = C_{1,[k-1]}^1 + s_{1,j} + p_{1,j} \) and \( C_j^2 = \max(C_{2,[k-1]}^2 + s_{2,j}, C_j^1) + p_{2,j} \). Consequently, we have:

\[
\delta_j = C_j^2 - p_{2,j} - C_j^1
\]

\[
= \max(C_{2,[k-1]}^2 + s_{2,j}, C_j^1) - C_j^1
\]
\[
= \max(C^2_{[k-1]} + s_{2,j} - C^1_{j}, 0)
\]
\[
= \max(C^2_{[k-1]} - C^1_{[k-1]} + s_{2,j} - s_{1,j} - p_{1,j}, 0)
\]
\[
= \max(\delta_{[k-1]} + p_{2,[k-1]} + s_{2,j} - s_{1,j} - p_{1,j}, 0)
\]
\[
\geq \max(\min_{h \neq j}(\delta^{|k-1|}_h + p_{2,h}) + s_{2,j} - s_{1,j} - p_{1,j}, 0)
\]

Therefore, a valid lower bound on \(\delta_j\) if job \(j\) is scheduled at position \(k\) is \(\delta^{|k|}_j = \max(\min_{h \neq j}(\delta^{|k-1|}_h + p_{2,h}) + s_{2,j} - s_{1,j} - p_{1,j}, 0)\). Now, since each job has to be assigned exactly one position, then a better lower bound on \(\Delta\) can be obtained by solving the assignment problem where the cost of assigning job \(j\) to position \(k\) is \(\delta^{|k|}_j\). Let \(Z^*\) denote the minimum total assignment cost. Hence, the following lower bound for the \(F_2|ST_{si}|\sum C_j\) holds:

\[
\text{LB}^3_{WT} = \sum_{j=1}^{n} C^1_j + \sum_{j=1}^{n} p_{2,j} + Z^*. \text{Since } \delta^{|2|}_j = \delta^2_j, \text{ then } \text{LB}^3_{WT} \geq \text{LB}^2_{WT}.
\]

Using a dynamic programming algorithm, the computation of \(\delta^{|k|}_j\) for all \(j, k = 1, ..., n\) can be performed in \(O(n^2)\) time. The computation of \(Z^*\) requires \(O(n^3)\) time. Therefore, \(\text{LB}^3_{WT}\) can be computed in \(O(n^3)\) time.

| Data | Computation of \(\delta^{|k|}_j\) |
|------|---------------------------------|
| \(j\) | \(s_{1,j}\) | \(p_{1,j}\) | \(s_{2,j}\) | \(p_{2,j}\) | \(\delta^{|1|}_j\) | \(\delta^{|2|}_j\) | \(\delta^{|2|}_h + p_{2,h}\) | \(\delta^{|3|}_j\) | \(\delta^{|3|}_h + p_{2,h}\) | \(\delta^{|4|}_j\) |
| 1    | 3    | 4    | 8    | 7     | 1     | 5     | 12    | 9     | 16    | 9     |
| 2    | 7    | 5    | 14   | 2     | 2     | 7     | 9     | 10    | 12    | 10    |
| 3    | 11   | 7    | 2    | 8     | 0     | 0     | 8     | 0     | 8     | 0     |
| 4    | 4    | 8    | 12   | 5     | 0     | 4     | 9     | 8     | 3     | 8     |

Table 1
Illustration of \(\text{LB}^3_{WT}\) computation

**Example:** Consider the four-job instance which data is described in Table 1. We have \(Z^* = \delta^{|1|}_2 + \delta^{|1|}_1 + \delta^{|3|}_4 + \delta^{|4|}_3 = 15\) and \(\text{LB}^3_{WT} = 106 + 22 + 15 = 143\). Note that \(\text{LB}_1 = 128\), \(\text{LB}^2_{WT} = 131\) and \(\text{LB}^2_{WT} = 139\).

### 4 New single machine-based lower bounds

In this section, we assume that the capacity of the first machine is relaxed. The obtained problem is a single machine problem \(1|s_j, r_j|\sum C_j\) with setup times \(s_j = s_{2,j}\) and release dates \(r_j = s_{1,j} + p_{1,j}\). Unfortunately, the \(1|s_j, r_j|\sum C_j\) is \(\mathcal{NP}\)-hard, since the \(1|r_j|\sum C_j\) is known to be \(\mathcal{NP}\)-hard. However, any lower bound for the \(1|s_j, r_j|\sum C_j\) is a valid lower bound for
the $F2 | ST_{si} | \sum C_j$. Note that the lower bound $LB_2$ proposed by Bagga and Khurana [3] constitutes a trivial lower bound for the $1 | s_j, r_j | \sum C_j$. In the sequel, we provide an equivalence result for the single machine problem with setup times and release dates, and show how to derive lower bounds for the $1 | s_j, r_j | \sum C_j$ which dominate $LB_2$.

**Theorem 4.1** $1|r_j, s_j|f$ and $1 | r_j | f$ are equivalent

**Proof.** Obviously, any $1 | r_j | f$ instance can be stated as a $1|r_j, s_j|f$ one with zero setup times. Now, we show that a $1|r_j, s_j|f$ instance can be transformed into a $1 | r_j | f$ one. For that purpose, let $p'_j = p_j + s_j$ and $r'_j = \max(r_j - s_j, 0)$ for all $j = 1, ..., n$. Now, consider the following nonlinear mathematical formulation of the $1 | r_j, s_j | f$.

- $x_{ij} = \begin{cases} 1 & \text{if job } j \text{ is processed at position } i \\ 0 & \text{if not} \end{cases}$
- $C_i$ the completion time of the job which is sequenced at position $i$

\[
\begin{align*}
\text{Minimize} & \quad f(C_1, \ldots, C_n) \\
\sum_{i=1}^{n} x_{ij} &= 1 \quad j = 1, \ldots, n \\
\sum_{j=1}^{n} x_{ij} &= 1 \quad i = 1, \ldots, n \\
C_1 &= \sum_{j=1}^{n} x_{1j}(\max(r_j, s_j) + p_j) \\
C_i &= \sum_{j=1}^{n} x_{ij}(\max(r_j, C_{i-1} + s_j) + p_j) \quad i = 2, \ldots, n \\
x_{ij} &= \{0, 1\} \quad i, j = 1, \ldots, n \\
C_i &\geq 0 \quad i = 1, \ldots, n
\end{align*}
\]

We have:

\[
C_1 = \sum_{j=1}^{n} x_{1j}(\max(r_j, s_j) + p_j) = \sum_{j=1}^{n} x_{1j}(\max(r_j - s_j, 0) + p'_j)
\]
Similarly, we have:

$$C_i = \sum_{j=1}^{n} x_{ij}(\max(r_j, C_{i-1} + s_j) + p_j) = \sum_{j=1}^{n} x_{ij}(\max(r_j - s_j, C_{i-1}) + p'_j)$$  

(9)

Since $C_{i-1} \geq 0$, then we have:

$$C_i = \sum_{j=1}^{n} x_{ij}(\max(0, r_j - s_j, C_{i-1}) + p'_j)$$

$$= \sum_{j=1}^{n} x_{ij}(\max(\max(0, r_j - s_j), C_{i-1}) + p'_j)$$  

(10)

Therefore, if constraints 4 and 5 are replaced by 8 and 10, respectively, then we obtain an equivalent 1|r_j|f formulation with processing times $p'_j$ and release dates $r'_j$.

\[ \square \]

**Corollary 4.2** 1|r_j, s_j| \( \sum \) \( C_j \) and 1|r_j| \( \sum \) \( C_j \) are equivalent

Consequently, valid lower bounds for the 1 | r_j, s_j | \( \sum \) \( C_j \) can be derived by using the above transformation. A first single machine-based lower bound for F2 | STsi | \( \sum \) \( C_j \), denoted hereafter by LB_{SM}^1, can be derived by computing the optimal total completion time of the preemptive version of the obtained 1 | r_j | \( \sum \) \( C_j \) problem. LB_{SM}^1 can be computed in \( O(n \log n) \) by using the SRPT rule.

Interestingly, T’Kindt and Della Croce [4] proposed an improved preemptive lower bound for the 1 | r_j | \( \sum \) \( C_j \) which can be computed in \( O(n \log n) \) time. Clearly, their result can be used to derive a second single machine-based lower bound for the F2 | STsi | \( \sum \) \( C_j \), denoted hereafter by LB_{SM}^2. Obviously, we have LB_{SM}^2 \geq LB_{SM}^1.

5 Preliminary computational results

In order to assess the quality of the different proposed lower bounds, we carried out a series of experiments on randomly generated instances. The processing and setup times were randomly generated from uniform distributions with $p_{i,j}$ from [1, 100] and $s_{i,j}$ from [0, 100K], where $K \in \{0.25, 0.5, 0.75, 1\}$. The number of jobs $n$ was taken equal to 10, 30, 50, 70 and 100. Fifty replicates were generated for each problem size and each value of $K$.

Table 2 reports, for each lower bound, the percentage of times it yields the maximal value over all of the bounds described in this paper. It is clear that
the lower bounds of Bagga and Khurana [3] are largely outperformed by our proposed lower bounds. The highest performance being achieved by $LB_2^{WT}$ and $LB_3^{WT}$.

### Table 2
Performance of the lower bounds

<table>
<thead>
<tr>
<th>$n$</th>
<th>$LB_1$</th>
<th>$LB_2$</th>
<th>$LB_1^{WT}$</th>
<th>$LB_2^{WT}$</th>
<th>$LB_3^{WT}$</th>
<th>$LB_{1 SM}$</th>
<th>$LB_{2 SM}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>29.5%</td>
<td>0.0%</td>
<td>50.5%</td>
<td>67.0%</td>
<td>70.0%</td>
<td>8.5%</td>
<td>30.0%</td>
</tr>
<tr>
<td>30</td>
<td>7.0%</td>
<td>0.0%</td>
<td>18.0%</td>
<td>63.5%</td>
<td>65.5%</td>
<td>7.5%</td>
<td>34.5%</td>
</tr>
<tr>
<td>50</td>
<td>3.0%</td>
<td>0.0%</td>
<td>7.5%</td>
<td>53.0%</td>
<td>54.5%</td>
<td>11.0%</td>
<td>45.5%</td>
</tr>
<tr>
<td>70</td>
<td>1.0%</td>
<td>0.0%</td>
<td>4.0%</td>
<td>60.5%</td>
<td>60.5%</td>
<td>8.5%</td>
<td>39.5%</td>
</tr>
<tr>
<td>100</td>
<td>0.5%</td>
<td>0.0%</td>
<td>1.5%</td>
<td>59.5%</td>
<td>60.5%</td>
<td>9.0%</td>
<td>39.5%</td>
</tr>
</tbody>
</table>

| $K$  | 0.25  | 17.6%  | 0.0%        | 28.4%       | 60.4%       | 60.8%       | 8.4%        | 39.2%       |
|      | 0.50  | 8.8%   | 0.0%        | 18.4%       | 62.8%       | 64.4%       | 8.8%        | 35.6%       |
|      | 0.75  | 4.8%   | 0.0%        | 10.4%       | 58.4%       | 60.0%       | 8.0%        | 40.0%       |
|      | 1.00  | 1.6%   | 0.0%        | 8.0%        | 61.2%       | 63.6%       | 10.4%       | 36.4%       |

References


