Strongly Fully Polynomial Time Approximation Scheme for the two-parallel capacitated machines scheduling problem

Imed Kacem*, Yann Lanuel and Myriam Sahnoune

Universite Paul Verlaine – Metz,
Laboratoire d’Informatique Théorique et Appliquée (LITA),
UFR M.I.M, Ile du Saulcy 57000, Metz, France
E-mail: kacem@univ-metz.fr
E-mail: yann.lanuel@univ-metz.fr
E-mail: sahnoune@univ-metz.fr
*Corresponding author

Abstract: We study the n-job two-parallel machines scheduling problem with the aim of minimising the total flow-time. In this problem, instead of allowing both machines to be continuously available as it is often assumed in the literature, we consider that one of the machines is available for a specified period of time after which it can no longer process any job. On the basis of the modification of an exact algorithm execution, we establish the existence of a strongly Fully Polynomial Time Approximation Scheme (FPTAS) for the above-mentioned problem.

Keywords: approximation; scheduling; FPTAS; fully polynomial time approximation scheme; heuristic; availability constraint.

Reference to this paper should be made as follows: Kacem, I., Lanuel, Y. and Sahnoune, M. (2011) ‘Strongly Fully Polynomial Time Approximation Scheme for the two-parallel capacitated machines scheduling problem’, Int. J. Planning and Scheduling, Vol. 1, Nos. 1/2, pp.32–41.

Biographical notes: Imed Kacem received his Diploma of ‘INGENIEUR’ from ENSAIT (French Grande Ecole) and MS from Lille 1 University, both in 2000; PhD in Computer Science from the Ecole Centrale de Lille in 2003 and his Habilitation Degree (HDR) from Paris-Dauphine University in 2007. He is a Full Professor at Paul Verlaine University of Metz (UPVM) and the Head of its Computer Science Department. His research interests include combinatorial optimisation and scheduling. He is the author of several publications in refereed journals, conferences, books and chapters of books. He is an Area Editor for Computers & Industrial Engineering. He is on the editorial boards of EJIE, IJAOM, IJPS, Advances in OR, IJAMC, JSCI, JPAM and JISE. He was the Organisation Chairman of CIE39, ICSSSM06 and WAC/ISIAC06. He is listed in Who’s Who in the World.

Yann Lanuel received his BS and MS from Metz University and PhD in Computer Science from the Metz University in 1994. He is an Associate
1 Introduction

Scheduling problems with unavailability constraints have attracted numerous researchers from all over the world. This has been motivated by practical and real industrial problems (maintenance tasks, non-availability of resources ...). It is noteworthy that during the last decade numerous problems of this class have been addressed in the literature (for more details, see the state-of-the-art paper by Schmidt (2000)). However, to the best of our knowledge, the studied problem in this paper has only been addressed in few references. That is why this paper is a good attempt to solve this problem by establishing the existence of a strongly FPTAS for the above-mentioned problem.

More precisely, the studied problem is the \(n\)-job two-parallel machines scheduling problem with the aim of minimising the total flow-time. In this problem, instead of allowing both machines to be continuously available as it is often assumed in the literature, we consider that one of the machines is available for a specified period of time after which it can no longer process any job. Lee and Liman (1993) proposed a heuristic that has a worst-case error bound of 1/2. Recently, Liao et al. (2009) proposed upper and lower bounds and elaborated a branch-and-bound algorithm to solve the problem. Other general approximation approaches can be applied to the studied problem (see, for example, Kovalyov and Kubiak (1999b) or Woeginger (2000)) for obtaining an FPTAS. However, the time complexities of all of them are not strongly polynomial.

For self-consistency, we recall some necessary definitions related to the approximation area. A \(\rho\)-approximation algorithm for a problem of minimising an objective function \(\varphi\) is an algorithm such that for every instance \(\pi\) of the problem it gives a solution \(S_\pi\) verifying \(\varphi(S_\pi)/\varphi(OPT_\pi) \leq \rho\) where \(OPT_\pi\) is the optimal solution of \(\pi\). Also, \(\rho\) is called the worst-case bound of the above-mentioned algorithm. The approximation is tight if \(\rho\) is the best possible (i.e., the smallest value we can obtain by the algorithm for all the instances of the problem). A class of \((1 + \varepsilon)\)-approximation algorithms is an FPTAS, if its running time is bounded by a polynomial function in \(1/\varepsilon\) and the instance size for every \(\varepsilon > 0\). A class of \((1 + \varepsilon)\)-approximation algorithms is a Polynomial-Time Approximation Scheme (PTAS), if its running time is an arbitrary function in \(1/\varepsilon\) and the instance size for every \(\varepsilon > 0\).
The paper is organised as follows. Section 2 describes the problem. Then, a dynamic programming algorithm is presented in Section 3. Finally, we prove the existence of an FPTAS for the total completion time minimisation problem in Section 4. Finally, Section 5 concludes the paper.

2 Flow-time minimisation on capacitated two-parallel machines

The problem is to schedule $n$ jobs on two parallel machines, with the aim of minimising the total completion time (i.e., the total flow-time since all the jobs are ready for processing at time 0). Every job $i$ has a processing time $p_i$. The first machine is available for a specified period of time $[0, T_1]$ (i.e., after $T_1$, it can no longer process any job). Every machine can process at most one job at a time. Without loss of generality, we consider that all data are integers and that jobs are indexed according to the Shortest Processing Time (SPT) rule:

$$p_1 \leq p_2 \leq \cdots \leq p_n.$$  \hfill (1)

Because of the dominance of the SPT order, an optimal solution is composed of two sequences (one sequence for each machine) of jobs scheduled in non-decreasing order of their indexes (Smith, 1956). Figure 1 illustrates a feasible schedule for a 7-job instance.

**Figure 1** Problem illustration

<table>
<thead>
<tr>
<th>Machine 1</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$T_1$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Machine 2</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

In the remainder of the paper, $Q$ denotes the studied problem, $F^*(Q)$ denotes the minimal sum of the completion times for problem $Q$ and $F_S(Q)$ is the sum of the completion times of schedule $S$ for problem $Q$.

**Proposition 1:** If $\sum_{i=1}^n p_i \leq 2T_1$, then problem $(Q)$ can be polynomially solved.

**Proof:** We relax the unavailability constraint (i.e., we assume that the first machine is continuously available). Then, the relaxed problem has an SPT rule optimal solution according to Smith (1956). Let $\sigma_1$ be the obtained schedule by applying SPT for the relaxed problem, $B'_1$ be the completion time of the last job scheduled on the first machine and $B'_2$ denote the completion time of the last job scheduled on the second machine. By assumption, $B'_1 + B'_2 \leq 2T_1$. Hence, either $B'_1 \leq T_1$ or
$B'_2 \leq T_1$ must hold. If $B'_1 \leq T_1$, then $\sigma_1$ is also optimal for the original problem $Q$. If $B'_2 \leq T_1$, then by exchanging the role of the two machines, we obtain a new optimal schedule $\sigma'_1$ that is also feasible for the original problem $Q$ and this completes the proof.

On the basis of the result of Proposition 1, we only consider the case where

$$\sum_{i=1}^{n} p_i > 2T_1. \quad (2)$$

3 Dynamic programming algorithm

The problem can be optimally solved by applying the following standard dynamic programming algorithm $B$. This algorithm generates iteratively some sets of states. At every iteration $k$, a set $U_k$ composed of states is generated ($0 \leq k \leq n$). Each state $[t, f]$ in $U_k$ can be associated to a feasible schedule for the first $k$ jobs. Variable $t$ denotes the completion time of the last job scheduled on the first machine before $T_1$ and $f$ is the total flow-time of the corresponding schedule. This algorithm can be described as follows:

Algorithm $B$

(i) Set $U_0 := \{[0, 0]\}$.

(ii) For $k \in \{1, 2, \ldots, n\}$,

Set $U_k := \emptyset$

For every state $[t, f]$ in $U_{k-1}$:

1) Put $[t, f + (\sum_{i=1}^{k} p_i - t)]$ in $U_k$ (i.e., job $k$ is scheduled on $M_2$)

2) Put $[t + p_k, f + (t + p_k)]$ in $U_k$ if $t + p_k \leq T_1$ (i.e., job $k$ is scheduled on $M_1$)

Remove $U_{k-1}$

(iii) $F^*(Q) := \min \{f \mid [t, f] \in U_n\}$.

The analysis of this dynamic algorithm shows that its complexity can be bounded by $O(nT_1Z)$ where $Z$ is an upper bound (on the optimal flow-time) obtained by Heuristic $H$ (described in the next section). This complexity can be reduced to $O(nT_1)$ if at every step $k$ in the algorithm, we only keep for $t$ one state $[t, f]$ with the smallest value of $f$.

4 FPTAS

4.1 Description

Here, we are interested in the existence of an FPTAS for this problem. We present a new algorithm for this problem based on a trimming technique.
The algorithm starts by applying a simple heuristic $H$ to obtain a feasible solution in $O(n \log(n))$ time. This heuristic consists in scheduling all the jobs on the second machine in the SPT order. Obviously, the following inequality holds:

$$\frac{F_H(Q)}{F^*(Q)} \leq 2. \quad (3)$$

The last inequality can be obtained by comparing $F_H(Q)$ with the optimal solution of the relaxation in which the first machine is continuously available and every job is split into two equal pieces.

In the second step of our FPTAS, we modify the execution of algorithm $B$ to reduce the running time. The approach of modifying the execution of an exact algorithm to design FPTAS was initially proposed by Ibarra and Kim (1975) for solving the knapsack problem. It is noteworthy that during the last decades numerous scheduling problems have been addressed by applying such an approach. A sample of these papers includes Gens and Levner (1981), Kacem (2009), Kacem (2010), Kacem and Kellerer (2011), Kacem and Mahjoub (2009), Kacem and Haouari (2009), Sahni (1976), Kovalyov and Kubiak (1999a), Kellerer and Strusevich (2006) and Woeginger (2000).

Given an arbitrary $\varepsilon > 0$, we define

$$q = \left\lceil \frac{4n}{\varepsilon} \right\rceil, \quad (4)$$

$$\delta_1 = \frac{F_H(Q)}{q} \quad (5)$$

and

$$\delta_{2,k} = \min \left\{ \frac{T_1, \sum_{h=1}^{k} p_h}{q} \right\} \quad \forall k = 1, 2, \ldots, n. \quad (6)$$

We split interval $[0, F_H(Q)]$ into $q$ subintervals $I_r^1 = [(r-1)\delta_1, r\delta_1]_{1 \leq r \leq q}$ of length $\delta_1$. We also split interval $[0, \min\{T_1, \sum_{h=1}^{k} p_h\}]$ into $q$ equal subintervals $I_{s,k}^2 = [(s-1)\delta_{2,k}, s\delta_{2,k}]_{1 \leq s \leq q}$ of length $\delta_{2,k}$ at every iteration $k$. Our algorithm $B'_\varepsilon$ generates reduced sets $\tilde{U}_k$ instead of sets $U_k$. The algorithm can be described as follows:

**Algorithm $B'_\varepsilon$**

(i) Set $\tilde{U}_0 := \{[0,0]\}$.

(ii) For $k \in \{1, 2, 3, \ldots, n\}$,

   Set $\tilde{U}_k := \emptyset$

   For every state $[t, f]$ in $\tilde{U}_{k-1}$:

   1) Put $[t, f + (\sum_{i=1}^{k} p_i - t)]$ in $\tilde{U}_k$

   2) Put $[t + p_k, f + (t + p_k)]$ in $\tilde{U}_k$ if $t + p_k \leq T_1$.
Remove $\widetilde{U}_{k-1}$

Let $[t, f]_{r, s}$ be the state in $\widetilde{U}_k$ such that $f \in I^1_r$ and $t \in I^2_{s, k}$ with the smallest possible $t$ (ties are broken by choosing the state of the smallest $f$).

Set $\widetilde{U}_k := \{[t, f]_{r, s} | 1 \leq r \leq q, 1 \leq s \leq q\}$.

(iii) $\mathcal{F}_{B'_1}(\mathcal{Q}) := \min \{f \mid [t, f] \in \widetilde{U}_n\}$.

Note that the added step in Algorithm $B'_1$ with respect to Algorithm $B$ is to ensure the feasibility of solutions owing to the capacity constraint on the first machine.

The worst-case analysis of this algorithm and the comparison of the execution of algorithms $B$ and $B'_1$ allow us to deduce the following theorem.

**Theorem 2:** Algorithm $B'_1$ is an FPTAS and it can be implemented in $O(n^3/\varepsilon^2)$ time.

4.2 Proof of the theorem

The proof is based on the following two lemmas.

**Lemma 3:** Algorithm $B'_1$ can be implemented in $O(n^3/\varepsilon^2)$ time.

**Proof:** Heuristic $H$ can be computed in $O(n \log(n))$ time. From the construction of Algorithm $B'_1$, at each iteration $k$ ($k \in \{1, 2, \ldots, n\}$) we have $|\widetilde{U}_k| \leq q^2 \leq \left(\frac{4n}{\varepsilon} + 1\right)^2$. Hence,

$$\sum_{k=1}^{n} |\widetilde{U}_k| \leq n \left(\frac{4n}{\varepsilon} + 1\right)^2.$$  \(\square\)

In conclusion, the algorithm can be implemented in $O(n \log(n) + n^3/\varepsilon^2)$ time.

**Lemma 4:** Let $\eta_k = \sum_{h=1}^{k} \delta_{2,h}$. For every state $[t, f] \in U_k$ ($k \in \{0, 1, \ldots, n\}$), Algorithm $B'_1$ generates at least one state $[\tilde{t}, \tilde{f}] \in \widetilde{U}_k$ such that:

$$\tilde{t} \leq t,$$

and

$$\tilde{f} \leq f + k \delta_1 + \sum_{z=1}^{k} \eta_z.$$  \(\square\)

**Proof:** We prove the result by induction. For $k = 0$, $\widetilde{U}_0 = U_0$ and the lemma is obvious. Let us assume the result holds until level $k - 1$ and let us prove it for level $k$. Consider a state $[t, f] \in U_k$. Two cases can be distinguished:

1st case: $[t, f] = [t', f'] + \left(\sum_{i=1}^{k} p_i - t'\right)$ where $[t', f'] \in U_{k-1}$. State $[t', f']$ belongs to $U_{k-1}$. This implies the existence of $[\tilde{t}', \tilde{f}'] \in \widetilde{U}_{k-1}$ such that $\tilde{t}' \leq t'$, and
\[
\tilde{f}' \leq f' + (k - 1)\delta_1 + \sum_{z=1}^{k-1} \eta_z. \quad \text{Hence, the construction of state } [\tilde{t}', \tilde{f}' + (\sum_{i=1}^{k} p_i - \tilde{t}')] \text{ is considered at iteration } k. \text{ Such a state can be eliminated and replaced by an approximate one } [\theta_1, \theta_2] \text{ such that }
\]
\[
\theta_1 \leq \tilde{t}' \leq t',
\]
and
\[
\theta_2 \leq f' + \left( \sum_{i=1}^{k} p_i - \tilde{t}' \right) + \delta_1
\]
\[
\leq f' + (k - 1)\delta_1 + \sum_{z=1}^{k-1} \eta_z + \left( \sum_{i=1}^{k} p_i - \tilde{t}' \right) + \delta_1
\]
\[
\leq f' + k\delta_1 + \sum_{z=1}^{k-1} \eta_z + \left( \sum_{i=1}^{k} p_i - \tilde{t}' \right)
\]
\[
< f' + k\delta_1 + \sum_{z=1}^{k-1} \eta_z + \left( \sum_{i=1}^{k} p_i - \tilde{t}' \right) + t' - t'
\]
\[
= f' + k\delta_1 + \left( \sum_{i=1}^{k} p_i - t' \right) + t' - \tilde{t}' + \sum_{z=1}^{k-1} \eta_z
\]
\[
= f + k\delta_1 + t' - \tilde{t}' + \sum_{z=1}^{k-1} \eta_z.
\]

It is easy to see that \( t' - \tilde{t}' \leq \sum_{h=1}^{k} \delta_{2,h} = \eta_k. \) Then, \( \theta_2 \leq f + k\delta_1 + \sum_{z=1}^{k} \eta_z. \) Hence, \([\theta_1, \theta_2]\) is an approximate state verifying the two conditions.

2nd case: \([t, f] = [t' + p_k, f' + (t' + p_k)]\) where \([t', f'] \in \mathcal{U}_{k-1}. \) State \([t', f']\) belongs to \( \mathcal{U}_{k-1}. \) This implies the existence of \([\tilde{t}', \tilde{f}'] \in \mathcal{U}_{k-1} \) such that \( \tilde{t}' \leq t' \) and \( \tilde{f}' \leq f' + (k - 1)\delta_1 + \sum_{z=1}^{k-1} \eta_z. \) Hence, the construction of state \([\tilde{t}' + p_k, \tilde{f}' + (\tilde{t}' + p_k)]\) is considered at iteration \( k. \) Such a state can be eliminated and replaced by an approximate one \((\theta_1', \theta_2')\) such that
\[
\theta_1' \leq \tilde{t}' + p_k \leq t' + p_k = t,
\]
and
\[
\theta_2' \leq \tilde{f}' + (\tilde{t}' + p_k) + \delta_1
\]
\[
\leq f' + (k - 1)\delta_1 + \sum_{z=1}^{k-1} \eta_z + (\tilde{t}' + p_k) + \delta_1
\]
\[
\leq f' + (k - 1)\delta_1 + \sum_{z=1}^{k-1} \eta_z + (t' + p_k) + \delta_1
\]
\[\leq f' + (t' + p_k) + k\delta_1 + \sum_{z=1}^{k-1} \eta_z\]

\[= f + k\delta_1 + \sum_{z=1}^{k-1} \eta_z < f + k\delta_1 + \sum_{z=1}^{k} \eta_z.\]

Hence, \([\theta'_1, \theta'_2]\) is an approximate state verifying the two conditions.

In the two cases, the lemma holds. \(\square\)

From Lemma 3, the complexity of \(B'_\varepsilon\) is strongly polynomial. Let \([t^*, f^*]\) be a state from \(U_n\) associated to the optimal solution. Hence, from Lemma 4, there exists \([\tilde{t}, \tilde{f}]\in U_n\) such that

\[\tilde{f} \leq f^* + n\delta_1 + \sum_{z=1}^{n} \eta_z\]

Hence,

\[\tilde{f} \leq F^*(Q) + n \frac{F_H(Q)}{4} + \sum_{z=1}^{n} \sum_{h=1}^{\min\{T_1, \sum_{k=1}^{h} p_k\}} \delta_{2,h}\]

\[\leq F^*(Q) + n \frac{F_H(Q)}{4} + \sum_{z=1}^{n} \sum_{h=1}^{\sum_{k=1}^{h} p_k} \frac{\delta_{2,h}}{4n}\]

\[< F^*(Q) + \varepsilon \frac{F_H(Q)}{4} + \sum_{z=1}^{n} \sum_{h=1}^{\sum_{k=1}^{h} p_k} \frac{\delta_{2,h}}{4n}\]

\[< F^*(Q) + \varepsilon \frac{F^*(Q)}{2} + \sum_{z=1}^{n} \sum_{h=1}^{\sum_{k=1}^{h} p_k} \frac{\delta_{2,h}}{4n}\]

From the definition of the SPT order, we deduce

\[\tilde{f} \leq F^*(Q) + \varepsilon \frac{F^*(Q)}{2} + \sum_{z=1}^{n} \sum_{h=1}^{\sum_{k=1}^{h} p_k} \frac{\delta_{2,h}}{4n}\]

\[\leq F^*(Q) + \varepsilon \frac{F^*(Q)}{2} + \sum_{z=1}^{n} \sum_{h=1}^{\sum_{k=1}^{h} p_k} \frac{\delta_{2,h}}{4n}\]

\[< F^*(Q) + \varepsilon \frac{F^*(Q)}{2} + \sum_{z=1}^{n} \sum_{h=1}^{\sum_{k=1}^{h} p_k} \frac{\delta_{2,h}}{4n}\].
Note that $\mathcal{F}_H(Q) = np_1 + (n-1)p_2 + (n-2)p_3 + \cdots + 1.p_n$. Thus,

$$\tilde{f} \leq \mathcal{F}^*(Q) + \epsilon \frac{\mathcal{F}^*(Q)}{2} + \epsilon \frac{\mathcal{F}_H(Q)}{4} \leq (1 + \epsilon)\mathcal{F}^*(Q).$$

Since $\mathcal{F}_{B_1}(Q) \leq \tilde{f}$, therefore $B_1'$ provides $(1 + \epsilon)$-schedule in a strongly polynomial time and this completes the proof.

5 Conclusion

In this paper, we deal with a scheduling problem and we propose an FPTAS for solving it. Such a problem consists in sequencing $n$ jobs on two parallel machines with the aim of minimising the total flow-time. In this problem, instead of allowing both machines to be continuously available as it is often assumed in the literature, we consider that one of the machines is available for a specified period of time after which it can no longer process any job. In this paper, we establish the existence of an FPTAS for the above-mentioned problem. The complexity of the proposed FPTAS is strongly polynomial.

In our future works, we hope to extend these results to the weighted version of this problem. The development of better approximation algorithms by combining our approach with other techniques ((Kovalyov and Kubiak, 1999b) for example) is also a challenging subject.

References


