Hao ZHOU, Liping CAO, Qi WEI, Zhenyu SHU, Yiwei JIANG, 2025. An optimal algorithm for preemptive scheduling on non-simultaneously available uniform machines. *Frontiers of Information Technology & Electronic Engineering,* 26(3):472-478. https://doi.org/10.1631/FITEE.2300767

An optimal algorithm for preemptive scheduling on non-simultaneously available uniform machines

Key words: Optimal algorithm; Preemptive scheduling; Uniform machine; Non-simultaneous available times

Corresponding author: Yiwei JIANG
E-mail: ywjiang@zjgsu.edu.cn
ORCID: https://orcid.org/0000-0002-9779-0613

Motivation

1. Unlike classical parallel-machine scheduling where all the machines are available simultaneously at time zero, the machines in our problem may not be available at time zero.
2. This model has many applications in real industry settings; e.g., machines may require different setup times or warm-up times before they start processing jobs due to preventive maintenance and adjustment requirements.

Main idea

1. We first convert all the *m* uniform machines to *m* virtual machines such that, at any time, the earlier the available time of a virtual machine, the greater its speed.
2. Then we schedule the current job so as to make a virtual machine with as small a completion time as possible to be a full machine.

Method

1. Machine conversion

We convert all the *m* uniform machines to *m* virtual machines whose speed is defined as follows:

$$v_i(t)=\begin{cases}0, & 0<t\le r_i,\\ v_{ji}, & r_j<t\le r_{j+1},\ j=i,i+1,\cdots,m-1,\\ v_{mi}, & t>r_m.\end{cases}$$

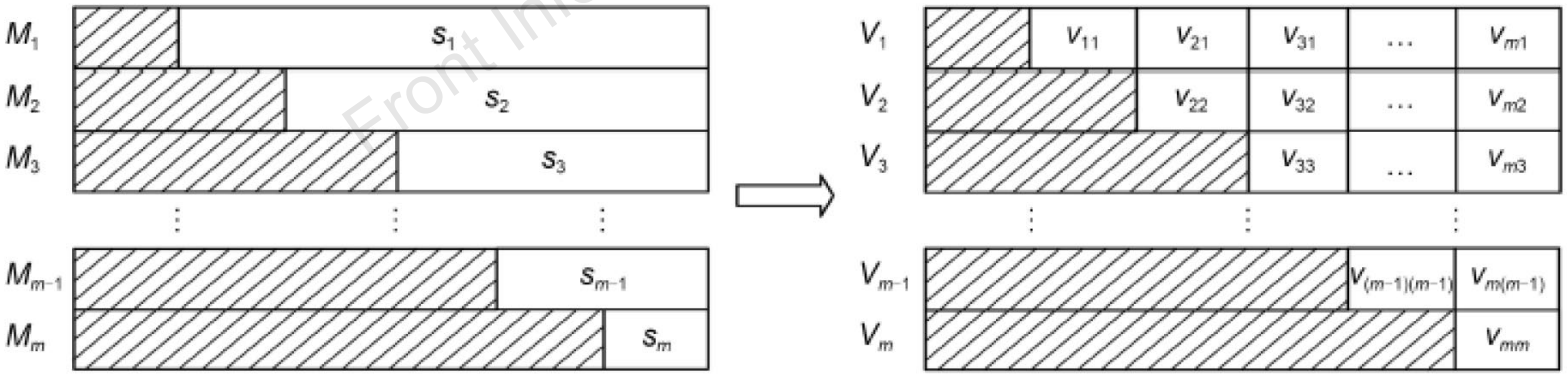


Fig. 1 Converting from real machines to virtual machines

Method (Cont'd)

2. Job schedule:

Schedule the current job so as to make a virtual machine with as small a completion time as possible to be a full machine.

Algorithm 1 Optimal schedule

// Step 1

Let $l_i=r_i(i=1,2,\cdots,m)$, $S_{\mathrm{NFVM}}=\{V_1,V_2,\cdots,V_m\}$, and $q_{\mathrm{NFVM}}=m$.

// Step 2

For any $p_j, j=1,2,\cdots,n$.

// Step 3

If $p_j \le W_{q_{\mathrm{NFVM}}}$

Schedule p_j on virtual machine $V_{q_{\mathrm{NFVM}}}$ as early as possible.

Find a time T such that

$$\int_{l_{q_{\mathrm{NFVM}}}}^{T} v_{q_{\mathrm{NFVM}}}(t)\,\mathrm{d}t=p_j.$$

If $T=w_Q$

Update $S_{\mathrm{NFVM}}=S_{\mathrm{NFVM}}\backslash\{V_{q_{\mathrm{NFVM}}}\}$ and

$q_{\mathrm{NFVM}}=\max_{V_i\in S_{\mathrm{NFVM}}}\{i\}$; $j=j+1$, go back to Step 2.

Else

Update $l_{q_{\mathrm{NFVM}}}=T$ and $W_{q_{\mathrm{NFVM}}}=W_{q_{\mathrm{NFVM}}}-p_j$;

$j=j+1$, go back to Step 2.

End if

// Step 4

Else

Find two adjacent virtual machines V_k and V_h in S_{NFVM} such that $W_k<p_j\le W_h$.

If $\int_{l_h}^{l_h} v_h(t)\,\mathrm{d}t+\int_{l_k}^{w_Q} v_k(t)\,\mathrm{d}t\ge p_j$

Partition p_j into two parts $p_j^{(1)}$ and $p_j^{(2)}$ such that

$p_j^{(1)}=W_k$ and $p_j^{(2)}=p_j-W_k$.

Find a time T such that $p_j^{(2)}=\int_{l_h}^{T} v_h(t)\,\mathrm{d}t$.

Schedule $p_j^{(1)}$ on V_k in the time interval $[l_k, w_Q)$ and $p_j^{(2)}$ on V_h in $[l_h, T)$.

Update $S_{\mathrm{NFVM}}=S_{\mathrm{NFVM}}\backslash\{V_k\}$, $q_{\mathrm{NFVM}}=\max_{V_i\in S_{\mathrm{NFVM}}}\{i\}$,

$l_h=T$, and $W_h=W_h+W_k-p_j$; $j=j+1$, go back to Step 2.

// Step 5

Else

Find a time T such that

$$\int_{l_h}^{T} v_h(t)\,\mathrm{d}t+\int_{T}^{w_Q} v_k(t)\,\mathrm{d}t=p_j.$$

Partition p_j into two parts $p_j^{(1)}$ and $p_j^{(2)}$ such that

$$p_j^{(1)}=\int_{T}^{w_Q} v_k(t)\,\mathrm{d}t \text{ and } p_j^{(2)}=p_j-p_j^{(1)}=\int_{l_h}^{T} v_h(t)\,\mathrm{d}t.$$

Schedule $p_j^{(1)}$ on V_k in the time interval $[T, w_Q)$ and $p_j^{(2)}$ on V_h in $[l_h, T)$. Change the time slot $[l_k, T]$ of the two virtual machines; we have a new virtual machine V_h with $l_h=l_k$ and a new "full" virtual machine V_k.

Update $S_{\mathrm{NFVM}}=S_{\mathrm{NFVM}}\backslash\{V_k\}$, $q_{\mathrm{NFVM}}=\max_{V_i\in S_{\mathrm{NFVM}}}\{i\}$,

$l_h=l_k$, and $W_h=W_h+W_k-p_j$; $j=j+1$, go back to Step 2.

End if

End if

Algorithm illustration

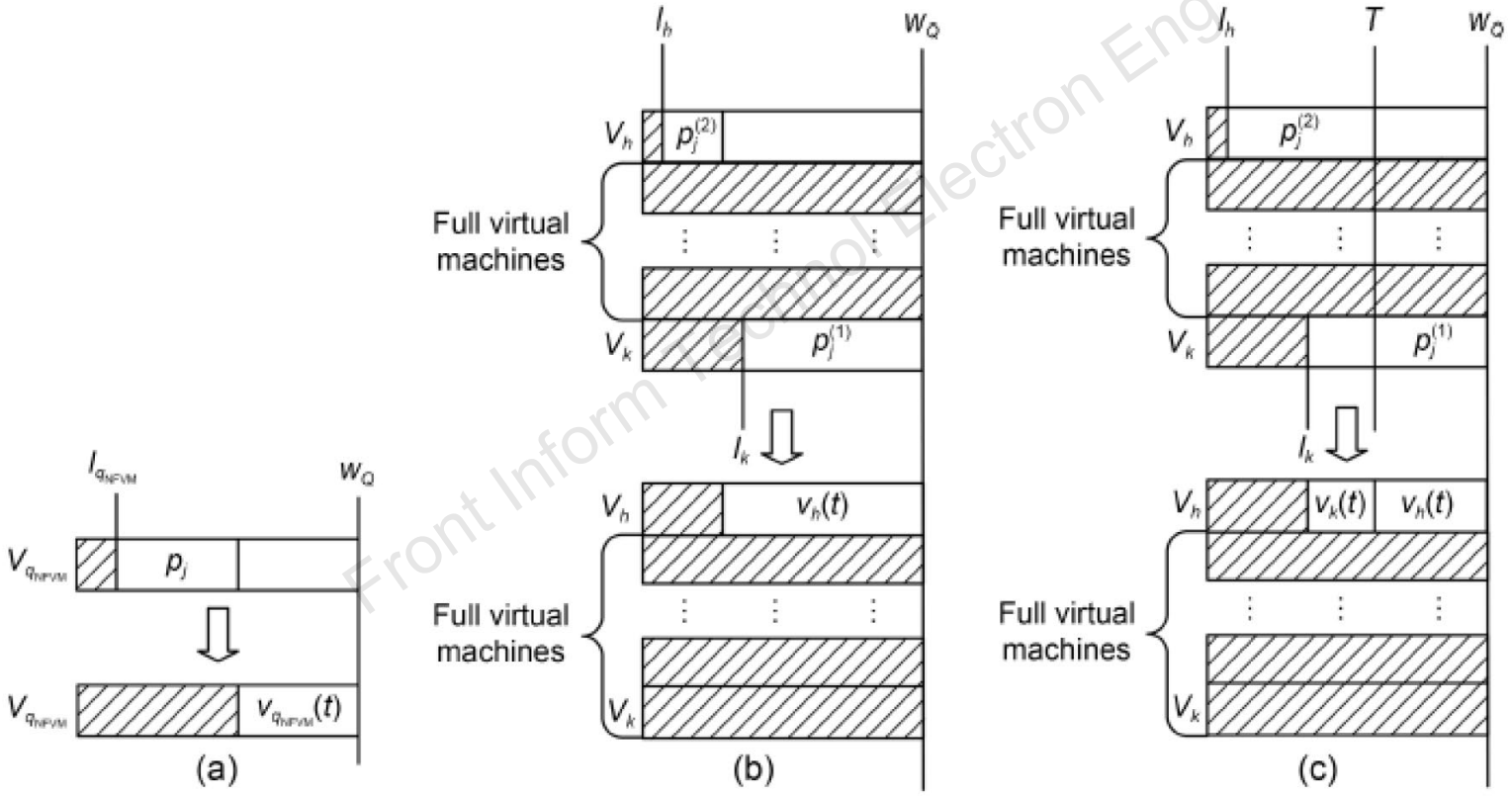


Fig. 2 Jobs scheduled in Step 3 (a), Step 4 (b), and Step 5 (c)

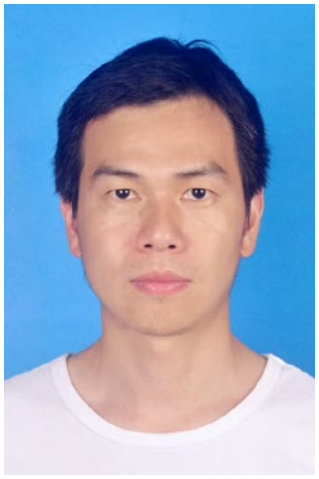
Major results

Theorem Algorithm 1 generates an optimal schedule for problem $Qm, r_i|\text{pmtn}|C_{\max}$.

- The time complexity of our algorithm is $O\left(nm+m^2\right)$.
- The number of preemptions is at most $\frac{1}{2}(m^2+3\text{m})-2$.

Conclusions

1. We provide a lower bound on the optimal makespan by transforming the *m* uniform machines into *m* virtual machines.
2. We provide a solution algorithm for the preemptive scheduling on parallel uniform machines with non-simultaneous available time, i.e., $Qm, r_i|\text{pmtn}|C_{\max}$.



Hao Zhou is currently a full Professor at Zhejiang Shuren University. His major research interests are in mathematical models, scheduling theory and approximation algorithms. He has published 20 papers in journals such as *Theoretical Computer Science*, *Asia-pacific Journal of Operational Research*, and *Acta Mathematica Sinica-English Series*.

Yiwei JIANG is currently a full Professor at Zhejiang Gongshang University. His major research interests are in scheduling theory, supply chain management, approximation algorithms, and online algorithms for the combinatorial optimization problem. He has published over 80 papers in journals such as *Naval Research Logistics*, *European Journal of Operational Research*, *Information Sciences*, and *Computers & Industrial Engineering*.