Optimal Online Load Maximization with Commitment
Samin Jamalabadi, Uwe Schwiegelshohn
TU Dortmund University
Chris Schwiegelshohn
Sapienza University of Rome
Pm|online, ε, commit|\( \sum p_j \cdot (1-U_j) \)

- **Pm**: \( m \) parallel identical machines.
- **online**: jobs are submitted over time.
  - We do not know the existence nor any properties of future jobs.
- **ε**: a job \( J_j \) has deadline \( d_j \geq r_j + \varepsilon \cdot p_j \) with constant slack parameter \( \varepsilon \).
  - \( r_j \): submission time of job \( J_j \)
  - \( p_j \): processing time of job \( J_j \)
- **commit**: we must decide immediately after submission whether to reject a new job \( J_j \) \((U_j = 1)\) or to accept it \((U_j = 0)\).
  - For \( U_j = 0 \), we must also immediately fix the start time of the job.
  - We must complete every accepted job on time.
- \( \sum p_j \cdot (1-U_j) \): we want to maximize the total processing time of all accepted jobs.
Algorithm Choices

- Acceptance
- Allocation
- Timing
Acceptance Algorithms for $P2$

Threshold acceptance

Threshold

Greedy acceptance
Algorithm Choices

- Acceptance
  - Greedy: the resulting schedule completes all accepted jobs on time.
  - Threshold: the deadline of an accepted job is at least as large as a deadline threshold.

- Allocation

- Timing
Allocation for $P2$ with Greedy Acceptance

Skewed allocation

Balanced allocation
Algorithm Choices

- Acceptance
  - Greedy: the resulting schedule completes all accepted jobs on time.
  - Threshold: the deadline of an accepted job is at least as large as a deadline threshold.

- Allocation
  - Skewed: the candidate machine with the highest load
  - Balanced: the candidate machine with the minimal load (for greedy) or the candidate machine that increases the threshold by the smallest amount (for threshold).

- Timing
Job Starting for $P2$ with Greedy Acceptance

Delay

Semi-active
Algorithm Choices

- Acceptance
  - Greedy: the resulting schedule completes all accepted jobs on time.
  - Threshold: the deadline of an accepted job is at least as large as a deadline threshold.

- Allocation
  - Skewed: the candidate machine with the highest load
  - Balanced: the candidate machine with the minimal load (for greedy) or the candidate machine that increases the threshold by the smallest amount (for threshold).

- Timing
  - Semi-active: as early as possible on the allocated machine.
  - Delay: possible intermediate idle time on the allocated machine.
Threshold Calculation

- The machines are indexed in decreasing order of their outstanding loads.
- For machine $m_i$, we calculate a machine specific threshold using a function $f_i(\varepsilon)$ and the outstanding load of the machine at time $t$. 
  \[ d_{lim, i} \bigg|_t = load(m_i) \bigg|_t \cdot f_i(\varepsilon) + t \]
- The threshold is the maximum of the machine specific thresholds. 
  \[ d_{lim} \bigg|_t = \max_{1 \leq i \leq m} \left. d_{lim, i} \right|_t \]
- We set $f_m(\varepsilon) = \frac{1+\varepsilon}{\varepsilon}$ and determine the remaining $f_i(\varepsilon)$ recursively. 
  \[ m \cdot f_i(\varepsilon) + 1 = \frac{\sum_{h=1}^{i-1} f_i(\varepsilon) + 1}{\sum_{h=1}^{i-1} f_i(\varepsilon) - (i - 1) + 1} = \text{const} \quad \text{for} \quad 1 \leq i \leq m \]
Competitive Ratio

- Threshold allocation with a skewed and semi-active schedule has the competitive ratio
  \[ m \cdot f_1(\varepsilon) + 1 \geq 2m + 1 \text{ for } f_1(\varepsilon) \geq 2. \]
- \( \varepsilon_T = \arg \max \{ f_1(\varepsilon) = 2 \} \) decreases with increasing \( m \).

<table>
<thead>
<tr>
<th>( m )</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \varepsilon_T )</td>
<td>0.2857</td>
<td>0.0900</td>
<td>0.0291</td>
<td>0.0098</td>
</tr>
</tbody>
</table>

- Greedy allocation with a skewed and semi-active schedule has the competitive ratio
  \[ \frac{1}{m} + \frac{1 + \varepsilon}{\varepsilon} \text{ for } 0 < \varepsilon \leq 1. \]
Greedy Acceptance

- Intuitively, greedy acceptance is the simplest approach.

- The competitive ratio of greedy acceptance is identical to the competitive ratio of the following min-threshold approach for $0 < \varepsilon \leq 1$:

$$d_{lim,i}\big|_t = \left.\text{load}(m_i)\right|_t \cdot \frac{1 + \varepsilon}{\varepsilon} + t$$

$$d_{lim}\big|_t = \min_{1 \leq i \leq m} d_{lim,i}\big|_t$$
Proof Concepts

- Key lemma for the (max-)threshold approach:
  - Allocation of a new job to a machine without the maximum outstanding load will turn this machine into the machine with the maximum outstanding load if $f_1(\epsilon) \geq 2$ holds.

- Partitioning of the resulting schedule into several intervals
  - We determine how much load of every interval cannot be executed outside of this interval in any optimal schedule that has accepted the corresponding jobs.
Previous Results

- For $m = 1$, greedy acceptance (Goldwasser 1999, 2003) with a semi-active schedule has the tight competitive ratio
  \[ 1 + \frac{1 + \varepsilon}{\varepsilon} \text{ for } 0 < \varepsilon. \]

- For $m > 1$, greedy allocation with a balanced and semi-active schedule has the competitive ratio (Kim, Chwa 2001)
  \[ 1 + \frac{1 + \varepsilon}{\varepsilon} \text{ for } 0 < \varepsilon. \]
Results for $P2$
Competitive Ratio

- Gap between greedy and (max-)threshold allocation at $\varepsilon_T$

<table>
<thead>
<tr>
<th>$m$</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gap</td>
<td>0</td>
<td>5.44</td>
<td>26.61</td>
<td>92.24</td>
</tr>
</tbody>
</table>

- For $m > 2$, we need $m$ algorithms covering different intervals within $(0,1]$.  

- The algorithms are a combination of min-thresholds and max-thresholds.
Lower Bound for $P2$ and Greedy Allocation

- Base job $p_i = 1$, $d_i$ large
- Up to four jobs $p_j = 1 - \delta$, $d_j = 2 \cdot (1 - \delta)$
- Optimal schedule $c = 5$
Lower Bound for $P2$ and Greedy Allocation

two jobs $p_j = 1 - 2\delta$

two jobs $p_k = p_j \cdot \frac{1}{\varepsilon}$

optimal schedule $c = (1 + 2\cdot \frac{1+\varepsilon}{\varepsilon})/2$
Lower Bound for $P2$ and (Max-)Threshold Allocation

- Base job $p_i = 1$, $d_i$ large
- Two jobs $p_j = 1 - \delta$, $d_j = p_j \cdot (1 + \varepsilon)$

Optimal schedule $c = (1 + 2 \cdot \frac{1+\varepsilon}{\varepsilon})/2$
Lower Bound for $P2$ and (Max-)Threshold Allocation

$\text{optimal schedule } c = 1 + 2f_1(\varepsilon)$

up to two jobs $p_j = (1 - \delta) \cdot (f_1(\varepsilon) - 1),
\quad d_j = (1 - \delta) \cdot f_1(\varepsilon)$
Lower Bound for \( P2 \) and (Max-)Threshold Allocation

\[
two 
\text{jobs } p_k = (1 - \delta) \cdot \frac{1}{\varepsilon}
\]

\[
1 - \delta 
(1 - \delta) \cdot \frac{1 + \varepsilon}{\varepsilon}
\]

optimal schedule \( c = \frac{(1 + 2 \cdot \frac{1 + \varepsilon}{\varepsilon})}{f_1(\varepsilon)} = 1 + 2f_1(\varepsilon)
\)
Interval \((1, \infty)\) for \(\varepsilon\)

- The optimal competitive ratio is larger for any \(\varepsilon \in (0,1]\) than the optimal competitive ratio for any \(\varepsilon \in (1, \infty)\).
  - The problem becomes easier for \(\varepsilon > 1\)?
  - Previous results seem to support this claim.
- Observation: The presented optimal online algorithms for \(\varepsilon \in (0,1]\) only use semi-active schedules avoiding any start-time problem.
- It is not possible to obtain the competitive ratio \(\frac{1}{m} + \frac{1+\varepsilon}{\varepsilon}\) for all \(\varepsilon \in (1, \infty)\) when using only semi-active schedules.
  - We consider an example with \(\varepsilon = 2\) and the \(P2\) environment.
- Progression of time limits the competitive ratio for large \(\varepsilon\) and \(m \geq 3\).
Lower Bound for $P2$, $\varepsilon = 2$, and Semi-active Schedules

base job $p_i = 1$, $d_i$ large

up to six jobs $p_j = \frac{1}{3}$, $d_j = \frac{4}{3} - 3\delta$

optimal schedule $c = \frac{9}{4}$
Lower Bound for $P2$, $\varepsilon = 2$, and Semi-active Schedules

six jobs $p_j = \frac{1}{3} - \delta$

$1 - 3\delta$

optimal schedule $c = \frac{11}{5}$
Lower Bound for $Pm$ and Large $\varepsilon$

- Base job $p_i = 1$, $d_i$ large
- Many jobs small $p_j$, $d_j = 1$
- Optimal schedule $c = 2$
Lower Bound for $Pm$ and Large $\varepsilon$

many jobs
small $p_j$
$d_j = 1/2$

optimal schedule $c = \frac{4m+2}{3m+1}$
Conclusion

- For the problem $Pm|\text{online, } \varepsilon, \text{commit}|\sum p_j : (1-U_j)$, we presented online algorithms with an optimal competitive ratio for $0 < \varepsilon \leq 1$.

- The optimal algorithm consists of $m$ different algorithms that are each valid for a subinterval of $(0,1]$ of $\varepsilon$.

- The algorithms use thresholds, skewed allocation and semi-active schedules.

- For $\varepsilon > 1$, the optimal competitive ratio is smaller but the algorithms are more complicated.