New Heuristics and Integer Programming Formulations for Scheduling Divisible Load Tasks

Elbio R. Abib (Microsoft, USA)
Celso C. Ribeiro (UFF, Brazil)

IEEE CISched 2009, Nashville
Agenda

- Divisible load model
  - System model and problem formulation
  - Single- and multi-installment scheduling
- Single-installments: mixed integer programming
- Linear-time algorithm for a given activation order
- Fast constructive heuristic with feedback
- Computational experiments
- Multiple-installments: mixed integer programming
Divisible load model
Divisible load model

- Load may be split continuously into arbitrarily many small chunks
- No precedence constraints
System model and problem formulation

- Interconnection topology: star network
  - Dedicated grid
- Model: one master - n workers
  - Master owns the total load W
- No communication/computation overlap in any processor
- No communication overlap through the master

![Diagram showing a star network with P0 at the center and P1, P2, P3, Pn connected to P0]
System model and problem formulation

- Single-installment scheduling
  - Each processor receives portion $\alpha_i$ of total load
  - Master takes $g_i + G_i \alpha_i$ time units to send the data to processor $P_i$
  - Processor $P_i$ takes $w_i \alpha_i$ time units to process data

Variable communication time $G_i \alpha_i$

Fixed latency $g_i$

Variable computation time $w_i \alpha_i$
Single-installment scheduling

**Optimal scheduling**

**Non-optimal scheduling**
Multi-installment scheduling

Communication/computation concurrency

New period

New period
Related work

- Divisible load model introduced by Cheng and Robertazzi (1988)
- Effect of latency in communication studied by Blazewicz and Drozdowski (1997)
- Beaumont et al. (2005): non-linear integer programming formulation for single-installment systems with latencies
- Linear integer programming formulations for single- and multi-installment systems with latencies not available
Single-installment mixed integer programming formulation
Single-installment scheduling

- Problem consists of determining
  - the processors to be used (and their number),
  - their activation order,
  - and their loads,
- ... so as to minimize the makespan.
Formulation

\( T^* = \text{minimum } T \) subject to:

\[
\sum_{i=1}^{n} x_{ij} \leq 1 \quad j = 1, \ldots, n
\]

\[
\sum_{j=1}^{n} x_{ij} \leq 1 \quad i = 1, \ldots, n
\]

\[
\sum_{i=1}^{n} x_{ij} \geq \sum_{i=1}^{n} x_{i,j+1} \quad j = 1, \ldots, n - 1
\]

\[
\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ij} = W
\]

\[
\alpha_{ij} \leq W x_{ij} \quad i, j = 1, \ldots, n
\]

\[
t_1 = 0
\]

\[
t_j \geq t_{j-1} + \sum_{i=1}^{n} (g_i x_{i,j-1} + G_i \alpha_{i,j-1}) \quad j = 2, \ldots, n
\]

\[
t_j + \sum_{i=1}^{n} (g_i x_{ij} + (G_i + w_i) \alpha_{ij}) = T \quad j = 1, \ldots, n
\]

\[
x_{ij} \in \{0, 1\}
\]

\[
\alpha_{ij} \geq 0
\]
Formulation

\[ x_{ij} = \begin{cases} 1, & \text{if processor } P_i \text{ is the } j\text{-th to be activated and to receive data} \\ 0, & \text{otherwise} \end{cases} \]

\[ \alpha_{ij} = \begin{cases} >0, & \text{is the amount of data sent to } P_i \text{ if it is the } j\text{-th to be activated} \\ 0, & \text{otherwise} \end{cases} \]

\[ t_j \text{ is the time in which the } j\text{-th processor to be activated starts receiving its data} \]

\[ T^* = \min T \]
subject to:
\[ \sum_{i=1}^{n} x_{ij} \leq 1 \quad \text{for } j = 1, \ldots, n \]
\[ \sum_{j=1}^{n} x_{ij} \leq 1 \quad \text{for } i = 1, \ldots, n \]
\[ \sum_{i=1}^{n} x_{i,j} \geq \sum_{i=1}^{n} x_{i,j+1} \quad \text{for } j = 1, \ldots, n-1 \]
\[ \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ij} = W \]
\[ \alpha_{ij} \leq Wx_{ij} \quad \text{for } i, j = 1, \ldots, n \]
\[ t_1 = 0 \]
\[ t_j \geq t_{j-1} + \sum_{i=1}^{n} (g_i x_{i,j-1} + G_i \alpha_{i,j-1}) \quad \text{for } j = 2, \ldots, n \]
\[ t_j + \sum_{i=1}^{n} (g_i x_{ij} + (G_i + w_i) \alpha_{ij}) = T \quad \text{for } j = 1, \ldots, n \]
\[ x_{ij} \in \{0, 1\} \quad \text{for } i, j = 1, \ldots, n \]
\[ \alpha_{ij} \geq 0 \quad \text{for } i, j = 1, \ldots, n. \]
Formulation

\[ T^* = \min \text{subject to:} \]
\[ \sum_{i=1}^{n} x_{ij} \leq 1 \]
\[ \sum_{j=1}^{n} x_{ij} \leq 1 \quad i = 1, \ldots, n \]
\[ \sum_{i=1}^{n} x_{ij} \geq \sum_{i=1}^{n} x_{i,j+1} \quad j = 1, \ldots, n-1 \]
\[ \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ij} = W \]
\[ \alpha_{ij} \leq W x_{ij} \quad i, j = 1, \ldots, n \]
\[ t_1 = 0 \]
\[ t_j \geq t_{j-1} + \sum_{i=1}^{n} \left( g_i x_{i,j-1} + G_i \alpha_{i,j-1} \right) \quad j = 2, \ldots, n \]
\[ t_j + \sum_{i=1}^{n} \left( g_i x_{ij} + (G_i + w_i) \alpha_{ij} \right) = T \quad j = 1, \ldots, n \]
\[ x_{ij} \in \{0, 1\} \quad i, j = 1, \ldots, n \]
\[ \alpha_{ij} \geq 0 \quad i, j = 1, \ldots, n \]
Formulation

\[ x_{ij} = 1, \text{ if processor } P_i \text{ is the } j\text{-th to be activated and to receive data} \]
\[ x_{ij} = 0, \text{ otherwise} \]

\[ \alpha_{ij} > 0, \text{ is the amount of data sent to } P_i \text{ if it is the } j\text{-th to be activated} \]
\[ \alpha_{ij} = 0, \text{ otherwise} \]

\[ t_j \text{ is the time in which the } j\text{-th processor to be activated starts receiving its data} \]

\[ T^* = \min T \]
subject to:
\[ \sum_{i=1}^{n} x_{ij} \leq 1 \]
\[ \sum_{j=1}^{n} x_{ij} \leq 1 \]
\[ \sum_{i=1}^{n} x_{ij} \geq \sum_{i=1}^{n} x_{i,j+1} \quad j = 1, \ldots, n - 1 \]
\[ \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ij} = W \quad \text{(5)} \]
\[ \alpha_{ij} \leq W x_{ij} \quad i, j = 1, \ldots, n \]
\[ t_1 = 0 \quad \text{(6)} \]
\[ t_j \geq t_{j-1} + \sum_{i=1}^{n} (g_i x_{i,j-1} + G_i \alpha_{i,j-1}) \quad j = 2, \ldots, n \]
\[ t_j + \sum_{i=1}^{n} (g_i x_{ij} + (G_i + w_i) \alpha_{ij}) = T \quad j = 1, \ldots, n \]
\[ x_{ij} \in \{0, 1\} \quad i, j = 1, \ldots, n \]
\[ \alpha_{ij} \geq 0 \quad i, j = 1, \ldots, n. \]

A processor may be the \((j+1)-th\) to be activated only if there are other \(j\) processors already activated.
Formulation

$x_{ij} = 1$, if processor $P_i$ is the $j$-th to be activated and to receive data
$x_{ij} = 0$, otherwise

$\alpha_{ij} > 0$, is the amount of data sent to $P_i$ if it is the $j$-th to be activated
$\alpha_{ij} = 0$, otherwise

$t_j$ is the time in which the $j$-th processor to be activated starts receiving its data

\begin{equation}
T^* = \text{minimum } T
\end{equation}

subject to:
\begin{align}
\sum_{i=1}^{n} x_{ij} &\leq 1 & j &= 1, \ldots, n \\
\sum_{j=1}^{n} x_{ij} &\leq 1 & i &= 1, \ldots, n \\
\sum_{i=1}^{n} x_{ij} &\geq \sum_{i=1}^{n} x_{i,j+1} & j &= 1, \ldots, n-1 \\
\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ij} &= W \\
\alpha_{ij} &\leq W x_{ij} \\
t_1 &= 0 \\
t_j &\geq t_{j-1} + \sum_{i=1}^{n} (g_i x_{i,j-1} + G_i \alpha_{i,j-1}) & j &= 2, \ldots, n \\
t_j + \sum_{i=1}^{n} (g_i x_{ij} + (G_i + w_i) \alpha_{ij}) &= T & j &= 1, \ldots, n \\
x_{ij} &\in \{0, 1\} & i, j &= 1, \ldots, n \\
\alpha_{ij} &\geq 0 & i, j &= 1, \ldots, n.
\end{align}
Formulation

$x_{ij} = 1$, if processor $P_i$ is the $j$-th to be activated and to receive data
$x_{ij} = 0$, otherwise

$\alpha_{ij} > 0$, is the amount of data sent to $P_i$ if it is the $j$-th to be activated
$\alpha_{ij} = 0$, otherwise

$t_j$ is the time in which the $j$-th processor to be activated starts receiving its data

\[
T^* = \min T
\]
subject to:
\[
\sum_{i=1}^{n} x_{ij} \leq 1 \quad j = 1, \ldots, n
\]
\[
\sum_{j=1}^{n} x_{ij} \leq 1 \quad i = 1, \ldots, n
\]
\[
\sum_{i=1}^{n} x_{ij} \geq \sum_{i=1}^{n} x_{i,j+1} \quad j = 1, \ldots, n - 1
\]
\[
\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ij} = W
\]
\[
\alpha_{ij} \leq W x_{ij}
\]
\[
t_1 = 0
\]
\[
t_j \geq t_{j-1} + \sum_{i=1}^{n} (g_i x_{i,j-1} + G_i \alpha_{i,j-1})
\]
\[
t_j + \sum_{i=1}^{n} (g_i x_{ij} + (G_i + w_i) \alpha_{ij}) = T
\]
\[
x_{ij} \in \{0, 1\}
\]
\[
\alpha_{ij} \geq 0
\]

Processor $i$ can only receive load as the $j$-th if it is the $j$-th to be activated
Formulation

$x_{ij} = 1$, if processor $P_i$ is the $j$-th to be activated and to receive data

$x_{ij} = 0$, otherwise

$\alpha_{ij} > 0$, is the amount of data sent to $P_i$ if it is the $j$-th to be activated

$\alpha_{ij} = 0$, otherwise

$t_j$ is the time in which the $j$-th processor to be activated starts receiving its data

\[
T^* = \min \ T
\]
subject to:
\[
\sum_{i=1}^{n} x_{ij} \leq 1 \quad j = 1, \ldots, n
\]
\[
\sum_{j=1}^{n} x_{ij} \leq 1 \quad i = 1, \ldots, n
\]
\[
\sum_{i=1}^{n} x_{ij} \geq \sum_{i=1}^{n} x_{i,j+1} \quad j = 1, \ldots, n - 1
\]
\[
\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ij} = W
\]
\[
\alpha_{ij} \leq W x_{ij}
\]
\[
t_1 = 0
\]
\[
t_j \geq t_{j-1} + \sum_{i=1}^{n} (g_i x_{ij} + G_i \alpha_{i,j-1}) \quad j = 2, \ldots, n
\]
\[
t_j + \sum_{i=1}^{n} (g_i x_{ij} + (G_i + w_i) \alpha_{ij}) = T \quad j = 1, \ldots, n
\]
\[
x_{ij} \in \{0, 1\}
\]
\[
\alpha_{ij} \geq 0
\]
Fromulation

\[ x_{ij} = \begin{cases} 1, & \text{if processor } P_i \text{ is the } j\text{-th to be activated and to receive data} \\ 0, & \text{otherwise} \end{cases} \]

\[ \alpha_{ij} > 0, \text{ is the amount of data sent to } P_i \text{ if it is the } j\text{-th to be activated} \]

\[ \alpha_{ij} = 0, \text{ otherwise} \]

\[ t_j \text{ is the time in which the } j\text{-th processor to be activated starts receiving its data} \]

\[ T^* = \text{minimum } T \]

subject to:

\[ \sum_{i=1}^{n} x_{ij} \leq 1 \quad j = 1, \ldots, n \quad (2) \]

\[ \sum_{j=1}^{n} x_{ij} \leq 1 \quad i = 1, \ldots, n \quad (3) \]

\[ \sum_{i=1}^{n} x_{ij} \geq \sum_{i=1}^{n} x_{i,j+1} \quad (4) \]

\[ \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ij} = W \quad \alpha_{ij} \leq W x_{ij} \quad (5) \]

\[ t_1 = 0 \]

\[ t_j \geq t_{j-1} + \sum_{i=1}^{n} (g_i x_{i,j-1} + G_i \alpha_{i,j-1}) \quad j = 2, \ldots, n \quad (8) \]

\[ t_j + \sum_{i=1}^{n} (g_i x_{ij} + (G_i + w_i) \alpha_{ij}) = T \quad j = 1, \ldots, n \quad (9) \]

\[ x_{ij} \in \{0, 1\} \quad i, j = 1, \ldots, n \quad (10) \]

\[ \alpha_{ij} \geq 0 \quad i, j = 1, \ldots, n. \quad (11) \]
Formulation

\[ T^* = \text{minimum } T \]

subject to:

\[ \sum_{i=1}^{n} x_{ij} \leq 1 \quad j = 1, \ldots, n \]  
(1)

\[ \sum_{j=1}^{n} x_{ij} \leq 1 \quad i = 1, \ldots, n \]  
(2)

\[ \sum_{i=1}^{n} x_{ij} \geq \sum_{i=1}^{n} x_{i,j+1} \quad j = 1, \ldots, n - 1 \]  
(3)

\[ \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ij} = W \]  
(4)

\[ \alpha_{ij} \leq W x_{ij} \quad i, j = 1, \ldots, n \]  
(5)

\[ t_1 = 0 \]  
(6)

\[ t_j \geq t_{j-1} + \sum_{i=1}^{n} (g_i x_{i,j-1} + G_i \alpha_{i,j-1}) \quad j = 2, \ldots, n \]  
(7)

\[ t_j + \sum_{i=1}^{n} (g_i x_{ij} + (G_i + w_i) \alpha_{ij}) = T \quad j = 1, \ldots, n \]  
(8)

\[ x_{ij} \in \{0, 1\} \]  

\[ \alpha_{ij} \geq 0 \]  

\[ x_{ij}=1, \text{ if processor } P_i \text{ is the } j-\text{th to be activated and to receive data} \]

\[ x_{ij}=0, \text{ otherwise} \]

\[ \alpha_{ij}>0, \text{ is the amount of data sent to } P_i \text{ if it is the } j-\text{th to be activated} \]

\[ \alpha_{ij}=0, \text{ otherwise} \]

\[ t_j \text{ is the time in which the } j-\text{th processor to be activated starts receiving its data} \]

All processors finish at the same time T
Linear-time algorithm for a given activation order
Linear-time algorithm for a given activation order

- Blazewicz and Drozdowski (1997): if the activation order and the number of processors are known, the optimal loads are:

\[
\alpha_k = \alpha_{\ell} \prod_{j=k+1}^{\ell} f_j + \sum_{j=k+1}^{\ell} \left( \frac{g_j}{w_j^{-1}} \prod_{i=k+1}^{j-1} f_i \right), \quad k = 1, \ldots, \ell-1
\]  

(14)

\[
\alpha_{\ell} = \frac{W - \sum_{k=1}^{\ell-1} \sum_{j=k+1}^{\ell} \left( \frac{g_j}{w_j^{-1}} \prod_{i=k+1}^{j-1} f_i \right)}{1 + \sum_{k=1}^{\ell-1} \prod_{j=k+1}^{\ell} f_j}
\]  

(15)

\[
V(\ell) = \frac{g_{\ell}}{w_{\ell-1}} \sum_{k=1}^{\ell-1} \prod_{i=k+1}^{\ell-1} f_i + V(\ell-1)
\]
Linear-time algorithm for a given activation order

- $F(\ell) = \sum_{k=1}^{\ell-1} \prod_{i=k+1}^{\ell-1} f_i$ may be recursively defined as
  
  $F(1) = 0, F(2) = 1, \text{ and } F(\ell) = 1 + F(\ell - 1) f_{\ell - 1}$

- $V(\ell) = \frac{g_\ell}{w_{\ell - 1}} F(\ell) + V(\ell - 1)$ may be computed in time $O(1)$

- Optimal solution has the maximum number $\ell^*$ of processors such that
  
  $V(\ell^*) \leq W$
Linear-time algorithm for a given activation order

Algorithm:

- Compute $F(k)$ for $k=1,\ldots, n$ in time $O(n)$
- Compute $V(k)$ for $k=1,\ldots, n$ in time $O(n)$
- Optimal number of processors is the largest number of processors $k$ such that $V(k) \leq W$
- Load assigned to each processor can be computed in time $O(n)$ as described by Blazewicz and Drozdowski (1997)
Fast constructive heuristic with feedback
Constructive feedback heuristic

- Heuristic for scheduling divisible loads may be seen as any algorithm that generates a “good” activation order and computes the associated optimal loads.
- Constructive feedback heuristic makes use of the idea of **equivalent processors**
- Each solution is uniquely associated with:
  - activation order given by a vector $\pi$
  - makespan $T$
Constructive feedback heuristic

- Equivalent processor:
  - Given a time period $T$, if a load $\alpha_i = (T-g_i) / (w_i+G_i)$ is sent to $P_i$ then it remains busy with communication and processing for this full time period.
  - Equivalent to a processor $P_i^{eq}$ with the same processing power, no communication latency, and throughput $1/G_i^{eq} = 1/[G_i + (g_i / \alpha_i)]$

- Optimal activation order for a system with no latencies: processors with higher communication throughput receive data first.
Constructive feedback heuristic

Create activation order \( \pi \) with higher throughput processors first

UB = optimal makespan for activation order \( \pi \)

Repeat

   BestOrder = \( \pi \)
   \( T^* = UB \)

   Compute new order \( \pi \)

   UB = optimum makespan for new activation order \( \pi \)

Until UB \( \geq T^* \)
Constructive feedback heuristic

Create activation order $\pi$ with higher throughput processors first

$UB = \text{optimal makespan for activation order } \pi$

Repeat

BestOrder = $\pi$

$T^* = UB$

For $j = 1, ..., n$ do

Compute equivalent processor $P_{i_{eq}}$ for each $P_i$ not in $\pi[1], ..., \pi[j-1]$

$\pi[j] = \text{processor whose equivalent has the highest throughput}$

Update remaining time $UB$ by subtracting the time taken by that processor

$UB = \text{optimum makespan for activation order } \pi$

Until $UB \geq T^*$
Computational experiments
Computational experiments

- 120 grid configurations
  - Number of processors: 10, 20, 40, 80, and 160
  - 24 configurations of $w_i$, $G_i$, and $g_i$
- Load $W$: 100, 200, 400, 800, 1600, and 3200
- CPLEX time limit 3600 seconds
Computational experiments

**CPLEX solved 490 out of 720 test instances**

<table>
<thead>
<tr>
<th>$w_i$</th>
<th>$g_i$</th>
<th>$G_i$</th>
<th>$n = 10$</th>
<th>$n = 20$</th>
<th>$n = 40$</th>
<th>$n = 80$</th>
<th>$n = 160$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>opt.</td>
<td>opt.</td>
<td>opt.</td>
<td>opt.</td>
<td>opt.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>time</td>
<td>time</td>
<td>time</td>
<td>time</td>
<td>time</td>
</tr>
<tr>
<td>low</td>
<td>low</td>
<td>low</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.25</td>
<td>0.81</td>
<td>43.81</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>low</td>
<td>low</td>
<td>high</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.05</td>
<td>0.08</td>
<td>0.36</td>
<td>1.37</td>
<td>4.94</td>
</tr>
<tr>
<td>low</td>
<td>high</td>
<td>low</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.08</td>
<td>0.20</td>
<td>0.46</td>
<td>2.78</td>
<td>9.27</td>
</tr>
<tr>
<td>low</td>
<td>high</td>
<td>high</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.14</td>
<td>0.25</td>
<td>0.50</td>
<td>1.20</td>
<td>3.57</td>
</tr>
<tr>
<td>high</td>
<td>low</td>
<td>low</td>
<td>18</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>14.64</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>high</td>
<td>low</td>
<td>high</td>
<td>18</td>
<td>18</td>
<td>16</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.06</td>
<td>9.37</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>high</td>
<td>high</td>
<td>low</td>
<td>18</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2.85</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>high</td>
<td>high</td>
<td>high</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.13</td>
<td>0.68</td>
<td>52.52</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>
Computational experiments

Feedback heuristic found optimal solutions for 398 out of the 490 instances for which CPLEX found the optimum

<table>
<thead>
<tr>
<th>$w_i$</th>
<th>$g_i$</th>
<th>$G_i$</th>
<th>10</th>
<th>20</th>
<th>40</th>
<th>80</th>
<th>160</th>
</tr>
</thead>
<tbody>
<tr>
<td>low</td>
<td>low</td>
<td>low</td>
<td>11</td>
<td>10</td>
<td>4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>low</td>
<td>low</td>
<td>high</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>17</td>
</tr>
<tr>
<td>low</td>
<td>high</td>
<td>low</td>
<td>13</td>
<td>17</td>
<td>17</td>
<td>11</td>
<td>13</td>
</tr>
<tr>
<td>low</td>
<td>high</td>
<td>high</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>14</td>
</tr>
<tr>
<td>high</td>
<td>low</td>
<td>low</td>
<td>18</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>high</td>
<td>low</td>
<td>high</td>
<td>18</td>
<td>18</td>
<td>16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>high</td>
<td>high</td>
<td>low</td>
<td>16</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>high</td>
<td>high</td>
<td>high</td>
<td>14</td>
<td>14</td>
<td>5</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>
Computational experiments

Average deviation from optimal value smaller than 0.5% for most of the remaining 92 instances solved to optimality by CPLEX

Heuristic ran for 3ms on average and never for more than 32ms

Fig. 5. Percent deviation of non-optimal makespans
Multiple-installment mixed integer programming formulation
Formulation

Model 2 Multiple-installment mixed integer program

\[ T^* = \text{minimum } T \]  
\text{subject to:} 
\[ \sum_{i=1}^{n} x_{kij} \leq 1 \]  
\[ \sum_{j=1}^{n} x_{kij} \leq 1 \]  
\[ \sum_{i=1}^{n} x_{kij} \geq \sum_{i=1}^{n} x_{ki,j+1} \]  
\[ \sum_{k=1}^{p} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{kij} = W \]  
\[ \alpha_{kij} \leq W x_{kij} \]  
\[ t_{11} = 0 \]  
\[ t_{kj} \geq t_{kj-1} + \sum_{i=1}^{n} (g_i x_{kji,j-1} + G_i \alpha_{kij,j-1}) \]  
\[ j = 1, \ldots, n, \quad k = 1, \ldots, p \]  
\[ t_{k,j} + \sum_{i=1}^{n} (g_i x_{kij} + G_i \alpha_{kij}) \geq t_{k-1,j} + \sum_{i=1}^{n} (g_i x_{k-1,ij} + (G_i + w_i) \alpha_{k-1,ij}) \]  
\[ j = 1, \ldots, n, \quad k = 1, \ldots, p \]  
\[ t_{k1} \geq t_{k-1,n} + \sum_{i=1}^{n} (g_i x_{k-1,1} + G_i \alpha_{k-1,1}) \]  
\[ k = 2, \ldots, p \]  
\[ x_{kij} \in \{0, 1\} \]  
\[ \alpha_{kij} \geq 0 \]  

In this case, model also determines the optimal number of installments.
Preliminary results show significant improvements in the makespans are possible, with respect to those obtained by single- and multi-round heuristics.
Concluding remarks

- New mixed integer programming formulations for single- and multi-round schedulings.
- Linear-time algorithm for the special case in which the processor activation order is known.
- Fast and effective greedy-with-feedback heuristic.
- Randomized multistart version of feedback heuristic with local search.
- Extension to multi-round schedulings.