Tardiness Models

Contents

1. Moor’s algorithm which gives an optimal schedule with the minimum number of tardy jobs 1 $\| \sum U_j$
2. An algorithm which gives an optimal schedule with the minimum total tardiness 1 $\| \sum T_j$

Literature:
- Scheduling, Theory, Algorithms, and Systems, Michael Pinedo, Prentice Hall, 1995, Chapters 3.3 and 3.4
- or new: Second Addition, 2002, Chapter 3.

Optimal schedule has this form $j_1, \ldots, j_k$ meet their due dates $d_j$, do not meet their due dates $d_j$, EDD rule

Moor’s algorithm for 1 $\| \sum U_j$

Optimal schedule has this form $j_{\bar{d}}$, $j_{\bar{d}}$

Step 1
$J = \emptyset$
$J^d = \emptyset$
$J^C = \{1, \ldots, n\}$

Step 2
Let $j^*$ be such that $d_j = \min_{j \in J^C} d_j$
Add $j^*$ to $J$
Delete $j^*$ from $J^C$

Step 3
If $d_{j^*} \leq \sum_{j \in J} p_j$
else let $k^*$ be such that $p_{k^*} = \max_{j \in J^C} p_j$
Add $k^*$ to $J^d$

Step 4
If $J^F = \emptyset$ STOP
else go to Step 2.

Example


t_1 = 7 < d_1 = 9
j^* = 1
$J = \{1\}$, $J^d = \emptyset$, $J^C = \{2, 3, 4, 5\}$, $t_1 = 7 < 9 = d_1$

$t_2 = 15 < 17 = d_2$

$t_3 = 19 > 18 = d_3$

$k^* = 2$

$t_4 = 17 < 19 = d_4$

$t_5 = 23 > 21 = d_5$

The Total Tardiness 1 $\| \sum T_j$ is NP hard

Lemma. If $p_j < p_k$ and $d_j < d_k$ then there exists an optimal sequence in which job $j$ is scheduled before job $k$.

Lemma. There exists an integer $\delta$ $0 \leq \delta \leq n - k$ such that there is an optimal schedule $S$ in which job $k$ is preceded by jobs $j \leq k + \delta$ and followed by jobs $j > k + \delta$.
PRINCIPLE OF OPTIMALITY, Bellman 1956.
An optimal policy has the property that whatever the initial state and the initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.

Algorithm
Dynamic programming procedure: recursively the optimal solution for some job set \( J \) starting at time \( t \) is determined from the optimal solutions to subproblems defined by job subsets of \( S' \subset S \) with start times \( t' \geq t \).

\( J(j, l, k) \) contains all the jobs in a set \( \{ j, j+1, \ldots, l \} \) with processing time \( \leq p_k \)

V( \( J(j, l, k) \), t) total tardiness of the subset under an optimal sequence if this subset starts at time \( t \)

Initial conditions:
\( V(\emptyset, 1) = 0 \)
\( V(\{j\}, t) = \max(0, t + p_j - d_j) \)

Recursive conditions:
\( V(J(j,k), l) = \min_{j'}(V(J(j,k'), l) + \max(0, C_{j'}(\delta) - d_j', 0) + V(J(k'+1, l, k'), C_{j'}(\delta))) \)

where \( \delta \) is such that \( p_{\delta} = \max( p_{j'} | j' \in J(j, l, k) ) \)

Optimal value function is obtained for \( V(\{1, \ldots, n\}, 0) \)

Example
<table>
<thead>
<tr>
<th>jobs</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p_j )</td>
<td>121</td>
<td>79</td>
<td>147</td>
<td>83</td>
<td>130</td>
</tr>
<tr>
<td>( d_j )</td>
<td>260</td>
<td>266</td>
<td>266</td>
<td>316</td>
<td>337</td>
</tr>
</tbody>
</table>
\( k = 3, \ 0 \leq \delta \leq 2 \)
\( d_2 = 266 \)
\( V(\{1,2,\ldots,5\}, 0) = \min \{ V(J(1,3,3),0) + 81 + V(J(4,5,3), 347) \}
\( V(J(1,4,3),0) + 164 + V(J(5,5,3), 430) \}
\( V(J(1,5,3),0) + 294 + V(\emptyset, 560) \}
\( V(\emptyset, 560) = 0 \)
\( 1, 2, 4 \) forward algorithm
\( 1, 2, 4, 5 \) is NP hard
\( 1, 2, 4, 5 \) is NP hard, pseudo-polynomial algorithm based on dynamic programming

<table>
<thead>
<tr>
<th>jobs</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p_j )</td>
<td>4, 5</td>
<td>4, 5</td>
<td>4, 5</td>
<td>4, 5</td>
<td>4, 5</td>
</tr>
<tr>
<td>( T_j )</td>
<td>430 - 336 = 94</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( T_j )</td>
<td>560 - 337 = 229</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5, 4</td>
<td>( T_j ) = 477 - 337 = 140</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( T_j + T_j )</td>
<td>( T_j ) = 560 - 336 = 224</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( T_j + T_j )</td>
<td>( T_j ) = 364</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\( C_{\delta}(1, d_1) = 347 + 83 - 266 = 430 - 266 = 164 \)
\( C_{\delta}(2, d_2) = 430 + 130 - 266 = 560 - 266 = 294 \)
\( \{1, 2, \ldots, 5\}, 0 = 0 \) achieved with the sequence \( 1, 2, 4 \) and \( 2, 1, 4 \)
\( V(J(1, 4, 3), 0) = 0 \) achieved with the sequence \( 1, 2, 4, 5 \) and \( 2, 1, 4, 5 \)
\( V(J(5, 5, 3), 0) = 430 \) \( V(J(5, 5, 3), 0) = 223 \)

Optimal sequences:
\( 0 + 81 + 317 \)
\( 0 + 164 + 223 \)
\( 76 + 294 + 0 = 370 \)

Summary

- 1 \( \sum U_j \) is forward algorithm
- 1 \( \sum w_i U_j \) is NP hard
- 1 \( \sum T_j \) is NP hard, pseudo-polynomial algorithm based on dynamic programming