2-Machine Flow Shop

We wish to sequence $n$ jobs, each requiring processing on machine #1, followed by machine #2.

$P_{ij} =$ processing time for job $i$ on machine #j

makespan = total amount of time required to complete processing of all $n$ jobs

Objective: Sequence the jobs so as to minimize the makespan
**Example**

<table>
<thead>
<tr>
<th>JOB</th>
<th>Processing time (hrs)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Machine #1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
</tr>
</tbody>
</table>

Suppose we schedule the jobs for the sequence \(\{1, 2, 3, 4, 5, 6\}\)

Machine #1

<table>
<thead>
<tr>
<th></th>
<th>Machine #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>

Can we reduce the makespan by changing the sequence of the jobs?
**sequence \{1,2,3,4,5,6\}**

<table>
<thead>
<tr>
<th>Job</th>
<th>Machine 1</th>
<th>Machine 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>s</td>
<td>f</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>18</td>
</tr>
<tr>
<td>4</td>
<td>18</td>
<td>29</td>
</tr>
<tr>
<td>5</td>
<td>29</td>
<td>36</td>
</tr>
<tr>
<td>6</td>
<td>36</td>
<td>40</td>
</tr>
</tbody>
</table>

s = start time, f = finish time

Makespan = 54

---

**Johnson's Algorithm**

-an optimizing algorithm for scheduling the 2-machine flow shop, assuming that *no passing* is allowed (i.e., jobs are processed in the same sequence on both machines)

-construnctions a sequence by "growing" it from both ends (front and back)
**step 0**  Initialize $S_0 = S_1 = \emptyset$ and $I = \{1, 2, 3, \ldots, n\}$

**step 1**  Find $\min_{i \in \mathbb{I}, j \in \{1, 2\}} \{p_{ij} = p_{ij}\}$

**step 2**  If $\hat{j} = 1$, then $S_0 = S_0, \hat{i}$ (i.e., append job $\hat{i}$ to the beginning of the sequence)
Otherwise (i.e., $\hat{j} = 2$), $S_1 = \hat{i}, S_1$ (i.e., append job $\hat{i}$ to the end of the sequence)

**Step 3**  Remove $\hat{i}$ from $I$. If $I \neq \emptyset$ then go to step 1.
Else the optimal sequence is $S = S_0, S_1$

---

**EXAMPLE**

<table>
<thead>
<tr>
<th>JOB</th>
<th>Processing time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Machine 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
</tr>
</tbody>
</table>

$S_0 = S_1 = \emptyset$ and $I = \{1, 2, 3, 4, 5, 6\}$

Minimum $p_{ij}$ is $p_{11}$
Therefore,
$S_0 = \{1\}, S_1 = \emptyset$
$I = \{2, 3, 4, 5, 6\}$
Minimum $p_{ij}$ is $p_{22}$

Therefore, $S_0 = \{1\}$, $S_1 = \{2\}$

$I = \{3, 4, 5, 6\}$

<table>
<thead>
<tr>
<th>JOB</th>
<th>Processing time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Machine 1</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
</tr>
</tbody>
</table>

Minimum $p_{ij}$ is $p_{61}$

Therefore, $S_0 = \{1, 6\}$

$S_1 = \{2\}$

$I = \{3, 4, 5\}$

<table>
<thead>
<tr>
<th>JOB</th>
<th>Processing time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Machine 1</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
</tr>
</tbody>
</table>

1 6 2
Minimum \( p_{ij} \) is \( p_{42} \)

Therefore,

\( S_0 = \{1, 6\} \)
\( S_1 = \{4, 2\} \)

and

\( I = \{3, 5\} \)

\[
\begin{array}{c|cc}
\text{JOB} & \text{Processing time} & \\
 & \text{Machine 1} & \text{Machine 2} \\
\hline
3 & 9 & 7 \\
4 & 11 & \textcircled{5} \\
5 & 7 & 10
\end{array}
\]

Minimum \( p_{ij} \) is \( p_{51} = p_{32} \)

Therefore,

\( S_0 = \{1, 6, 5\} \)
\( S_1 = \{3, 4, 2\} \)

and \( I = \emptyset \)

\[
\begin{array}{c|cc}
\text{JOB} & \text{Processing time} & \\
 & \text{Machine 1} & \text{Machine 2} \\
\hline
3 & 9 & \textcircled{7} \\
5 & \textcircled{7} & 10
\end{array}
\]

The optimal sequence is

\( S = S_0 \cup S_1 \)

\[
\begin{array}{cccccccc}
1 & 6 & 5 & 3 & 4 & 2
\end{array}
\]
Optimal Sequence = \{1, 6, 5, 3, 4, 2\}

<table>
<thead>
<tr>
<th>Job</th>
<th>Machine 1</th>
<th>Machine 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>s f</td>
<td>s f</td>
</tr>
<tr>
<td>1</td>
<td>0 2</td>
<td>2 8</td>
</tr>
<tr>
<td>6</td>
<td>2 6</td>
<td>8 16</td>
</tr>
<tr>
<td>5</td>
<td>6 13</td>
<td>16 26</td>
</tr>
<tr>
<td>3</td>
<td>13 22</td>
<td>26 33</td>
</tr>
<tr>
<td>4</td>
<td>22 33</td>
<td>33 38</td>
</tr>
<tr>
<td>2</td>
<td>33 40</td>
<td>40 43</td>
</tr>
</tbody>
</table>

s = start time, f = finish time

Makespan = 43

Optimal Sequence = \{1, 6, 5, 3, 4, 2\}
3-Machine Flow Shop

Special conditions under which Johnson's Algorithm can be used to minimize the makespan:

- All jobs are to be processed on Machines #1, 2, & 3 in that order.
- The processing time on Machine #2 is dominated either by the time on Machine #1 or Machine #3.

\[
\begin{align*}
either & \quad \min \{p_{i1}\} \geq \max \{p_{i2}\} \\
or & \quad \min \{p_{i3}\} \geq \max \{p_{i2}\}
\end{align*}
\]

<table>
<thead>
<tr>
<th>JOB</th>
<th>Processing Times (hrs)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Machine A</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
</tr>
</tbody>
</table>

Processing times on machine 2 are dominated by those on machine 3.

5 = \min \{p_{i1}\} < \max \{p_{i2}\} = 6
6 = \min \{p_{i3}\} \geq \max \{p_{i2}\} = 6
Define two “dummy” machines, 1’ and 2’, with processing times:

\[ p_{i1'} = p_{i1} + p_{i2} \]
\[ p_{i2'} = p_{i2} + p_{i3} \]

Apply Johnson’s Algorithm to the two-machine problem with these two dummy machines. The resulting sequence is optimal for the 3-machine problem.

### Original Data:

<table>
<thead>
<tr>
<th>JOB</th>
<th>MACHINE</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>10</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>8</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>5</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>6</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>8</td>
<td>5</td>
<td>8</td>
</tr>
</tbody>
</table>

### “Dummy” Machine Data:

<table>
<thead>
<tr>
<th>JOB</th>
<th>MACHINE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>13</td>
</tr>
</tbody>
</table>
Johnson's Algorithm

Three-Machine Problem

Two "dummy machines" are defined:

Processing Times

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>16</td>
<td>13</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>5</td>
<td>13</td>
<td>13</td>
</tr>
</tbody>
</table>

The sequence found by this algorithm is: 3 4 5 1 2
The sequence is guaranteed to be optimal!

Optimal Schedule:

<table>
<thead>
<tr>
<th>Job</th>
<th>Machine 1</th>
<th>Machine 2</th>
<th>Machine 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>s</td>
<td>s</td>
<td>s</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>11</td>
<td>17</td>
</tr>
<tr>
<td>5</td>
<td>11</td>
<td>19</td>
<td>24</td>
</tr>
<tr>
<td>1</td>
<td>19</td>
<td>29</td>
<td>35</td>
</tr>
<tr>
<td>2</td>
<td>29</td>
<td>37</td>
<td>42</td>
</tr>
</tbody>
</table>

s = start time, f = finish time
Makespan = 48
**Optimal Sequence:**

Makespan = 48, Sequence: 3 4 5 1 2

---

**Compare with an arbitrary sequence:**

Makespan = 54, Sequence: 1 2 3 4 5
Random Job Sequencing Problem (M=5, N=3, seed = 662020)

5 Jobs, 3 Machines

Processing Times

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td>7</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>13</td>
<td>7</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>19</td>
<td>13</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
<td>12</td>
<td>17</td>
</tr>
</tbody>
</table>

Times on machine 2 are NOT dominated by either machine 1 or 3!

Johnson's Algorithm

Three-Machine Problem

Two "dummy machines" are defined:

Processing Times

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>23</td>
<td>23</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>19</td>
</tr>
<tr>
<td>4</td>
<td>25</td>
<td>32</td>
</tr>
<tr>
<td>5</td>
<td>28</td>
<td>29</td>
</tr>
</tbody>
</table>

The sequence found by this algorithm is: 1 2 4 5 3

Not guaranteed to be optimal!
Schedule found by Johnson's Algorithm:

Random Job Sequencing Problem (M=5, N=3, seed = 662020)

Makespan = 90, Sequence: 1 2 4 5 3

---

Schedule found by Johnson's Algorithm:

Random Job Sequencing Problem (M=5, N=3, seed = 662020)

<table>
<thead>
<tr>
<th>Job</th>
<th>Machine 1</th>
<th>Machine 2</th>
<th>Machine 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>s</td>
<td>f</td>
<td>s</td>
</tr>
<tr>
<td>1</td>
<td>0 6</td>
<td>6 7</td>
<td>7 13</td>
</tr>
<tr>
<td>2</td>
<td>6 22</td>
<td>22 29</td>
<td>29 45</td>
</tr>
<tr>
<td>4</td>
<td>22 28</td>
<td>29 48</td>
<td>48 61</td>
</tr>
<tr>
<td>5</td>
<td>28 44</td>
<td>48 60</td>
<td>61 78</td>
</tr>
<tr>
<td>3</td>
<td>44 57</td>
<td>60 67</td>
<td>78 90</td>
</tr>
</tbody>
</table>

s = start time, f = finish time
Makespan = 90
Branch-and-Bound Algorithm for 3-Machine Flowshop Problem

-suggested by Ignall & Schrage

Suppose that the first $r$ jobs in the sequence have been tentatively fixed:

$$J_r = \{ j_1, j_2, \ldots, j_r \}$$

Denote by $\overline{J}_r$ the set of $(n-r)$ jobs not yet sequenced.

Let $\text{TIME}_1(J_r)$, $\text{TIME}_2(J_r)$, and $\text{TIME}_3(J_r)$ be the times at which machines 1, 2, & 3 (respectively) complete processing the jobs in $J_r$.

Lower Bounds on makespan of all completions of the partial sequence $J_r$:

\[
\text{TIME}_1(J_r) + \sum_{i \in \overline{J}_r} p_{i1} + \min_{i \in \overline{J}_r} \{ p_{i2} + p_{i3} \}
\]

Makespan if the job with shortest processing times on machines 2&3 need not wait

\[
\text{TIME}_2(J_r) + \sum_{i \in \overline{J}_r} p_{i2} + \min_{i \in \overline{J}_r} \{ p_{i3} \}
\]

Makespan if the job with least time on machine #3 need not wait

\[
\text{TIME}_3(J_r) + \sum_{i \in \overline{J}_r} p_{i3}
\]

Makespan if no job needs to wait for machine #3
Example

Suppose

\[ J_1 = \{1\} \]

<table>
<thead>
<tr>
<th>JOB</th>
<th>Machine Processing Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>14</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
</tr>
</tbody>
</table>

\[ \text{TIME}1(J_1) = 14 \]

\[ \text{TIME}2(J_1) = 26 \]

\[ \text{TIME}3(J_1) = 39 \]

\[ J_1 = \{1\} \]

\[ \text{TIME}3(J_1) = 39 \]

\[ \text{TIME}3(J_r) + \sum_{i \in \overline{J}_r} p_{i3} \]

<table>
<thead>
<tr>
<th>JOB</th>
<th>Machine Processing Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>14</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
</tr>
</tbody>
</table>

\[ \text{Lower Bound #3:} \]

\[ 39 + 6 = 45 \]
\[ J_1 = \{1\} \]
\[ \text{TIME2}(J_1) = 26 \]
\[ \text{TIME2}(J_r) + \sum_{i \in J_r} p_{i2} + \min_{i \in J_r} \{p_{i3}\} \]

<table>
<thead>
<tr>
<th>Job</th>
<th>Machine Processing Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>14</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
</tr>
</tbody>
</table>

Lower Bound #2:
\[ 26 + 26 + 1 = 53 \]

\[ J_1 = \{1\} \]
\[ \text{TIME1}(J_1) = 14 \]
\[ \text{TIME1}(J_r) + \sum_{i \in J_r} p_{i1} + \min_{i \in J_r} \{p_{i2} + p_{i3}\} \]

<table>
<thead>
<tr>
<th>Job</th>
<th>Machine Processing Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>14</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
</tr>
</tbody>
</table>

Lower Bound #1:
\[ 14 + 15 + (1+1) = 31 \]
Branch-and-Bound Algorithm for 3-Machine Flowshop Problem

Branching will be done by choosing the next job to be added to the end of sequence $J_r$:

$$J_{r+1} = J_r \cup \{i_{r+1}\}$$

for each possible value of $i_{r+1} \in J_r$

Random Job Sequencing Problem ($M=5$, $N=3$, seed = 662020)

Using Johnson's algorithm, job sequence is: 1 2 4 5 3
Makespan is 90, our original incumbent.

We now begin the branch-and-bound algorithm

*** New incumbent: 89 ***

Optimal sequence is 1 4 2 3 5  
CPU time = 31.8 sec.  
# subproblems enumerated = 53

--- compared to

$5! = 120$

total sequences

(44.2%)
## Optimal Schedule

Random Job Sequencing Problem (M=5, N=3, seed = 662020)

<table>
<thead>
<tr>
<th>Job</th>
<th>Machine 1</th>
<th>Machine 2</th>
<th>Machine 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>s f</td>
<td>s f</td>
<td>s f</td>
</tr>
<tr>
<td>4</td>
<td>0 6</td>
<td>6 7</td>
<td>7 13</td>
</tr>
<tr>
<td>2</td>
<td>12 28</td>
<td>31 38</td>
<td>44 60</td>
</tr>
<tr>
<td>3</td>
<td>28 41</td>
<td>41 48</td>
<td>60 72</td>
</tr>
<tr>
<td>5</td>
<td>41 57</td>
<td>57 69</td>
<td>72 89</td>
</tr>
</tbody>
</table>

s = start time, f = finish time
Makespan = 89

## Optimal Schedule:

Makespan = 89, Sequence: 1 4 2 3 5
Random Job Sequencing Problem (M=5, N=3, seed = 662020)

Using Johnson's algorithm, job sequence is: 1 2 4 5 3
Makespan is 90, our original incumbent.

We now begin the branch-and-bound algorithm

Subproblem number 0: J=

Subproblem number 1: J= 1
Completion times: 6 7 13
Lower bounds: 76 64 71

Subproblem number 2: J= 1 2
Completion times: 22 29 45
Lower bounds: 76 79 87
Subproblem number 3: J = 1 2 3
Completion times: 35 42 57
Lower bounds: 86 86 87

Subproblem number 4: J = 1 2 3 4
Completion times: 41 61 74
Lower bounds: 86 90 91
--- Fathomed by bound ---

Subproblem number 5: J = 1 2 3 5
Completion times: 51 63 80
Lower bounds: 89 95 93
--- Fathomed by bound ---
--- Subproblem 3: Fathomed by enumeration ---

Subproblem number 6: J = 1 2 4
Completion times: 28 48 61
Lower bounds: 76 79 90
--- Fathomed by bound ---
Subproblem number 7: J = 1 2 5  
Completion times: 38 50 67  
Lower bounds: 76 88 92  
--- Fathomed by bound ---  

--- Subproblem 2: Fathomed by enumeration ---  

Subproblem number 8: J = 1 3  
Completion times: 19 26 38  
Lower bounds: 80 77 84  
Subproblem number 9: J = 1 3 2  
Completion times: 35 42 58  
Lower bounds: 86 86 88  
Subproblem number 10: J = 1 3 2 4  
Completion times: 41 61 74  
Lower bounds: 86 90 91  
--- Fathomed by bound ---  
Subproblem number 11: J = 1 3 2 5  
Completion times: 51 63 80  
Lower bounds: 89 95 93  
--- Fathomed by bound ---  
--- Subproblem 9: Fathomed by enumeration ---
Subproblem number 12: J = 1 3 4
Completion times: 25 45 58
Lower bounds: 80 80 91
--- Fathomed by bound ---

Subproblem number 13: J = 1 3 5
Completion times: 35 47 64
Lower bounds: 80 86 93
--- Fathomed by bound ---
--- Subproblem 8: Fathomed by enumeration ---

Subproblem number 14: J = 1 4
Completion times: 12 31 44
Lower bounds: 76 69 89

Subproblem number 15: J = 1 4 2
Completion times: 28 38 60
Lower bounds: 76 69 89

Subproblem number 16: J = 1 4 2 3
Completion times: 41 48 72
Lower bounds: 86 77 89

--- Subproblem 16: Fathomed by enumeration ---

NEW INCUMBENT! 

Subproblem number 17: J = 1 4 2 3 5
Completion times: 57 60 89
*** New incumbent: 89 ***

Subproblem number 18: J = 1 4 2 5
Completion times: 44 56 77
Lower bounds: 76 75 89
--- Fathomed by bound ---
--- Subproblem 15: Fathomed by enumeration ---

Subproblem number 19: J = 1 4 3
Completion times: 25 38 56
Lower bounds: 80 73 89
--- Fathomed by bound ---

Subproblem number 20: J = 1 4 5
Completion times: 28 43 61
Lower bounds: 76 69 89
--- Fathomed by bound ---
--- Subproblem 14: Fathomed by enumeration ---
Subproblem number 21: J = 1 5
Completion times: 22 34 51
Lower bounds: 76 79 92
--- Fathomed by bound ---
--- Subproblem 1: Fathomed by enumeration ---

Subproblem number 22: J = 2
Completion times: 16 23 39
Lower bounds: 64 68 87

Subproblem number 23: J = 2 1
Completion times: 22 24 45
Lower bounds: 76 74 87

Subproblem number 24: J = 2 1 3
Completion times: 35 42 57
Lower bounds: 86 86 87

Subproblem number 25: J = 2 1 3 4
Completion times: 41 61 74
Lower bounds: 86 90 91
--- Fathomed by bound ---

Subproblem number 26: J = 2 1 3 5
Completion times: 51 63 80
Lower bounds: 89 95 93
--- Fathomed by bound ---
--- Subproblem 24: Fathomed by enumeration ---

Subproblem number 27: J = 2 1 4
Completion times: 28 47 60
Lower bounds: 76 78 89
--- Fathomed by bound ---

Subproblem number 28: J = 2 1 5
Completion times: 38 50 67
Lower bounds: 76 88 92
--- Fathomed by bound ---
--- Subproblem 23: Fathomed by enumeration ---

Subproblem number 29: J = 2 3
Completion times: 29 36 51
Lower bounds: 64 74 87
Subproblem number 30: J = 2 3 1
Completion times: 35 37 57
Lower bounds: 86 81 87

Subproblem number 31: J = 2 3 1 4
Completion times: 41 60 73
Lower bounds: 86 89 90
--- Fathomed by bound ---

Subproblem number 32: J = 2 3 1 5
Completion times: 51 63 80
Lower bounds: 89 95 93
--- Fathomed by bound ---

--- Subproblem 30: Fathomed by enumeration ---

Subproblem number 33: J = 2 3 4
Completion times: 35 55 68
Lower bounds: 64 74 91
--- Fathomed by bound ---

Subproblem number 34: J = 2 3 5
Completion times: 45 57 74
Lower bounds: 64 83 93
--- Fathomed by bound ---

--- Subproblem 29: Fathomed by enumeration ---

Subproblem number 35: J = 2 4
Completion times: 22 42 55
Lower bounds: 64 68 90
--- Fathomed by bound ---

Subproblem number 36: J = 2 5
Completion times: 32 44 61
Lower bounds: 64 77 92
--- Fathomed by bound ---

--- Subproblem 22: Fathomed by enumeration ---

Subproblem number 37: J = 3
Completion times: 13 20 32
Lower bounds: 64 65 84

Subproblem number 38: J = 3 1
Completion times: 19 21 38
Lower bounds: 80 72 84

Subproblem number 39: J = 3 1 2
Completion times: 35 42 58
Lower bounds: 86 86 88
Subproblem number 40: J = 3 1 2 4  
Completion times: 41 61 74  
Lower bounds: 86 90 91  
--- Fathomed by bound ---

Subproblem number 41: J = 3 1 2 5  
Completion times: 51 63 80  
Lower bounds: 89 95 93  
--- Fathomed by bound ---

--- Subproblem 39: Fathomed by enumeration ---

Subproblem number 42: J = 3 1 4  
Completion times: 25 44 57  
Lower bounds: 80 79 90  
--- Fathomed by bound ---

Subproblem number 43: J = 3 1 5  
Completion times: 35 47 64  
Lower bounds: 80 86 93  
--- Fathomed by bound ---

--- Subproblem 38: Fathomed by enumeration ---

Subproblem number 44: J = 3 2  
Completion times: 29 36 52  
Lower bounds: 64 74 88  

Subproblem number 45: J = 3 2 1  
Completion times: 35 37 58  
Lower bounds: 86 81 88  

Subproblem number 46: J = 3 2 1 4  
Completion times: 41 60 73  
Lower bounds: 86 89 90  
--- Fathomed by bound ---

Subproblem number 47: J = 3 2 1 5  
Completion times: 51 63 80  
Lower bounds: 89 95 93  
--- Fathomed by bound ---

--- Subproblem 45: Fathomed by enumeration ---

Subproblem number 48: J = 3 2 4  
Completion times: 35 55 68  
Lower bounds: 64 74 91  
--- Fathomed by bound ---
Subproblem number 49: J = 3 2 5
Completion times: 45 57 74
Lower bounds: 64 83 93
--- Fathomed by bound ---
--- Subproblem 44: Fathomed by enumeration ---

Subproblem number 50: J = 3 4
Completion times: 19 39 52
Lower bounds: 64 65 91
--- Fathomed by bound ---

Subproblem number 51: J = 3 5
Completion times: 29 41 58
Lower bounds: 64 74 93
--- Fathomed by bound ---
--- Subproblem 37: Fathomed by enumeration ---

Subproblem number 52: J = 4
Completion times: 6 25 38
Lower bounds: 64 58 89
--- Fathomed by bound ---

Subproblem number 53: J = 5
Completion times: 16 28 45
Lower bounds: 64 68 92
--- Fathomed by bound ---
--- Subproblem 0: Fathomed by enumeration ---

Optimal sequence is 1 4 2 3 5

# subproblems enumerated = 53