FLOWSHOP SCHEDULING

This Hypercard stack was prepared by:
Dennis L. Bricker,
Dept. of Industrial Engineering,
University of Iowa,
Iowa City, Iowa 52242
e-mail: dbricker@icaen.uiowa.edu

author

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

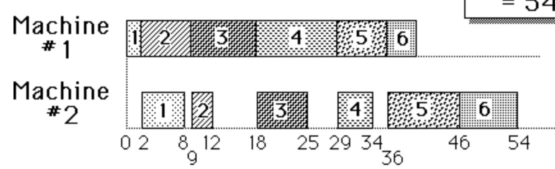
Objective: Sequence the jobs so as to minimize the makespan

EXAMPLE

JOB	Processing time (hrs) Machine #1	Processing time (hrs) Machine #2
1	2	6
2	7	3
3	9	7
4	11	5
5	7	10
6	4	8

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

makespan = 54



Can we reduce the makespan by changing the sequence of the jobs?

sequence {1,2,3,4,5,6}

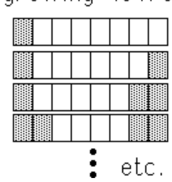
Job i	Machine 1 s	Machine 1 f	Machine 2 s	Machine 2 f
1	0	2	2	8
2	2	9	9	12
3	9	18	18	25
4	18	29	29	34
5	29	36	36	46
6	36	40	46	54

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)

-construsts a sequence by "growing" it from both ends (front and back)



etc.

step 0 Initialize $S_0 = S_1 = \varnothing$ and I={1,2,3,...n}

step 1 Find $\underset{i\in I,\, j\in\{1,2\}}{\textbf{minimum}} \{p_{ij}\} = p_{\hat{i}\hat{j}}$

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 $I \neq \varnothing$ then go to step 1.
Else the optimal sequence is $S = S_0, S_1$

Example

EXAMPLE

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

Minimum p_{ij} is p_{11}

Therefore, S_0={1}, $S_1 = \varnothing$
I={2,3,4,5,6}

JOB	Processing time Machine 1	Processing time Machine 2
1	2	6
2	7	3
3	9	7
4	11	5
5	7	10
6	4	8

1					

Minimum p_{ij} is p_{22}

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

JOB	Processing time Machine 1	Machine 2
2	7	3
3	9	7
4	11	5
5	7	10
6	4	8

1					2

Minimum p_{ij} is p_{61}

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

JOB	Processing time Machine 1	Machine 2
3	9	7
4	11	5
5	7	10
6	4	8

1	6				2

Minimum p_{ij} is p_{42}

Therefore,
$S_0=\{1,6\}$
$S_1=\{4,2\}$
and

$I=\{3,5\}$

JOB	Processing time Machine 1	Machine 2
3	9	7
4	11	5
5	7	10

1	6			4	2

Minimum p_{ij} is $p_{51} = p_{32}$

Therefore,
$S_0=\{1,6,5\}$
$S_1=\{3,4,2\}$

and $I = \varnothing$

JOB	Processing time Machine 1	Machine 2
3	9	7
5	7	10

The optimal sequence is
$\mathbf{S} = \mathbf{S}_0 \cup \mathbf{S}_1$

=

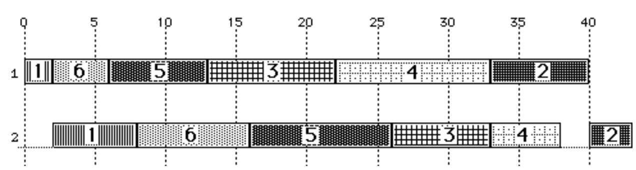
1	6	5	3	4	2

Optimal Sequence = {1,6,5,3,4,2}

Job i	Machine 1 s	Machine 1 f	Machine 2 s	Machine 2 f
1	0	2	2	8
6	2	6	8	16
5	6	13	16	26
3	13	22	26	33
4	22	33	33	38
2	33	40	40	43

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 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.

either $\min\{p_{i1}\} \geq \max\{p_{i2}\}$
or $\min\{p_{i3}\} \geq \max\{p_{i2}\}$

JOB	Processing Times (hrs) Machine A	Machine B	Machine C
1	10	6	7
2	8	2	6
3	5	2	10
4	6	6	7
5	8	5	8

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

EXAMPLE

$5 = \min\{p_{i1}\} < \max\{p_{i2}\} = 6$
$6 = \min\{p_{i3}\} \geq \max\{p_{i2}\} = 6$

Application of Johnson's Algorithm to 3-Machine Problem

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:

JOB	MACHINE A	MACHINE B	MACHINE C
1	10	6	7
2	8	2	6
3	5	2	10
4	6	6	7
5	8	5	8

"Dummy" Machine Data:

JOB	MACHINE 1'	MACHINE 2'
1	16	13
2	10	8
3	7	12
4	12	13
5	13	13

Johnson's Algorithm

Three-Machine Problem

Two "dummy machines" are defined:

Processing Times

i	1	2
1	16	13
2	10	8
3	7	12
4	12	13
5	13	13

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

The sequence is guaranteed to be optimal!

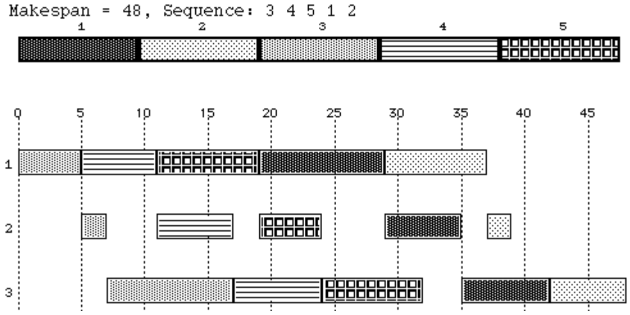
Optimal Schedule:

Job i	Machine 1 s	Machine 1 f	Machine 2 s	Machine 2 f	Machine 3 s	Machine 3 f
3	0	5	5	7	7	17
4	5	11	11	17	17	24
5	11	19	19	24	24	32
1	19	29	29	35	35	42
2	29	37	37	39	42	48

s = start time, f = finish time
Makespan = 48

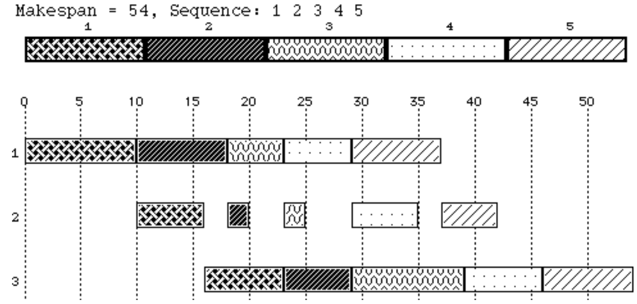
Optimal Sequence:

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

i	1	2	3
1	6	1	6
2	16	7	16
3	13	7	12
4	6	19	13
5	16	12	17

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

Johnson's Algorithm

Three-Machine Problem

Two "dummy machines" are defined:

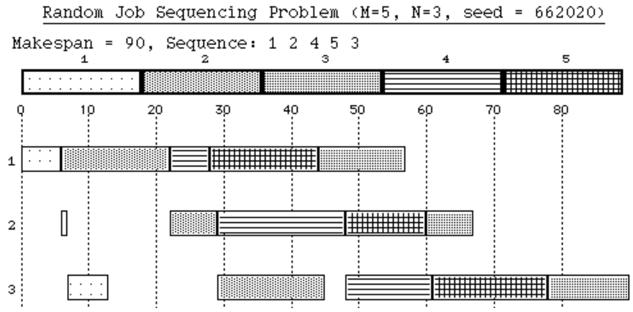
Processing Times

i	1	2
1	7	7
2	23	23
3	20	19
4	25	32
5	28	29

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

Not guaranteed to be optimal!

Schedule found by Johnson's Algorithm:



Schedule found by Johnson's Algorithm:

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

Job	Machine 1		Machine 2		Machine 3	
i	s	f	s	f	s	f
1	0	6	6	7	7	13
2	6	22	22	29	29	45
4	22	28	29	48	48	61
5	28	44	48	60	61	78
3	44	57	60	67	78	90

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:

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

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

Let TIME1(J_r), TIME2(J_r), and TIME3(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 :

$$\mathrm{TIME1}(J_r) + \sum_{i\in\bar{J}_r} p_{i1} + \min_{i\in\bar{J}_r}\{p_{i2}+p_{i3}\}$$

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

$$\mathrm{TIME2}(J_r) + \sum_{i\in\bar{J}_r} p_{i2} + \min_{i\in\bar{J}_r}\{p_{i3}\}$$

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

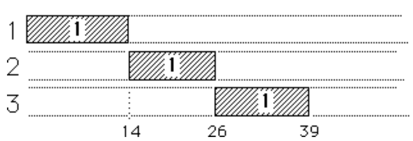
$$\mathrm{TIME3}(J_r) + \sum_{i\in\bar{J}_r} p_{i3}$$

Makespan if no job needs to wait for machine #3

Example

Suppose $J_1 = \{1\}$

JOB	Machine Processing Time		
i	1	2	3
1	14	12	13
2	8	5	3
3	1	20	2
4	6	1	1



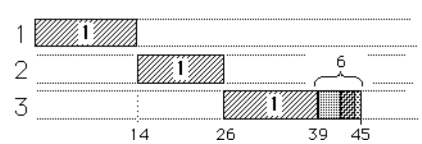
TIME1(J_1) = 14
TIME2(J_1) = 26
TIME3(J_1) = 39

$J_1 = \{1\}$

TIME3(J_1) = 39

$$\mathrm{TIME3}(J_r) + \sum_{i\in\bar{J}_r} p_{i3}$$

JOB	Machine Processing Time			
i	1	2	3	
1	14	12	13	
2	8	5	3	
3	1	20	2	6
4	6	1	1	



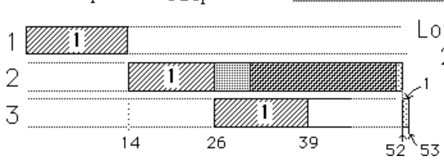
Lower Bound #3:
39 + 6 = 45

$J_1 = \{1\}$

TIME2(J_1) = 26

$$\mathrm{TIME2}(J_r) + \sum_{i\in\bar{J}_r} p_{i2} + \min_{i\in\bar{J}_r}\{p_{i3}\}$$

JOB	Machine Processing Time			
i	1	2	3	
1	14	12	13	
2	8	5	3	
3	1	20	2	6
4	6	1	1	



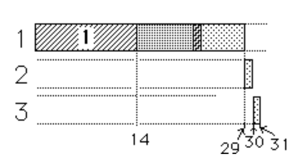
Lower Bound #2:
26 + 26 + 1=53

$J_1 = \{1\}$

TIME1(J_1) = 14

$$\mathrm{TIME1}(J_r) + \sum_{i\in\bar{J}_r} p_{i1} + \min_{i\in\bar{J}_r}\{p_{i2}+p_{i3}\}$$

JOB	Machine Processing Time			
i	1	2	3	
1	14	12	13	
2	8	5	3	
3	1	20	2	6
4	6	1	1	



Lower Bound #1:

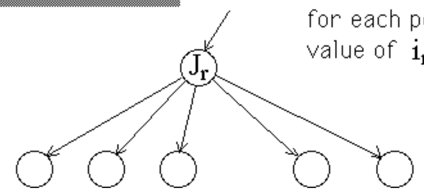
14 + 15 + (1+1) = 31

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 $i_{r+1} \in \bar{J}_r$



Branch-and-Bound
5-job, 3-machine
Sequencing Problem

```
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-&-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

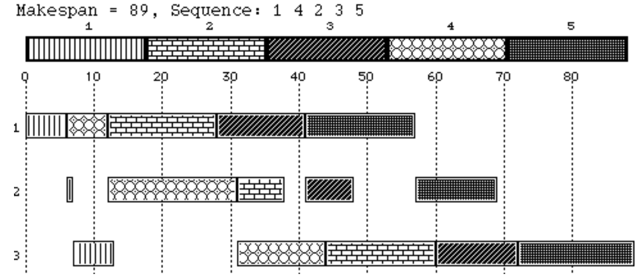
Random Job Sequencing Problem (M=5, N=3, seed = 662020)

Job	Machine 1		Machine 2		Machine 3	
i	s	f	s	f	s	f
1	0	6	6	7	7	13
4	6	12	12	31	31	44
2	12	28	31	38	44	60
3	28	41	41	48	60	72
5	41	57	57	69	72	89

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

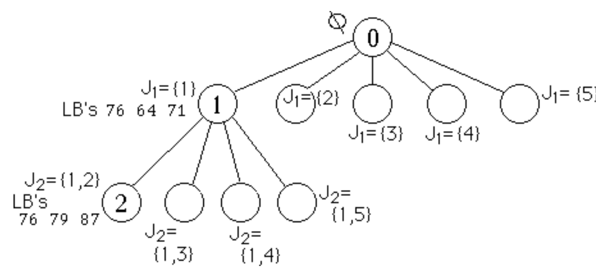
Optimal Schedule:

Makespan = 89, Sequence: 1 4 2 3 5

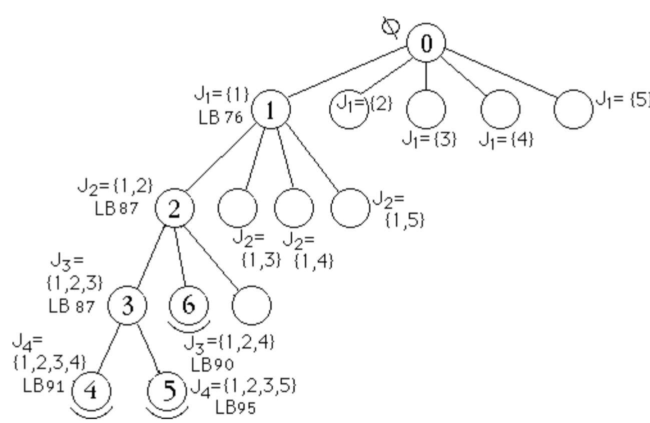


Branch-and-Bound
5-job, 3-machine
Sequencing Problem

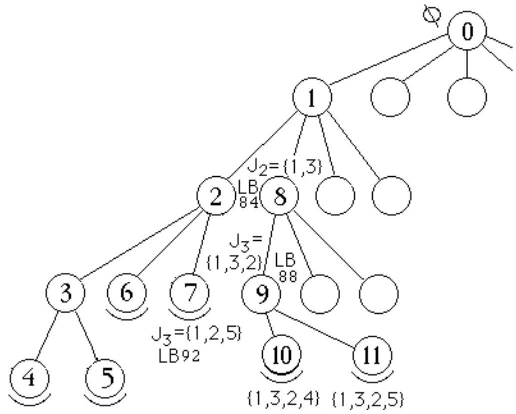
```
Random Job Sequencing Problem (M=5, N=3, seed = 662020)

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

We now begin the branch-&-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
```

NEW INCUMBENT!

```
                Subproblem number 17: J= 1 4 2 3 5
                Completion times: 57 69 89
                *** New incumbent: 89 ***
           --- Subproblem 16: Fathomed by enumeration

           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
```