Branch and Bound

- Enumerative method
- Guarantees finding the best schedule
- Basic idea:
  - Look at a set of schedules
  - Develop a bound on the performance
  - Discard (fathom) if bound worse than best schedule found before

Classic Result

- The EDD rule is optimal for $1\| L_{\text{max}}$
- If jobs have different release dates, which we denote $1|r_j| L_{\text{max}}$
  
  then the problem is NP-Hard
- What makes it so much more difficult?
Delay Schedule

Add a delay

What makes this problem hard is that the optimal schedule is not necessarily a non-delay schedule
Final Classic Result

- The preemptive EDD rule is optimal for the preemptive (prmp) version of the problem

\[ 1|r_j, \text{prmp}|L_{\text{max}} \]

- Note that in the previous example, the preemptive EDD rule gives us the optimal schedule

Branch and Bound

- The problem

\[ 1|r_j|L_{\text{max}} \]

cannot be solved using a simple dispatching rule so we will try to solve it using branch and bound

- To develop a branch and bound procedure:
  - Determine how to branch
  - Determine how to bound
### Data

<table>
<thead>
<tr>
<th>Jobs</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_j$</td>
<td>4</td>
<td>2</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>$r_j$</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>$d_j$</td>
<td>8</td>
<td>12</td>
<td>11</td>
<td>10</td>
</tr>
</tbody>
</table>
Branch and bound

- Enumeration in a search tree
  - Each node is a partial solution, i.e. a part of the solution space

Branching

- Root node
- Child nodes
- Levels 0, 1, 2

Branching examples:

- (•,•,•,•)
- (1,*•,•,•)
- (2,*•,•,•)
- (3,*•,•,•)
- (4,*•,•,•)
Branching

Discard immediately because
\[ r_3 = 3 \]
\[ r_4 = 5 \]

Need to develop lower bounds on these nodes and do further branching.
Bounding (in general)

- Typical way to develop bounds is to relax the original problem to an easily solvable problem.
- Three cases:
  - If there is no solution to the relaxed problem there is no solution to the original problem.
  - If the optimal solution to the relaxed problem is feasible for the original problem then it is also optimal for the original problem.
  - If the optimal solution to the relaxed problem is not feasible for the original problem it provides a bound on its performance.

Relaxing the Problem

- The problem $1|r_j, prmp|L_{\text{max}}$ is a relaxation to the problem $1|r_j|L_{\text{max}}$.
- Not allowing preemption is a constraint in the original problem but not the relaxed problem.
- We know how to solve the relaxed problem (preemptive EDD rule).
Bounding

- Preemptive EDD rule optimal for the preemptive version of the problem
- Thus, solution obtained is a lower bound on the maximum delay
- If preemptive EDD results in a non-preemptive schedule all nodes with higher lower bounds can be discarded.

Lower Bounds

- Start with (1,•,•,•): $r_4 = 5$
  - Job with EDD is Job 4 but
  - Second earliest due date is for Job 3

![Diagram showing job due dates and maximum delay](image)
Branching

$L_{\text{max}} \geq 5$

$(1,\bullet,\bullet,\bullet)$

$(2,\bullet,\bullet,\bullet)$

$(3,\bullet,\bullet,\bullet)$

$(4,\bullet,\bullet,\bullet)$

$L_{\text{max}} \geq 6$

$(1,2,\bullet,\bullet)$

$(1,3,\bullet,\bullet)$

$(1,3,4,2)$

$L_{\text{max}} = 5$

$(1,3,4,2)$