Branch and Bound

- One way of dealing with an NP-complete problem.
- Not polynomial.
- Complete enumeration in the worst case, but uses a clever idea to cut off much of the search space.

Branch and Bound for minimizing (0-1) IPs
- Solve an LP relaxation
- Choose a variable $x_i$ with value strictly between 0 and 1.
- Branch on whether $x_i = 0$ or $x_i = 1$.
- Bound
  - Generate feasible solutions to the IP
  - At each node generate a lower bound (using the LP) on the solutions in that branch.
  - If the lower bound on a branch is better than a feasible solution, kill that branch.
- When there is no more branching to be done, stop.
Dealing w/ NP-complete problems

1. Exact solution
2. Fast algorithm
3. Works for all inputs
   
   For NP-C problem, give up on 1 of these features
Minimizing
Real feasible solution

Int. scan 10
Int. sol. 7

Different problems

Branching

LP, value = 10.5

Obj. 21/2

2 different problems
Branch and Bound

Combinatorial Branch and Bound for a minimization problem

- Enumerate possible values for a decision.
- Branch on all the possibilities
- Bound
  - Generate feasible solutions to original problem.
  - At each node generate a lower bound (using a relaxation) on the solutions in that branch.
  - If the lower bound on a branch is better than a feasible solution, kill that branch.
- When there is no more branching to be done, stop.
Branch and Bound for \( 1|r_j|L_{\text{max}} \)

- Branch on “which job runs in position \( i \).
- Use \( 1|r_j, pmtn|L_{\text{max}} \) as a lower bound

Example

<table>
<thead>
<tr>
<th>( j )</th>
<th>( r_j )</th>
<th>( p_j )</th>
<th>( d_j )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>6</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>

Get feasible solutions

Preemptive EDD

Among the jobs that have been released, run the one with earliest due date