Simulation

Lecture O2
Optimization: Integer Programming

Saeed Bastani
saeed.bastani@eit.lth.se

Spring 2017
Outline

✓ Introduction to Integer Programming (IP)
✓ Examples of IP
✓ Developing IP
✓ Branch and Bound (B&B) Method
✓ B&B Example (Minimization Problem)
✓ B&B Example (Maximization Problem)
✓ Solving IPs in MATLAB
One assumption of linear programming is that decision variables can take on fractional values such as $X_1 = 0.33$ or $X_3 = 1.57$. Yet a large number of business problems can be solved only if variables have integer values.

When an airline decides how many planes to purchase, it cannot place an order for 5.38 aircraft; it must order 4, 5, 6, or some other integer amount.
Introduction

- **Integer Programs (IP):** \( X \subseteq \mathbb{Z}^n \)
  - (NP-hard) computational complexity
- **Mixed Integer Linear Program (MILP):**
  - Generally (NP-hard)

\[
\begin{align*}
\text{min} & \quad c^T x \\
\text{s.t.} & \quad Ax \leq b \\
\text{where,} & \quad X \subseteq \mathbb{Z}^{n_i} \times \mathbb{R}^{n_r}
\end{align*}
\]

- However, many problems can be solved surprisingly quickly!
(Mixed) Integer Programming

• Integer Programming:
  ➢ all variables must have Integer values
• Mixed Integer Programming:
  ➢ some variables have integer values

Exponential solution times!
Example IP formulation

The Knapsack problem:

I wish to select items to put in my backpack.

- There are \( m \) items available.
- Item \( i \) weights \( w_i \) kg,
- Item \( i \) has value \( v_i \).
- I can carry \( Q \) kg.

Let \( x_i = \begin{cases} 1 & \text{if I select item } i \\ 0 & \text{otherwise} \end{cases} \)

\[
\begin{align*}
\text{max} & \quad \sum_{i} x_i v_i \\
\text{s.t.} & \quad \sum_{i} x_i w_i \leq Q \\
& \quad x_i \in \{0,1\}
\end{align*}
\]
Task Allocation

- $n$ jobs, $m$ machines
- Job $i$ has a load of $q_i$ (e.g. amount of CPU resource)
- The cost of doing job $i$ by machine $j$ is $c_{ij}$
- The load capacity of machine $j$ is $Q_j$

Objective: assign all jobs with a minimum total cost
Formulation

\[ x_{ij} = \begin{cases} 
1 & \text{if task } i \text{ is assigned to machine } j \\
0 & \text{otherwise}
\end{cases} \]

\[
\min \sum_{i,j} c_{ij}x_{ij}
\]

\[
\sum_j x_{ij} = 1 \quad \forall i
\]

\[
\sum_i x_{ij}q_i \leq Q_j \quad \forall j
\]
Vehicle Routing Problem (VRP)

What is the optimal set of routes for a fleet of vehicles to traverse in order to deliver to a given set of customers?

- \( n \) customers and \( m \) vehicles
- \( c_{i,j} \) – the distance or cost of travel from \( i \) to \( j \)
- \( q_j \) – load at \( j \)
- \( Q_k \) – capacity of vehicle \( k \)

What vehicle should visit each customer, and in what order, to minimize costs?

If \( m = 1 \) vehicle \( \rightarrow \) Travel Salesman Problem (TSP)
Traditional formulation

\[ x_{ijk} = \begin{cases} 
1 & \text{if } i \text{ precedes } j \text{ on vehicle } k \\
0 & \text{otherwise}
\end{cases} \]

\[
\text{minimize } \sum_{i,j,k} c_{ij} x_{ijk}
\]

\[
\sum_j \sum_k x_{ijk} = 1 \quad \forall i
\]

\[
\sum_j x_{ijk} = \sum_i x_{ijk} \quad \forall k
\]

\[
\sum_i \sum_j x_{ijk} q_j \leq Q_k \quad \forall k
\]

...
Logical constraints in IP

- If $x$ then not $y$ (assume $x, y \in \{0, 1\}$): \[(1 - x) M \geq y\]
  ($M$ is “big $M$” – a large value – larger than any feasible value for $y$)

- $x$ or $y$ or both ($x, y \in \{0, 1\}$): \[x + y \geq 1\]

- $x \leq 1$ or $x \geq 5$ ($x$ is real number):
  - define a binary variable $w \in \{0, 1\}$
    - if $w = 1$ \[x \leq 1 + M(1-w)\]
    - if $w = 0$ \[x \geq 5 - Mw\]

- $x + 2y \geq 10$ or $4x - 10y \leq 2$ ($x$ and $y$ are real numbers):
  - define a binary variable $w \in \{0, 1\}$ and big $M$
    - if $w = 1$ \[x + 2y \geq 10 - M(1-w)\]
    - if $w = 0$ \[4x - 10y \leq 2 + Mw\]
IP Formulation Tricks (2)

- For the purpose of this course, LP formulation is highly crucial. In your homework, you will be asked to do the formulation.
- So, start learning the tricks by practice!
- Find out interesting tricks here:
  

And here

Solving IPs

How can we solve IPs problems?
Solving IP

• Some problem classes have the “Integrality Property”: all solution naturally fall on integer points e.g.
  – Maximum Flow problems
  – Assignment problems

• If the constraint matrix has a special form, it will have the Integrality Property:
  – Totally unimodular
  – Balanced
  – Perfect

• But, not all problems have such properties
Maximize \( Z = 100x_1 + 150x_2 \)
subject to:
\[
8,000x_1 + 4,000x_2 \leq 40,000 \\
15x_1 + 30x_2 \leq 200 \\
x_1, x_2 \geq 0 \text{ and integer}
\]

Optimal Solution:
\[
Z = $1,055.56 \\
x_1 = 2.22 \\
x_2 = 5.55
\]

OBS! We get non-integer solution

Feasible Solution Space with Integer Solution Points
Solving IP

• How about solving LP Relaxation followed by rounding?
Solving IP

- In general, rounding does not work!

- LP solution provides lower bound (for minimization) and upper bound (for maximization) on IP

- But, rounding can be arbitrarily far away from integer solution
Solving IP

- Combine both approaches
  - Solve LP Relaxation to get fractional solutions
  - Create two sub-branches by adding constraints
Solving IP

- Combine both approaches
  - Solve LP Relaxation to get fractional solutions
  - Create two sub-branches by adding constraints
Solving IP

- Combine both approaches
  - Solve LP Relaxation to get fractional solutions
  - Create two sub-branches by adding constraints
An Example Maximization Problem

It takes 2 hours to wire each chandelier and 3 hours to wire a ceiling fan.

Final assembly of the chandeliers and fans requires 6 and 5 hours, respectively.

The production capability is such that only 12 hours of wiring time and 30 hours of assembly time are available.

If each chandelier produced nets the firm $7.00 and each fan $6.00, the Production mix decision can be formulated using LP as follows:
Harrison Electric Company

Maximize profit = $7.00 X_1 + $6.00 X_2

subject to:

\[ 2X_1 + 3X_2 \leq 12 \] (wiring hours)
\[ 6X_1 + 5X_2 \leq 30 \] (assembly hours)

\[ X_1, X_2 \geq 0 \]

where:

\[ X_1 = \text{number of chandeliers produced} \]
\[ X_2 = \text{number of ceiling fans produced} \]
Optimal LP Solution

$X_1 = 3.75, X_2 = 1.5, \text{ Profit } = \$35.25$

Possible Integer Solution

$6X_1 + 5X_2 \leq 30$ (assembly hours)

$2X_1 + 3X_2 \leq 12$ (wiring hours)
□ The optimal solution is $X_1 = 3.75$ chandeliers and $X_2 = 1.5$ ceiling fans.

□ Rounding to $X_1 = 4$ and $X_2 = 2$ makes the solution unfeasible.

□ Rounding to $X_1 = 4$ and $X_2 = 2$ is probably not the optimal feasible integer solution either.

□ There are 18 feasible integer solutions to this problem.

□ The optimal integer solution is $X_1 = 5$ and $X_2 = 0$, with a total profit of $35.00$.

□ The integer restriction reduced profit from $35.25$ to $35.00$.

□ An integer solution can never produce a greater profit than the LP solution to the same problem.
Listing all feasible solutions and selecting the one with the best objective function value is called the *enumeration* method. This can be virtually impossible for large problems where the number of feasible solutions is extremely large!

<table>
<thead>
<tr>
<th>Chandeliers (X1)</th>
<th>Ceiling Fans (X2)</th>
<th>Profit (Z)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>$0.00</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>14</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>21</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>28</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>35</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>20</td>
</tr>
</tbody>
</table>

Integer optimal solution
Listing all feasible solutions and selecting the one with the best objective function value is called the *enumeration* method. This can be virtually impossible for large problems where the number of feasible solutions is extremely large!

<table>
<thead>
<tr>
<th>Chandeliers (X1)</th>
<th>Ceiling Fans (X2)</th>
<th>Profit (Z)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td>27</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>34</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>19</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>26</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>33</td>
</tr>
<tr>
<td>0</td>
<td>3</td>
<td>18</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>25</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>24</td>
</tr>
</tbody>
</table>

Rounding optimal solution
Throughout the procedure, remember that the lower bound solution is determined by feasible integer solutions. Upper bound is determined by fractional LP solutions. Define two parameters LB and UB to update the lower bound and upper bound.

1. Solve the original problem using linear programming. If the answer satisfies the integer constraints, we are done. If not, this value provides an initial upper bound for the objective function.

2. Find any feasible solution that meets the integer constraints for use as a lower bound. Usually, rounding down each variable will accomplish this.
3. Branch on one variable from step 1 that does not have an integer value. Split the problem into two subproblems based on integer values that are above and below the noninteger value.

For example, if \( X_2 = 3.75 \) was in the optimal linear programming solution, introduce constraint \( X_2 \geq 4 \) in the first subproblem, and \( X_2 \leq 3 \) in the second subproblem.
4. Create nodes at the top of these new branches by solving the new problems.
5. a If a branch yields a solution that is **not feasible**, terminate the branch.

5. b If a branch yields a solution that is **feasible**, but **not an integer solution**, go to step 6.

5. c If the branch yields a feasible integer solution, look at the objective function. If its value equals the upper bound, an optimal solution has been reached.

   If it is **not equal to the upper bound**, but exceeds the lower bound, set it as the **new lower bound** and go to step 6.

   Finally, if it is less than the lower bound, terminate this branch.
6. Examine both branches again and set the **upper bound** equal to the maximum value of the objective function at all final nodes.

   If the **upper bound** equals the lower bound, stop.

   If not, go back to step 3.
Maximize profit = $7.00 X_1 + $6.00 X_2

subject to:

\[
2X_1 + 3X_2 \leq 12 \quad \text{(wiring hours)}
\]

\[
6X_1 + 5X_2 \leq 30 \quad \text{(assembly hours)}
\]

\[
X_1, X_2 \geq 0
\]

where:

\[X_1 = \text{number of chandeliers produced}\]

\[X_2 = \text{number of ceiling fans produced}\]
REVISITED
Harrison Electric Company

Optimal NON-INTEGRAL LP Solution
\( X_1 = 3.75, X_2 = 1.5, \text{Profit} = $35.25 \)

Possible Integer Solution

\[ 6X_1 + 5X_2 \leq 30 \text{ (assembly hours) } \]
\[ 2X_1 + 3X_2 \leq 12 \text{ (wiring hours) } \]
Branch-and-Bound Method

- Since $X_1$ and $X_2$ are not integers, the solution is not valid.
- The profit of $35.25$ will be the initial upper bound.
- Rounding down gives $X_1 = 3, X_2 = 1$, profit = $27.00$, which is feasible and can be used as a lower bound.

Original Non-Integer Solution

$X_1 = 3.75$
$X_2 = 1.5$
$P = 35.25$

Upper Bound = $35.25$
Lower Bound = $27.00$
(rouding down)
Branch-and-Bound Method

- We divide the problem into two subproblems, A and B
- We can branch on either the non-integer $X_1$ or $X_2$
- We choose $X_1$ this time

Original Non-Integer Solution

$X_1 = 3.75$
$X_2 = 1.5$
$P = 35.25$

Upper Bound = $35.25
Lower Bound = $27.00 (rounding down)
### Branch-and-Bound Method

<table>
<thead>
<tr>
<th>Subproblem A</th>
<th>Subproblem B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max $Z = 7X_1 + 6X_2$</td>
<td>Max $Z = 7X_1 + 6X_2$</td>
</tr>
<tr>
<td>s.t. $2X_1 + 3X_2 \leq 12$</td>
<td>s.t. $2X_1 + 3X_2 \leq 12$</td>
</tr>
<tr>
<td>$6X_1 + 5X_2 \leq 30$</td>
<td>$6X_1 + 5X_2 \leq 30$</td>
</tr>
<tr>
<td>$X_1 \geq 4$</td>
<td>$X_1 \leq 3$</td>
</tr>
</tbody>
</table>

**Original Non-Integer Solution**
- $X_1 = 3.75$
- $X_2 = 1.5$
- $P = 35.25$

**Upper Bound** = $35.25$

**Lower Bound** = $27.00$
(_rounding down)_
Branch-and-Bound Method

Subproblem A
- \( x_1 = 3.75 \)
- \( x_2 = 1.5 \)
- \( P = 35.25 \)
  - Upper Bound = $35.25
  - Lower Bound = $27.00 (rounding down)

Subproblem B
- \( x_1 = 3 \)
- \( x_2 = 2 \)
- \( P = 33.00 \)

Noninteger Solution
- Upper Bound = $35.20
- Lower Bound = $33.00

This Branch Solution Is Integer
New Lower Bound $33.00

STOP!
Subproblem A is now branched into two new subproblems, C and D.

Subproblem C has the additional constraint of $X_2 \geq 2$.

Subproblem D has the additional constraint of $X_2 \leq 1$.

The logic here is that since A’s optimal solution of $X_1 = 1.2$ is not feasible, the integer feasible answer must lie at $X_2 \geq 2$ or $X_2 \leq 1$. 
### Branch-and-Bound Method

<table>
<thead>
<tr>
<th>Subproblem C</th>
<th>Subproblem D</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Max Z = $7X_1 + $6X_2</strong></td>
<td><strong>Max Z = $7X_1 + $6X_2</strong></td>
</tr>
<tr>
<td>s.t. 2X_1 + 3X_2 &lt;= 12</td>
<td>s.t. 2X_1 + 3X_2 &lt;= 12</td>
</tr>
<tr>
<td>6X_1 + 5X_2 &lt;= 30</td>
<td>6X_1 + 5X_2 &lt;= 30</td>
</tr>
<tr>
<td>X_1 &gt;= 4</td>
<td>X_1 &gt;= 4</td>
</tr>
<tr>
<td>X_2 &gt;= 2</td>
<td>X_2 &lt;= 1</td>
</tr>
</tbody>
</table>

Subproblem C has no feasible solution whatsoever because the first two constraints are violated if X_1 >= 4 and X_2 >= 2 constraints are observed. We terminate this branch and do not consider its solution.

Subproblem D’s solution is X_1 = 4.17, X_2 = 1, profit = $35.16. This non-integer solution yields a new upper bound of $35.16.
Branch-and-Bound Method

- **Subproblem A**
  - $X_1 = 4$
  - $X_2 = 1.2$
  - $P = 35.20$

- **Subproblem B**
  - $X_1 = 3$
  - $X_2 = 2$
  - $P = 33.00$

- **Subproblem C**
  - $X_1 = 4.17$
  - $X_2 = 1$
  - $P = 35.16$
  - No Feasible Solution

- **Subproblem D**
  - Upper Bound = $35.16$
  - Lower Bound = $33.00$

- **Upper Bound** = $35.16$
- **Lower Bound** = $33.00$
Branch-and-Bound Method

Finally, we create subproblems E and F and solve for $X_1$ and $X_2$ with the additional constraints $X_1 =< 4$ and $X_1 = 5$.

**Subproblem E**

Max $Z = 7X_1 + 6X_2$

s.t.

- $2X_1 + 3X_2 =< 12$
- $6X_1 + 5X_2 =< 30$
- $X_1 = 4$
- $X_1 =< 4$
- $X_2 =< 1$

**Subproblem F**

Max $Z = 7X_1 + 6X_2$

s.t.

- $2X_1 + 3X_2 =< 12$
- $6X_1 + 5X_2 =< 30$
- $X_1 = 4$
- $X_1 = 5$
- $X_2 =< 1$
The stopping rule for the branching process is that we continue until the new upper bound is less than or equal to the lower bound or no further branching is possible. The latter is the case here since both branches yielded feasible integer solutions.
Branch & Bound (for Minimization IP)

• Branch and Bound Algorithm
  1. Solve LP relaxation to get a lower bound on cost for current branch
     • If solution exceeds upper bound, branch is terminated
     • If solution is integer, replace upper bound on cost
  2. Create two branched problems by adding constraints to original problem
     • Select integer variable with fractional LP solution
     • Add integer constraints to the original LP
  3. Repeat until no branches remain, return optimal solution.
An Example Minimization Problem
• Example: a problem with 4 variables, all required to be integer
Branch & Bound

Initial LP

$z^* = 356.1$

$x = (1.2, 2.6, 3.2, 2.8)$
Branch & Bound

Initial LP

\[ z^* = 356.1 \]
\[ x = (1.2, 2.6, 3.2, 2.8) \]

\[ x_1 \leq 1 \]

\[ x_1 \geq 2 \]
Branch & Bound

Initial LP

\[ z^* = 356.1 \]
\[ x = (1.2, 2.6, 3.2, 2.8) \]

\[ x_1 \leq 1 \]

\[ z^* = 364.1 \]
\[ x = (1, 2.8, 3.2, 2.4) \]

\[ x_1 \geq 2 \]
Branch & Bound

Initial LP

\[ z^* = 356.1 \]
\[ x = (1.2, 2.6, 3.2, 2.8) \]

\[ x_1 \leq 1 \]

\[ z^* = 364.1 \]
\[ x = (1, 2.8, 3.2, 2.4) \]

\[ x_1 \geq 2 \]

\[ z^* = \infty \]
Infeasible
Branch & Bound

Initial LP

\[
\begin{align*}
z^* &= 364.1 \\
x &= (1.2, 2.6, 3.2, 2.8)
\end{align*}
\]

\[
x_1 \leq 1
\]

\[
z^* = 364.1 \\
x = (1, 2.8, 3.2, 2.4)
\]

\[
x_1 \geq 2
\]

\[
z^* = \infty \\
\text{Infeasible}
\]
Branch & Bound

Initial LP

\[ z^* = 364.1 \]
\[ x = (1, 2.8, 3.2, 2.4) \]

\[ x \leq 1 \]
\[ x \geq 2 \]

\[ z^* = \infty \]
infeasible

\[ x \leq 2 \]
\[ x \geq 3 \]

\[ z^* = 356.1 \]
\[ x = (1.2, 2.6, 3.2, 2.8) \]
Branch & Bound

Initial LP

- \( z^* = 356.1 \)
  - \( x = (1.2, 2.6, 3.2, 2.8) \)

- \( x_1 \leq 1 \)
  - \( z^* = 364.1 \)
    - \( x = (1, 2.8, 3.2, 2.4) \)

- \( x_2 \leq 2 \)
  - \( z^* = 375.2 \)
    - \( x = (1, 2, 3.5, 3.1) \)

- \( x_1 \geq 2 \)
  - \( z^* = \infty \)
    - infeasible

- \( x_2 \geq 3 \)
  -
Branch & Bound

Initial LP

$z^* = 356.1$
$x = (1.2, 2.6, 3.2, 2.8)$

$x_1 \leq 1$

$z^* = 364.1$
$x = (1, 2.8, 3.2, 2.4)$

$x_1 \geq 2$

$z^* = \infty$
infeasible

$x_2 \leq 2$

$z^* = 375.2$
$x = (1, 2, 3.5, 3.1)$

$x_2 \geq 3$

$z^* = 384.1$
$x = (1, 3, 4.1, 2.2)$
Branch & Bound

Initial LP

\[ z^* = 356.1 \]
\[ x = (1.2, 2.6, 3.2, 2.8) \]

\[ x_1 \leq 1 \]

\[ z^* = 364.1 \]
\[ x = (1, 2.8, 3.2, 2.4) \]

\[ x_1 \geq 2 \]

\[ z^* = \infty \]
\[ \text{infeasible} \]

\[ x_2 \leq 2 \]

\[ z^* = 375.2 \]
\[ x = (1, 2, 3.5, 3.1) \]

\[ x_2 \geq 3 \]

\[ z^* = 384.1 \]
\[ x = (1, 3, 4.1, 2.2) \]

\[ x_3 \leq 3 \]

\[ x_3 \geq 4 \]
Branch & Bound

Initial LP

\[ z^* = 356.1 \]
\[ x = (1.2, 2.6, 3.2, 2.8) \]

\[ x_1 \leq 1 \]
\[ x_1 \geq 2 \]

\[ z^* = 364.1 \]
\[ x = (1, 2.8, 3.2, 2.4) \]

\[ x_2 \leq 2 \]
\[ x_2 \geq 3 \]

\[ z^* = \infty \]
\[ \text{infeasible} \]

\[ z^* = 375.2 \]
\[ x = (1, 2.3, 5.3, 1.1) \]

\[ x_3 \leq 3 \]
\[ x_3 \geq 4 \]

\[ z^* = 380 \]
\[ x = (1, 2.3, 4.3) \]

\[ x_3 \leq 3 \]
\[ x_3 \geq 4 \]

\[ z^* = 384.1 \]
\[ x = (1, 3, 4.1, 2.2) \]

Philip Kilby, Australian National University, 2008
Branch & Bound

Initial LP
\[ z^* = 356.1 \]
\[ x = (1.2, 2.6, 3.2, 2.8) \]

\[ x_1 \leq 1 \]
\[ z^* = 364.1 \]
\[ x = (1, 2.8, 3.2, 2.4) \]

\[ x_1 \geq 2 \]
\[ z^* = \infty \]
infeasible

\[ x_2 \leq 2 \]
\[ z^* = 375.2 \]
\[ x = (1, 2, 3.5, 3.1) \]

\[ x_2 \geq 3 \]
\[ z^* = 384.1 \]
\[ x = (1, 3, 4.1, 2.2) \]

\[ x_3 \leq 3 \]
\[ z^* = 380 \]
\[ x = (1, 2, 3, 4) \]

\[ x_3 \geq 4 \]
Branch & Bound

Initial LP

- $z^* = 356.1$
  - $x = (1.2, 2.6, 3.2, 2.8)$
- $x_1 \leq 1$
- $x_1 \geq 2$

- $z^* = 364.1$
  - $x = (1, 2.8, 3.2, 2.4)$
  - $z^* = \infty$
  - infeasible

- $x_2 \leq 2$
- $x_2 \geq 3$

- $z^* = 375.2$
  - $x = (1, 2, 3.5, 3.1)$

- $x_3 \leq 3$
- $x_3 \geq 4$

- $z^* = 380$
  - $x = (1, 2, 3, 4)$
- $z^* = 378.1$
  - $x = (1, 2, 4, 1.2)$
- $z^* = 384.1$
  - $x = (1, 3, 4.1, 2.2)$
Branch & Bound

Initial LP

\[ z^* = 364.1 \]
\[ x = (1, 2.8, 3.2, 2.4) \]

- \( x_1 \leq 1 \) : \[ z^* = 364.1 \]
  \[ x = (1, 2.8, 3.2, 2.4) \]
- \( x_1 \geq 2 \) : \[ z^* = \infty \] infeasible

- \( x_2 \leq 2 \) : \[ z^* = 375.2 \]
  \[ x = (1, 2.8, 3.5, 3.1) \]
- \( x_2 \geq 3 \) : \[ z^* = 384.1 \]
  \[ x = (1, 3.4, 1.2, 2.2) \]

- \( x_3 \leq 3 \) : \[ z^* = 380 \]
  \[ x = (1, 2.3, 4) \]
- \( x_3 \geq 4 \) : \[ z^* = 378.1 \]
  \[ x = (1, 2, 4, 1.2) \]

Philip Kilby, Australian National University, 2008
Branch & Bound

Initial LP

$z^* = 356.1$
$x = (1.2, 2.6, 3.2, 2.8)$

$x_1 \leq 1$

$z^* = 364.1$
$x = (1, 2.8, 3.2, 2.4)$

$x_1 \geq 2$

$z^* = \infty$
infeasible

$x_2 \leq 2$

$z^* = 375.2$
$x = (1, 2, 3.5, 3.1)$

$x_2 \geq 3$

$z^* = 384.1$
$x = (1, 3, 4.1, 2.2)$

$x_3 \leq 3$

$z^* = 380$
$x = (1, 2, 3, 4)$

$x_3 \geq 4$

$z^* = 378.1$
$x = (1, 2, 4, 1.2)$

$x_4 \leq 1$

$x_4 \geq 2$

Philip Kilby, Australian National University, 2008
Branch & Bound

Initial LP

- $z^* = 356.1$
- $x = (1.2, 2.6, 3.2, 2.8)$
- $x_1 \leq 1$
- $x_1 \geq 2$

- $z^* = 364.1$
- $x = (1, 2.8, 3.2, 2.4)$
- $x_2 \leq 2$
- $x_2 \geq 3$

- $z^* = 375.2$
- $x = (1, 2, 3.5, 3.1)$
- $x_3 \leq 3$
- $x_3 \geq 4$

- $z^* = 380$
- $x = (1, 2, 3, 4)$
- $x_4 \leq 1$
- $x_4 \geq 2$

- $z^* = 381$
- $x = (1, 2, 4, 0)$
- $z^* = 384.1$
- $x = (1, 3, 4.1, 2.2)$

- $z^* = \infty$
- infeasible

Philip Kilby, Australian National University, 2008
Branch & Bound

Initial LP

- \( z^* = 356.1 \)
- \( x = (1.2, 2.6, 3.2, 2.8) \)

\( x_1 \leq 1 \)

- \( z^* = 364.1 \)
- \( x = (1, 2.8, 3.2, 2.4) \)

- \( z^* = \infty \)
- infeasible

\( x_1 \geq 2 \)

\( x_2 \leq 2 \)

- \( z^* = 375.2 \)
- \( x = (1, 2, 3.5, 3.1) \)

\( x_2 \geq 3 \)

\( x_3 \leq 3 \)

- \( z^* = 380 \)
- \( x = (1, 2, 3, 4) \)

\( x_3 \geq 4 \)

\( x_4 \leq 1 \)

- \( z^* = 381 \)
- \( x = (1, 2, 4, 0) \)

\( x_4 \geq 2 \)
Branch & Bound

Initial LP

\[ z^* = 356.1 \]
\[ x = (1.2, 2.6, 3.2, 2.8) \]

- \( x_1 \leq 1 \)
- \( x_1 \geq 2 \)

- \( z^* = 364.1 \)
\[ x = (1, 2.8, 3.2, 2.4) \]

- \( x_2 \leq 2 \)
- \( x_2 \geq 3 \)

- \( z^* = \infty \)
infeasible

- \( z^* = 375.2 \)
\[ x = (1, 2, 3.5, 3.1) \]

- \( x_3 \leq 3 \)
- \( x_3 \geq 4 \)

- \( z^* = 380 \)
\[ x = (1, 2, 3, 4) \]

- \( z^* = 378.1 \)
\[ x = (1, 2, 4, 1.2) \]

- \( x_4 \leq 1 \)
- \( x_4 \geq 2 \)

- \( z^* = 381 \)
\[ x = (1, 2, 4, 0) \]

- \( z^* = 382.1 \)
\[ x = (1, 2, 4, 3.3) \]
Branch & Bound

Initial LP

\[ z^* = 364.1 \]
\[ x = (1, 2.8, 3.2, 2.4) \]

\[ x_1 \leq 1 \]

\[ z^* = 375.2 \]
\[ x = (1, 2.3.5, 3.1) \]

\[ x_2 \leq 2 \]

\[ z^* = 356.1 \]
\[ x = (1.2, 2.6, 3.2, 2.8) \]

\[ x_1 \geq 2 \]

\[ z^* = \infty \]
\[ \text{infeasible} \]

\[ x_2 \geq 3 \]

\[ z^* = 384.1 \]
\[ x = (1, 3, 4.1, 2.2) \]

\[ x_3 \leq 3 \]

\[ z^* = 380 \]
\[ x = (1, 2, 3, 4) \]

\[ x_3 \geq 4 \]

\[ z^* = 378.1 \]
\[ x = (1, 2, 4, 1.2) \]

\[ x_4 \leq 1 \]

\[ z^* = 381 \]
\[ x = (1, 2, 4, 0) \]

\[ x_4 \geq 2 \]

\[ z^* = 382.1 \]
\[ x = (1, 2, 4, 3.3) \]
Key Rules of B&B for Minimization IPs

• Each integer feasible solution is an upper bound on solution cost,
  ➢ Branching stops
  ➢ It can prune other branches
  ➢ Anytime result: can provide optimality bound

• Each LP-feasible solution is a lower bound on the solution cost
  ➢ Branching may stop if LB \geq UB
References

• **Book:**
  – *Integer Programming*, Michele Conforti, Gérard Cornuéjols, Giacomo Zambelli

• **Slides:**
  – Linear Programming, (Mixed) Integer Linear Programming, and Branch & Bound, Philip Kilby, Australian National University