Branch and Bound Method

The Branch and Bound (B&B) is a strategy to explore the solution space based on the implicit enumeration of the solutions:

- B&B examines disjoint subsets of solutions (branching)
- Evaluates such subsets on the basis of an estimation of the objective function (bounding) and eliminates the subsets that cannot contain the optimal solution
- The exploration is performed by solving a sequence of relaxed problem (RL) associated with the disjoint subsets of solutions

The idea is to subdivide the whole problem into a set of subproblems of progressively decreasing dimensions that can be more easily solved (*Divide and Conquer*)
Branch and Bound Method

- Different from Cutting Planes: B&B solves not a sequence of increasingly constrained problems but a set of problems that do not share solutions
- The problems are generated with a recursive hierarchical relation:
  - A parent problem is divided into two (or more) disjoint child problems
  - The solution for the parent problem corresponds to the optimal solution for one of the child problems
- Dividing (separating) problems ⇒ branching phase
- B&B exploration ⇒ represented by a tree graph
- Enumeration Tree:
  - Nodes = subproblems
  - Branches (links) = problem subdivisions – hierarchical relations
Branch and Bound Method

Example

Tree root: RL of original IP problem

Root descendants: two RLs obtained dividing RL₀ into two disjoint problems

RL₀ has non integer solution

RL₁ is the parent of RL₃ and RL₄ (successors nodes)

Leaf: node without descendants

Branch and Bound Method

RL₁ is the parent of RL₃ and RL₄ (successors nodes)
Branch and Bound Method

- Pure branching ends when added constraints fix all variables
- Binary tree as elimination tree for 0-1 LP problem:
  - branching = fixing the value of a variable to 0 or 1

Example with 3 binary variables

Tree with 3+1=4 levels
- $2^3=8$ leaves
- $2^3-1=7$ intermediate nodes

A tree generated by $n$ binary variables:
- $n+1$ levels
- $2^n$ leaves
- $2^n-1$ intermediate nodes
- $2^{n+1}-1$ nodes
Branch and Bound Method

- Pure branching ⇒ not an effective algorithm (total number of nodes greater than the number of solutions)
- Effectiveness relies on *Bounding*:
  - A bound estimates the best objective value for all subproblems generated from a parent node
  - Bounding ⇒ exploit information on bounds to avoid a complete tree exploration
  - Bounding allows eliminating (*pruning*) parts of the tree without exploring them ⇒ B&B is an implicit enumeration method
  - The B&B approach is a general purpose strategy that can be applied any combinatorial optimization problem
Branch and Bound Method

Consider the IP problem \((IP)\) \[ \max x_0 = c^T x \]
\[ Ax = b \]
\[ x \in \mathbb{Z}^n_+ \]

First B&B steps:
- Solve RL of (IP) associating the solution to root node (RL\(_0\))
  \[ \Rightarrow Z_0 \] objective value and \( x^0 \) optimal solution to (RL\(_0\))
  Assume \( x^0 \) not integer (at least a component is fractional)
- Choose a not integer solution component \( x_j^0 \)
- Partition the feasibility region of (RL)
  \[ P^0 = \{ x \in \mathbb{R}^n : Ax = b, x \geq 0 \} \]
  into two disjoint regions adding to \( P^0 \) one of the following constraints
  \[ x_j \leq \lfloor x_j^0 \rfloor \quad \text{or} \quad x_j \geq \lfloor x_j^0 \rfloor + 1 \]
Branch and Bound Method

The elimination tree: two disjoint relaxed problems are created

\[ Z_0 \quad x^0 \text{ not integer} \]

\[ x_j \leq \lfloor x_j^0 \rfloor \quad x_j \geq \lceil x_j^0 \rceil + 1 \]

Next steps: iterate for the child nodes RL_1 and RL_2

- solve the relaxed problem (e.g., RL_1)
- If a not integer solution is found then branch the tree by selecting a fractional component of the solution (branching)
Branch and Bound Method

The elimination tree

Active nodes: generated nodes that are not solved or solved but with not integer solution

Integer nodes: not explored anymore (node fathomed for integrality)

Unfeasible nodes: not explored anymore (node fathomed for unfeasibility)
Branch and Bound Method

The elimination tree: fathomed nodes are closed (not active)

Integer solution $\bar{x}^3$ is the first new current solution (incumbent solution)
Objective $Z_3$ is a Lower Bound (LB): since the problem is a maximization, we never accept an inferior solution
Branch and Bound Method

Bounding

• LB is fundamental for bounding
• Objective values associated with solved active nodes are *Upper Bound* (UB)
• In general, given:
  – \((RL_i)\) associated with node \(i\) \((A^i x = b^i\) set of original constraints plus constraints added added for branching)
    \[
    (RL_i) \quad \max x_0 = c^T x \\
    A^i x = b^i \\
    x \in \mathbb{R}^n_+
    \]
  – \(x^{lb}\) the best integer solution found so far
  – \(x^{ub}\) the not integer solution of \((RL_i)\)
Branch and Bound Method

If $c^T x^{ub} \leq c^T x^{lb}$ since $c^T x^* \leq c^T x^{ub}$
where $x^*$ is solution to the integer problem for node $i$

$$(IP_i) \max x_0 = c^T x$$

$$A^i x = b^i$$

$$x \in \mathbb{Z}^n_+$$

then the best integer solution to (IP$_i$) cannot never be better than current incumbent solution $x^{lb}$

$$c^T x^* \leq c^T x^{ub} \leq c^T x^{lb}$$

Denoting with $Z_{LB}$ the current LB, if for active node $i$ it holds $Z_i \leq Z_{LB}$
then node $i$ is **fathomed for bounding** (and closed)
Branch and Bound Method

- Whenever an integer solution $x^h$ for node $h$ is found
  
  If $Z_h > Z_{LB}$
  
  - $x^h$ becomes the new incumbent solution
  - LB is updated $Z_{LB} = Z_h$
  - Each active node UB is compared with the new LB to verify if any node can be fathomed for bounding

- Whenever a non-integer solution is found for a node, UB for the node is compared to the current LB to verify if the node can be fathomed per bounding
Branch and Bound Method

The elimination tree (example)

No active nodes: B&B terminates
The current incumbent solution is optimal
Branch and Bound Method

• If B&B terminates without founding any feasible solution (IP) is not feasible
• In general B&B tree is not completely explored since branching is stopped when:
  – (RL) for a node is unfeasible
  – the solution to a (RL) for a node is integer
  – the solution to a (RL) for a node is not integer but bounding guarantees that branching from that node cannot lead to the optimal solution
• NB: in case of minimization the terms UB and LB must be exchanged and the condition for fathoming for bounding must be reversed
Branch and Bound Algorithm

Algorithm steps:

1. **Initialization**
   - Let (RL₀) the root active node and \( P₀ \) the polyhedron associated with RL
   - Initialize \( Z_{LB} = -\infty \) as current LB and \( Z₀ = \infty \) as objective value (UB) for node (RL₀)

2. **Branching**
   - If no active node exists the go to step 7 otherwise select an active node \( j \)
   - If (RL\( j \)) has already been solved go to step 3 otherwise go to step 4

3. **Separation**
   - Select a basic fractional variable \( x_{B_i} = y^j_{i₀} \) and partition \( P_j \) as
     \[
     P_j \cap \{ x : x_{B_i} \leq \lfloor y^j_{i₀} \rfloor \} \quad P_j \cap \{ x : x_{B_i} \geq \lceil y^j_{i₀} \rceil + 1 \}
     \]
   - generating two new nodes with the same UB of node \( j \)
   - Go to step 2
Branch and Bound Algorithm

Algorithm steps:

4. Solution of \((RL_j)\)
   - Solve problem \((RL_j)\)
   - If no feasible solution exists the node is fathomed for unfeasibility (no more active); go to step 2
   - If an optimal solution \(x^j\) exists set \(Z_j = x_0^j\) and go to step 5

5. Fathoming for integrality
   - If \(x^j\) is not integer go to step 6 otherwise prune the node \(j\) and set \(Z_{LB} = \max\{Z_{LB}, Z_j\}\)
   - If LB is updated then the new incumbent solution is the one associated with node \(j\)
   - Go to step 6
Branch and Bound Algorithm

Algorithm steps:

6. Fathoming for bounding
   • Prune any active node $k$ such that $Z_k \leq Z_{LB}$
   • Go to step 2

7. Termination
   • The algorithm stops
   • If $Z_{LB} = -\infty$ then (IP) is not feasible
   • If $Z_{LB} > -\infty$ then the incumbent solution is optimal and the associated optimal objective value is $Z_{LB}$
Branch and Bound Algorithm

Remarks:

• B&B convergence in a finite number of steps is guaranteed if the problem has optimal finite solutions
• The algorithm performance depends on the different branching strategies adopted at step 2
• Two possible opposite branching strategies:
  – Depth First
  – Breadth First
Branch and Bound Algorithm

• Depth First (exploration in depth):
  – Whenever the current node is not fathomed, generate the two child nodes and continue by exploring one of them at the next level
  – Purpose: fix many variables to quickly reach an integer solution as LB
  – Reduced number of active nodes
Branch and Bound Algorithm

• Breadth First (exploration in breadth):
  – All nodes at a given level are explored before moving to the next level
  – Purpose: compute UB for many nodes hoping to prune many branches by bounding as soon as an integer solution is found
  – Memory usage is larger than depth first strategy
Branch and Bound Algorithm

- The two strategies are extreme cases: composed strategies may exist (explore in breadth only a subset of nodes)
- Selection strategies may be used at step 3
- The given examples use binary trees; more child nodes at a same level can be generated in case of integers (e.g.: 3 nodes $x_i \leq 3$, $4 \leq x_i \leq 5$, $x_i \geq 6$)
- Whenever breadth first is not used, another decision influencing B&B is the choice of the node to explore next
  - A strategy can be choose the node with the greatest UB (the most promising)
- Bounding is facilitated by the availability of good quality solutions: heuristic algorithms can be used to complete the partial solutions associated with B&B intermediate nodes
Branch and Bound: an example

Consider the IP problem

max \(- x_1 + 3 x_2\)
\[ x_1 - 3/5 x_2 \geq 0 \] (1)
\[ 3/2 x_1 + 2 x_2 \leq 8 \] (2)
\[ x_1 + 3/10 x_2 \leq 7/10 \] (3)
\[ x_1, x_2 \geq 0 \]

At node RL_0 the linear relaxation is solved

\( Z_0 = 6.62 \)
\( x_1 = 1.65 \quad x_2 = 2.75 \)
\( \text{Not integer solution} \)
Branch and Bound: an example

Choose $x_1$ for branching

$Z_0 = 6.62$

$x_1 \leq 1$

$x_1 \geq 2$

Branch and Bound: an example
Branch and Bound: an example

Solving RL₁

\[ x₁ \leq 1 \]

\[ Z₀ = 6.62 \]

\[ x₁ \geq 2 \]

\[ Z₁ = 4 \]

\[ \text{max} - x₁ + 3x₂ \]

\[ x₁ - \frac{3}{5} x₂ \geq 0 \]  \hspace{1cm} (1)

\[ \frac{3}{2} x₁ + 2x₂ \leq 8 \]  \hspace{1cm} (2)

\[ x₁ + \frac{3}{10} x₂ \leq \frac{7}{10} \]  \hspace{1cm} (3)

\[ x₁ \leq 1 \]  \hspace{1cm} (4)

\[ x₁, x₂ \geq 0 \]

Not integer solution
Branch and Bound: an example

Solving RL₂

\[ Z_0 = 6.62 \]
\[ x_1 \leq 1 \]
\[ x_1 \geq 2 \]
\[ Z_1 = 4 \]
\[ Z_2 = 5.5 \]

\[ Z_1 = 5.5 \]
\[ x_1 = 2 \]
\[ x_2 = 2.5 \]

Not integer solution

\[ \text{max} - x_1 + 3x_2 \]
\[ x_1 - 3/5x_2 \geq 0 \quad (1) \]
\[ 3/2x_1 + 2x_2 \leq 8 \quad (2) \]
\[ x_1 + 3/10x_2 \leq 7/10 \quad (3) \]
\[ x_1 \geq 2 \quad (4) \]
\[ x_1, x_2 \geq 0 \]
Branch and Bound: an example

Branching for RL₂ (the largest UB) separating with \( x_2 \)

\[
\begin{align*}
Z_0 &= 6.62 \\
Z_1 &= 4 \\
Z_2 &= 5.5
\end{align*}
\]

Not feasible solution
Branch and Bound: an example

Solving RL$_3$

max $- x_1 + 3x_2$

1. $x_1 - 3/5 x_2 \geq 0$
2. $3/2 x_1 + 2x_2 \leq 8$
3. $x_1 + 3/10 x_2 \leq 7/10$
4. $x_1 \geq 2$
5. $x_2 \leq 2$
6. $x_1, x_2 \geq 0$

Integer solution:

$Z_3 = 4$

$x_1 = 2$

$x_2 = 2$
Branch and Bound: an example

RL₁ is fathomed for bounding

\[ RL₀ \]

\[ Z₀ = 6.62 \]

\[ x₁ ≤ 1 \]

\[ Z₁ = 4 \]

RL₁ fathomed for bounding

\[ (UB) \ Z₁ ≤ Z₃ \ (LB) \]

\[ RL₂ \]

\[ Z₂ = 5.5 \]

\[ x₁ ≥ 2 \]

\[ x₂ ≤ 2 \]

\[ RL₃ \]

\[ Z₃ = 4 \]

\[ RL₄ \]

\[ x₂ ≥ 3 \]

Branch and Bound: an example
Branch and Bound: an example

Solving RL₄

\[ \text{max} - x₁ + 3x₂ \]
\[ x₁ - \frac{3}{5} x₂ \geq 0 \quad (1) \]
\[ \frac{3}{2} x₁ + 2x₂ \leq 8 \quad (2) \]
\[ x₁ + \frac{3}{10} x₂ \leq \frac{7}{10} \quad (3) \]
\[ x₁ \geq 2 \quad (4) \]
\[ x₂ \geq 3 \quad (5) \]
\[ x₁, x₂ \geq 0 \]

Not feasible solution

No more active nodes: the incumbent solution associated with RL₃ is optimal