Branch and Bound

(Wolsey: Chapter 7 (7.1, 7.2))

Consider

\[
(p) \quad z = \max_{x \in S} c(x)
\]

How can we break \((p)\) into a series of smaller (and easier) problems, solve the smaller problems, and then put the information together to solve \((p)\) ?

**Divide and conquer approach**

**Proposition:** Let \( S = S_1 \cup S_2 \cup \ldots \cup S_K \) be a decomposition of \( S \), and let

\[
z^K = \max_{x \in S_k} c(x) \quad , \quad k = 1, \ldots, K.
\]

Then \( z = \max_K z^K \).
A typical way is to decompose $S$ via an enumeration tree.

However: a complete enumeration is usually impossible (we cannot divide indefinitely).

So:

- How can we use some bounds on $x y$ intelligently?

- How can we put together bound information?

Example: binary enumeration tree for $S \leq 10, 1^3$
Implicit enumeration

Proposition: Let \( S = S_1 \cup \ldots \cup S_k \) be a decomposition of \( S \), and let \( z^k = \max_{x \in S_k} c(x) \); \( \bar{z}^k \) be an upper bound on \( z^k \), and \( \underline{z}^k \) be a lower bound on \( z^k \), \( k = 1, \ldots, K \).

Then:

\[
\bar{z} = \max_k \bar{z}^k
\]

is an upper bound on \( z \)

and

\[
\underline{z} = \max_k \underline{z}^k
\]

is a lower bound on \( z \)

- So, bound information (partial information) about subproblems can be put together to derive bounds on \( z \)!

- What can be deduced from these bounds, and which sets need further examination to compute \( z \)?
Examples

1. \[ \max \sum \begin{array}{c} 24 \text{ (u. b.)} \\ 13 \text{ (l. b.)} \end{array} \]

Based on the previous proposition:

\[ \bar{z} = \max_k z^k = \max \{ 20, 25 \} = 25 \]

\[ \underline{z} = \max_k z^k = \max \{ 20, 15 \} = 20 \]

Further observe that, since the upper and lower bounds on \( z_1 \) are equal, then \( z_1 = 20 \), and so there is no further reason to examine \( S_1 \); the branch \( S_1 \) of the enumeration tree can be pruned by optimality.
Further observe that, since $\bar{z} > 21$, and the upper bound $\bar{z}_1 = 20$, there is no optimal solution of (P) can belong to $S_1$; the branch $S_1$ of the enumeration tree can be pruned by bound.
\[ \begin{align*}
\bar{z} &= \max \{ 24, 37 \} = 37 \\
\Xi &= \max \{ 13, -\infty \} = 13
\end{align*} \]

Here no further conclusion can be drawn: we need to explore both \( S_1 \) and \( S_2 \).
So, we can list at least 3 reasons that allow us to "prune" the tree, and thus enumerate many solutions implicitly:

1) **Pruning by optimality:**

\[ z_\ell = \max_{x \in S_\ell} c(x) \quad \text{for some } \ell \]

has been solved;

2) **Pruning by bound:**

\[ \bar{z}_\ell \leq z \quad \text{for some } \ell \]

3) **Pruning by infeasibility:**

\[ S_\ell = \emptyset \quad \text{for some } \ell \]

Based on the above ideas, we can design an implicit enumeration framework, or Branch and Bound.
Branch and Bound framework

Procedure B & B \((P, z)\)

begin

\( Q_1 := \frac{1}{2}(P), z := -\infty; \)

* \( Q \) is the set of active (\( \neq \) no pruned) nodes of the tree *

repeat

\( (P') := \text{Next}(Q); Q_0 := Q \setminus \frac{1}{2}(P') \); \( z := \text{RELAX}(P') \);

if \( z > z \) then

begin

\( \hat{z} := \text{HEURISTIC}(P') \);

if \( \hat{z} > z \) then \( z := \hat{z} \);

if \( z > z \) then

\( Q_1 := Q \cup \text{BRANCH}(P') \)

end

until \( Q = \emptyset \)
Questions to be addressed to have a well-defined Branch and Bound algorithm

1. **RELAX**: upper (dual) bound
   < via a relaxation approach >
   HEURISTIC: lower (primal) bound
   < via a heuristic approach >
   - strong (upper) bounds
   - efficient procedures (quite rapid)
   - RELAX and HEURISTIC possibly interconnected
   - reoptimization techniques if possible

2. **NEXT**: order to examine subproblems
   (= visiting strategies of the enumeration tree)
   - topological visits: breadth-first (Q is a queue)
   - depth-first (Q is a stack)
information based visits:
e.g. *best-first* ("promising" mode)

3) **BRANCH**: how should S be decomposed?

- completeness is necessary:
  \[ S = S_1 \cup S_2 \cup \ldots \cup S_k \]

- additional (not necessary) properties:
  - partitioning:
    \[ S_i \cap S_j = \emptyset \quad \forall i \neq j \]
    * a unique path in the enumeration tree for each (examined) solution *
  - balancing:
    \[ |S_i| \leq |S_j| \quad \forall i \neq j \]
    * balanced enumeration tree *
    * relevant if a depth-first visit is adopted *
- compatibility with RELAX and HEURISTIC

example: Knapsack

\[ x_i = 0 \quad \text{or} \quad x_i = 1 \]

the two subproblems are still (reduced) Knapsack problems: we can use the same bounding procedure at each node!

few rows per node: too many rows may require a large amount of memory

Pre-processing: to accelerate the execution

example: Knapsack

1. If \( a_i > b \) we can fix \( x_i = 0 \)
2. If \( c_i < 0 \) and \( a_i > 0 \) \( \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad x_i = 0 \)
Examples

(1) Binary Knapsack

\[ \begin{align*}
(P) & \quad \text{max} \quad 8x_1 + 5x_2 + 5x_3 + 3x_4 + x_5 \\
& \quad 4x_1 + 3x_2 + 4x_3 + 3x_4 + 2x_5 \leq 12 \\
& \quad x_i \in \{0, 1\}, \quad i = 1, \ldots, 5
\end{align*} \]

Relax: Linear relaxation

Heuristic: Greedy heuristic (CUD)

Branch: on the only fractional variable of the LP optimal solution

Next: breadth-first, by giving priority to the subproblem obtained by fixing the fractional variable to 1

Notation (at each node):

- \( x^* \): optimal solution of the LP
- \( \bar{x} \): solution of greedy heuristic
- \( \bar{\overline{z}} \): upper bound \((\bar{\overline{z}} = c \cdot \bar{x})\)
- \( \overline{z} \): lower bound \((\overline{z} = c \cdot \bar{x})\)
- \( z \): current best solution value
Note that
\[
\frac{c_1}{a_1} \geq \frac{c_2}{a_2} \geq \frac{c_3}{a_3} \geq \frac{c_4}{a_4} \geq \frac{c_5}{a_5}
\]

Initialization: \( q_1 = \frac{1}{2}(p) \gamma; \ z_1 = -\infty \)

Radix mode: \( p \)

\[
x^* = (1, 1, 1, \frac{1}{3}, 1, 0)
\]
\[
\bar{z} = 19
\]
\[
\overline{x} = (1, 1, 1, 0, 1, 0)
\]
\[
z = 18
\]

Since \( z > \bar{z} (= -\infty) \) then \( z = 18 \)

Since \( \bar{z} > z \), then BRANCH on \( x_4 \)

\[
x_4 = 1
\]

\[
x^* = (1, 1, 1, \frac{1}{2}, 1, 0)
\]
\[
\bar{z} = 18 + \frac{1}{2}
\]
\[
\overline{x} = (1, 1, 0, 1, 1, 1)
\]
\[
z = 17
\]

Since \( \bar{z} > z \), then BRANCH on \( x_3 \)

\[
x_4 = 0
\]

\[
x^* = (1, 1, 1, 0, \frac{1}{2})
\]
\[
\bar{z} = 18 + \frac{1}{2}
\]
\[
\overline{x} = (1, 1, 1, 0, 0)
\]
\[
z = 18
\]

Since \( \bar{z} > z \), then BRANCH on \( x_5 \)
\[ x_4 = 1, x_3 = 1 \] \quad P_3

\[ x^* = (1, \frac{1}{3}, 1, 1, 0) \quad \bar{z} = 17 + \frac{2}{3} \]

Since \( \bar{z} \leq z = 18 \), then **pruning by bound**.

\[ x_4 = 1, x_3 = 0 \] \quad P_4

\[ x^* = (1, 1, 0, 1, 1) \quad \bar{z} = 17 \]

**pruning by optimality** (and also by bound).

\[ x_4 = 0, x_5 = 1 \] \quad P_5

\[ x^* = (1, 1, \frac{3}{4}, 0, 1) \quad \bar{z} = 17 + \frac{3}{4} \]

Since \( \bar{z} \leq z = 18 \), then **pruning by bound**.

\[ x_4 = 0, x_5 = 0 \] \quad P_6

\[ x^* = (1, 1, 1, 0, 0) \quad \bar{z} = 18 \]

**pruning by optimality** (also by bound).

\[ Q = \emptyset \quad \textbf{STOP} \]

\[ x = (1, 1, 1, 0, 0) \], corresponding to \( z = 18 \), is an optimal solution.
Properties:

- BRANCH: partitioning rule
- Connection between RELAX and HEURISTIC (both based on the CUD order)
- BRANCH is compatible with RELAX and HEURISTIC: each subproblem is a (reduced) Knapsack problem
- The upper bound may strictly decrease along each path (the optimal LP solution is cutoff from the two subproblems)

Note that:

$$\max \left\{ \hat{z}(P_1), \hat{z}(P_2) \right\} \leq z(P) \leq \max \left\{ z(P_1), z(P_2) \right\}$$

\[ \begin{align*}
\text{decreases} & \quad \text{increases}
\end{align*} \]
The corresponding enumeration tree

\[
\begin{align*}
& x_4 = 1 \\
& x_3 = 1 \\
& x_5 = 1 \\
& x_4 = 0 \\
& \bar{z} = 19 \\
& \bar{z} = 18 \\
& z = -\infty \\
& z = 18 \\
& \bar{z} = 18 + \frac{1}{2} \\
& \bar{z} = 18 \\
& z = 18 \\
& \bar{z} = 17 + \frac{2}{3} \\
& \bar{z} = 18 \\
& z = 17 \\
& \bar{z} = z = 17 \\
& \bar{z} = 17 + \frac{3}{4} \\
& z = z = 18 \\
& \text{pruning by optimality} \\
& \text{pruning by optimality} \\
\end{align*}
\]

Would it be possible to prune $P_1$ and $P_2$, so generating only two levels of the tree?
2) Constrained shortest path

![Diagram]

RELAX: shortest path computation
HEURISTIC: 
BRANCH: partitioning based on the shortest path
NEXT: breadth-first

**Initialization**: \( Q = \gamma(P) \gamma \); \( z = +\infty \)

Observation: minimization problem
shortest path from \( e \) to \( t \)

\[
\begin{align*}
1 & \rightarrow 2 \rightarrow 3 \\
& \quad \quad \downarrow \\
& \quad \quad \quad \downarrow \\
& \quad \quad \quad \quad \downarrow \\
& \quad \quad \quad \quad \quad \downarrow \\
& \quad \quad \quad \quad \quad \quad \downarrow \\
& \quad \quad \quad \quad \quad \quad \quad \downarrow \\
\end{align*}
\]

\( z = 3 \) lower bound

\( \bar{z} = +\infty \) no heuristic

\( \bar{z} < \bar{z} \) then BRANCH

Since \( \sum_{(i,j) \in \Pre} x_{ij} = 3 \geq 2 \), \( \text{Pre} \) is not feasible, so define the following branching rule:

\[
\begin{align*}
\overline{z} = +\infty \\
\bar{z} = 3 \\
\end{align*}
\]

Current enumeration tree

\[
\begin{align*}
x_{12} = 0 \\
x_{12} = 1 & \quad x_{23} = 0 \\
x_{12} = 1 & \quad x_{23} = 1 \\
x_{34} = 0
\end{align*}
\]
$P_1$ is a constrained shortest path in a reduced graph (delete $(1,2)$ from the graph)

shortest path from $x$ to $t$

$\begin{array}{c}
\text{3} \\
\text{1} \\
\text{0} \\
\text{0}
\end{array}$

$x \rightarrow 1 \rightarrow 3 \rightarrow 1 \rightarrow t$

$z = 4$

Since $z \leq e_{i\delta} = 2 = z$, then $P_{x,t}$ is a feasible path:

$z = 4$

Since $z < z (= +\infty)$, then $z = 4$

Since $z = 2$, then framing by optimality
$P_2$ : delete $(2,3)$ and "imposes" $(1,2)$:

Equivalently: delete $(2,3)$, $(1,2)$ and the arcs incident node 2, and look for a shortest path from 2 to 4:

\[ z = 1 + 4 = 5 \]

Since $z \leq 5$, then $P_{re}$ is a feasible path:

\[ z = 4 < 5 \] (no improvement of $z$)

Since $\frac{z}{5} > \frac{z}{4}$ (also by optimality)
"imposes" the subpath \((1, 2, 3)\) and delete \((3, 4)\):

Equivalently: look for a shortest path from 3 to 4 by removing nodes 1 and 2 and arc \((3, 4)\): no path

\[
Q = \emptyset; \quad z = 4
\]

constrained shortest path
Properties:
- **BRANCH**: partitioning rule
- **BRANCH is compatible with RELAX**: each subproblem is a (potentially) constrained shortest path
- The lower bound may strictly increase along each path of the tree

The corresponding enumeration tree:

- \( P \):
  - \( x_{12} = 0 \):\( \bar{z} = z = 4 \) pruned by optimality
  - \( x_{12} = 1 \):\( x_{23} = 0 \):
    - \( z = 3 \)
    - \( x_{12} = 1 \):
      - \( x_{23} = 1 \):
        - \( x_{24} = 0 \):
          - \( z = +\infty \) pruned by infeasibility
          - \( \bar{z} = 5 \) pruned by bound (and optimality)
      - \( \bar{z} = +\infty \) pruned by bound

TSP

**RELAX:**

ILP formulation on $G = (V, A)$:

$$\text{Min } \sum_{(i, j) \in A} c_{ij} x_{ij}$$

$$\sum_{(i, j) \in A} x_{ij} = 2 \quad \forall i \in V$$

*connection & <no subtour constraints>*

$\forall (i, j) \in A$

$x_{ij} \in \{0, 1\}$

*where:*

$x_{ij} = \begin{cases} 1 & \text{if } (i, j) \text{ belongs to the tour} \\ 0 & \text{otherwise} \end{cases}$
If we relax (i.e., remove) constraints \( n \), then we get a **minimum spanning tree** (MST); greedy algorithms (e.g., Kruskal)

**Heuristic**: no heuristic

**Branch**: see next

**Next**: breath-first visit

**Initialization**: \( Q := \emptyset, z := +\infty \)

**Minimization**:

We can refine the bound by adding the **minimum cost arc not belonging to the MST**: \( MST = \text{Minimum Spanning Tree} \)
\[ z = +\infty \]

Since \( z < \bar{z} \) then BRANCH:

\[ x_{12} = 0 \quad x_{14} = 0 \quad x_{15} = 0 \]

it is not a partitioning!

\[ p_1 \]

MST on a reduced graph (remove \((1,2)\))

![Graph](image)

\[ z = 7 \]

This is a Hamiltonian cycle (so a feasible solution): \( z = 7 \)

Since \( \bar{z} = \bar{z} = 7 \), then partitioning by optimality.
Since $z = 8 > z = 7$, then **penning** by bound

$\emptyset = \emptyset ; \; z = 7$
The corresponding enumeration tree:

\[
\begin{align*}
\text{P} & : z = +\infty \\
\text{P}_1 & : x_{12} = 0 \\
\text{P}_2 & : x_{14} = 0 \\
\text{P}_3 & : x_{15} = 0 \\
\text{P}_1 & : z = 4 \\
\text{P}_2 & : z = 8 \\
\text{P}_3 & : z = 8
\end{align*}
\]

Pruning by optimality, pruning by bound, pruning by bound.