Integer Programming: Branch-&-Bound Algorithm

Eduardo Camponogara

Department of Automation and Systems Engineering
Federal University of Santa Catarina

October 2016

Summary

Summary

Branch-and-Bound Algorithm

"Branch-and-bound" (B&B) is a kind of divide and conquer strategy for mixed-integer linear programming:

1. Divide P in an equivalent set of subproblems $\{SP_k\}$.
2. Solve the subproblems.
3. Obtain a solution for P from the solutions for $\{SP_k\}$.

Branch-and-Bound Algorithm

"Branch-and-bound" (B&B) is a kind of divide and conquer strategy for mixed-integer linear programming:

1. Divide P in an equivalent set of subproblems $\{SP_k\}$.
2. Solve the subproblems.
3. Obtain a solution for P from the solutions for $\{SP_k\}$.

Branch-and-Bound Algorithm

"Branch-and-bound" (B&B) is a kind of divide and conquer strategy for mixed-integer linear programming:

1. Divide P in an equivalent set of subproblems $\{SP_k\}$.
2. Solve the subproblems.
3. Obtain a solution for P from the solutions for $\{SP_k\}$.

Branch-and-Bound Algorithm

"Branch-and-bound" (B&B) is a kind of divide and conquer strategy for mixed-integer linear programming:

1. Divide P in an equivalent set of subproblems $\{SP_k\}$.
2. Solve the subproblems.
3. Obtain a solution for P from the solutions for $\{SP_k\}$.

Branch-and-Bound Algorithm

- The divisions are performed iteratively, such that the subproblems are easier to solve.
- Eliminate/Discard subproblems by implicit enumeration.
    - That is, a subproblem is discarded if it can be proven that it cannot produce the optimal solution.

Divide and Conquer

Consider the problem:

$$P: \; z = \max \{c^{\mathrm{T}} x : x \in S\}$$

How do we "break" P in small subproblems, and then recombine their solutions into a solution for the original problem?

Divide and Conquer

Consider the problem:

$$P: \; z = \max \{c^{\mathrm{T}} x : x \in S\}$$

How do we “break” P in small subproblems, and then recombine their solutions into a solution for the original problem?

Divide and Conquer

Proposition

- Let $S = S_1 \cup \ldots \cup S_K$ be a decomposition of S in K subsets.
- Let also $z^k = \max\{c^{\mathrm{T}}x : x \in S_k\}$ for $k = 1, \ldots, K$.
- Then, $z = \max\{z^k : k = 1, \ldots, K\}$.

A divide-and-conquer strategy can be illustrated with an enumeration tree (explicit).

Divide and Conquer

Proposition

- Let $S = S_1 \cup \ldots \cup S_K$ be a decomposition of S in K subsets.
- Let also $z^k = \max\{c^{\mathrm{T}}x : x \in S_k\}$ for $k = 1, \ldots, K$.
- Then, $z = \max\{z^k : k = 1, \ldots, K\}$.

A divide-and-conquer strategy can be illustrated with an enumeration tree (explicit).

Divide and Conquer

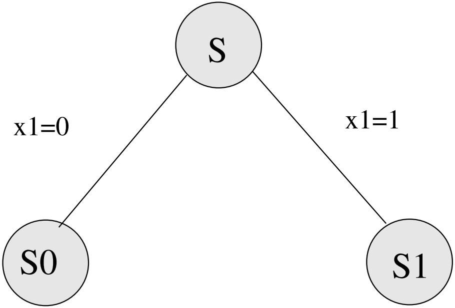
Proposition

- Let $S = S_1 \cup \ldots \cup S_K$ be a decomposition of S in K subsets.
- Let also $z^k = \max\{c^{\mathrm{T}}x : x \in S_k\}$ for $k = 1, \ldots, K$.
- Then, $z = \max\{z^k : k = 1, \ldots, K\}$.

A divide-and-conquer strategy can be illustrated with an enumeration tree (explicit).

Explicit Enumeration

For $S \subseteq \{0,1\}^3$ the enumeration tree is build as follows.



Explicit Enumeration Tree

- Clearly $S = S_0 \cup S_1$, such that:
  - $S_0 = \{x \in S : x_1 = 0\}$ e
  - $S_1 = \{x \in S : x_1 = 1\}$.
- Divide each subproblem in even smaller subproblems:
  - $S_0 = S_{00} \cup S_{01}$ and
  - $S_1 = S_{10} \cup S_{11}$,

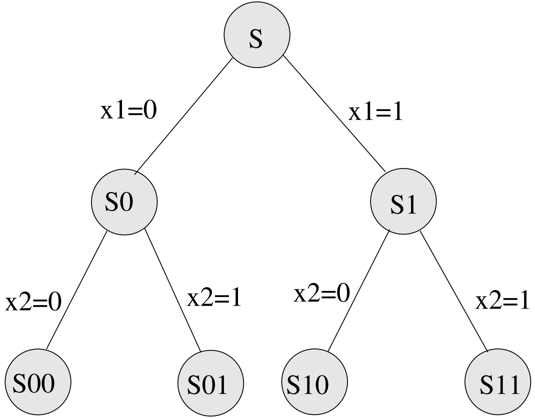
  where $S_{i_1 i_2} = \{x \in S_{i_1} : x_2 = i_2\}$.

Explicit Enumeration Tree

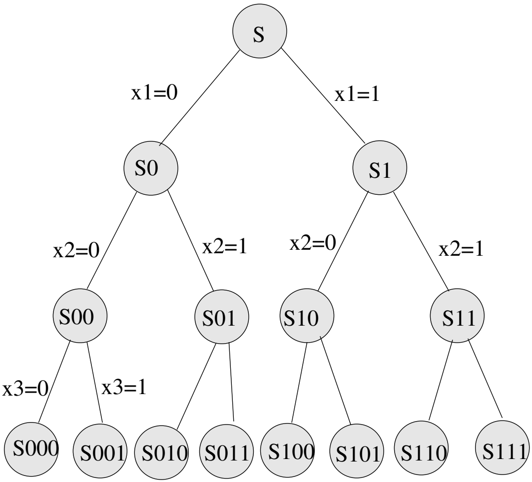
- Clearly $S = S_0 \cup S_1$, such that:
  - $S_0 = \{x \in S : x_1 = 0\}$ e
  - $S_1 = \{x \in S : x_1 = 1\}$.
- Divide each subproblem in even smaller subproblems:
  - $S_0 = S_{00} \cup S_{01}$ and
  - $S_1 = S_{10} \cup S_{11}$,

  where $S_{i_1 i_2} = \{x \in S_{i_1} : x_2 = i_2\}$.

Explicit Enumeration



Explicit Enumeration



Explicit Enumeration

- The above figure shows a complete enumeration tree.
- A leaf of the tree $S_{i_1 i_2 i_3}$ is nonempty if, and only if, $x = (i_1, i_2, i_3) \in S$.
- The leaves correspond to the candidate solutions.

Implicit Enumeration

- Complete enumeration is not viable for practical problems.
- We should use bounds for $\{z^k\}$ in an effective way, upper bounds (dual) and lower bounds (primal).

Implicit Enumeration

- Complete enumeration is not viable for practical problems.
- We should use bounds for $\{z^k\}$ in an effective way, upper bounds (dual) and lower bounds (primal).

Implicit Enumeration

Proposition

Let:

- $S = S_1 \cup \ldots \cup S_K$ be a decomposition of S in K subsets.
- $z^k = \max\{c^{\mathrm{T}}x : x \in S_k\}$ are optimal values for $k = 1, \ldots, K$.

Let:

- $\overline{z}^k$ be an upper bound for z^k.
- $\underline{z}^k$ be a lower bound for z^k.

Then:

a) $\overline{z} = \max\{\overline{z}^k : k = 1, \ldots, K\}$ defines an upper bound for z.

b) $\underline{z} = \max\{\underline{z}^k : k = 1, \ldots, K\}$ defines a lower bound for z.

Implicit Enumeration

Proposition

Let:

- $S = S_1 \cup \ldots \cup S_K$ be a decomposition of S in K subsets.
- $z^k = \max\{c^{\mathrm{T}}x : x \in S_k\}$ are optimal values for $k = 1, \ldots, K$.

Let:

- $\overline{z}^k$ be an upper bound for z^k.
- $\underline{z}^k$ be a lower bound for z^k.

Then:

a) $\overline{z} = \max\{\overline{z}^k : k = 1, \ldots, K\}$ defines an upper bound for z.

b) $\underline{z} = \max\{\underline{z}^k : k = 1, \ldots, K\}$ defines a lower bound for z.

Implicit Enumeration

Proposition

Let:

- $S = S_1 \cup \ldots \cup S_K$ be a decomposition of S in K subsets.
- $z^k = \max\{c^{\mathrm{T}}x : x \in S_k\}$ are optimal values for $k = 1, \ldots, K$.

Let:

- $\overline{z}^k$ be an upper bound for z^k.
- $\underline{z}^k$ be a lower bound for z^k.

Then:

a) $\overline{z} = \max\{\overline{z}^k : k = 1, \ldots, K\}$ defines an upper bound for z.

b) $\underline{z} = \max\{\underline{z}^k : k = 1, \ldots, K\}$ defines a lower bound for z.

Implicit Enumeration

Proposition

Let:

- $S = S_1 \cup \ldots \cup S_K$ be a decomposition of S in K subsets.
- $z^k = \max\{c^{\mathrm{T}} x : x \in S_k\}$ are optimal values for $k = 1, \ldots, K$.

Let:

- $\overline{z}^k$ be an upper bound for z^k.
- $\underline{z}^k$ be a lower bound for z^k.

Then:

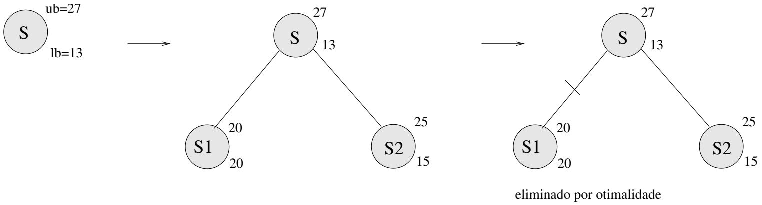
a) $\overline{z} = \max\{\overline{z}^k : k = 1, \ldots, K\}$ defines an upper bound for z.

b) $\underline{z} = \max\{\underline{z}^k : k = 1, \ldots, K\}$ defines a lower bound for z.

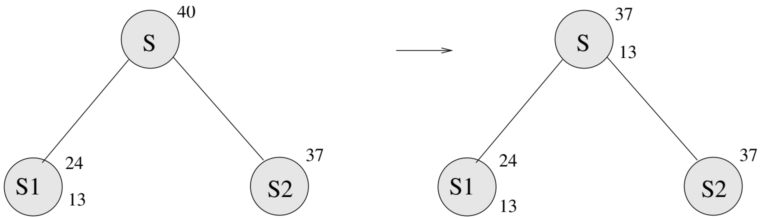
Branch-and-Bound Algorithm

- Let S be the initial set containing all problem solutions.
- Assume that $lb = 13$ is the lower bound and $ub = 27$ is the upper bound.

Cut Node by Optimality



Node Cannot Be Fathomed



Nenhum ramo da arvore pode ser eliminado

Implicit Enumeration

Three rules for cutting tree branches:

i) By optimality: $z_t = \max\{c^{\mathrm{T}}x : x \in S_t\}$ has been solved.

ii) By bounding: $\overline{z}_t < \underline{z}$.

iii) By infeasibility: $S_t = \emptyset$.

Summary

Branch-&-Bound Application

The branch-and-bound search will be illustrated in the following problem:

$$
\begin{array}{rrrrrr}
S: & z = & \max & 4x_1 & -x_2 & & \\
& & \text{s.t.}: & 7x_1 & -2x_2 & \leqslant & 14 \\
& & & & x_2 & \leqslant & 3 \\
& & & 2x_1 & -2x_2 & \leqslant & 3
\end{array}
$$

onde $x \in \mathbb{Z}^2_+$.

Branch-&-Bound Application

Bounding

The first upper bound is obtained by solving the linear relaxation, $R(S)$.

- It produces $\overline{z} = \frac{59}{7}$ at $(\overline{x}_1, \overline{x}_2) = (\frac{20}{7}, 3)$.
- We assume that $\underline{z} = -\infty$.

Branch-&-Bound Application

Branching

If $\underline{z} < \overline{z}$, S is broken in two subproblems.

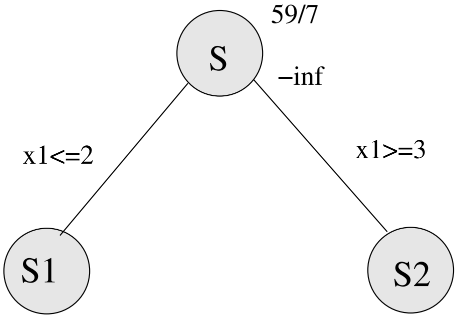
- Break S according with one fractional variable:

$$\begin{aligned} S_1 &= S \cap \{x : x_j \leqslant \lfloor \overline{x}_j \rfloor\} \\ S_2 &= S \cap \{x : x_j \geqslant \lceil \overline{x}_j \rceil\} \end{aligned}$$

- Clearly, $S = S_1 \cup S_2$.
- The list of active nodes becomes $L = \{S_1, S_2\}$.

Branch-&-Bound Application

Branching



Branch-&-Bound Application

Choosing a Node

- The list of active nodes $L = \{S_1, S_2\}$ contains two subsets.
- Arbitrarily, node S_1 is chosen.

Branch-&-Bound Application

Bounding

We solve the relaxation $R(S_1)$, meaning the LP:

$$
\begin{array}{rlrrcr}
S_1 : \quad \bar{z}_1 = & \max & 4x_1 & -x_2 & & \\
& \text{s.t.}: & 7x_1 & -2x_2 & \leqslant & 14 \\
& & & x_2 & \leqslant & 3 \\
& & 2x_1 & -2x_2 & \leqslant & 3 \\
& & x_1 & & \leqslant & 2 \\
& & x \in \mathbb{Z}_+^2 & & &
\end{array}
$$

for which the optimal solution is $(x_1^1, x_2^1) = (2, \frac{1}{2})$, thus inducing an upper bound $\bar{z} = \frac{15}{2}$.

Branch-&-Bound Application

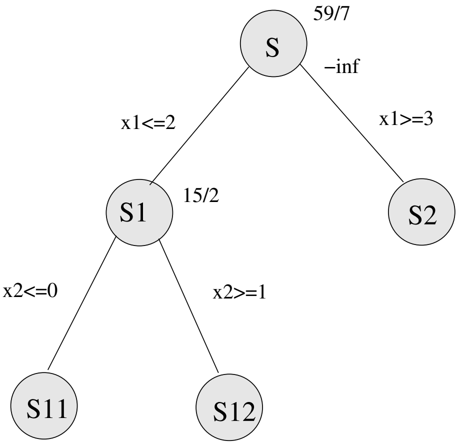
Branching

- Breaking S_1 in two sets:

$$\begin{aligned} S_{11} &= S_1 \cap \{x : x_2 \leqslant 0\} \\ S_{12} &= S_1 \cap \{x : x_2 \geqslant 1\} \end{aligned}$$

renders the active-node list $L = \{S_2, S_{11}, S_{12}\}$.

Branch-&-Bound Application



Branch-&-Bound Application

Branching

Arbitrarily we choose node S_2 from the active list $L = \{S_{11}, S_{12}, S_2\}$.

Branch-&-Bound Application

Bounding

Solving the linear relaxation $R(S_2)$:

$$
\begin{array}{llrrcr}
S_2: \quad z_2 = & \max & 4x_1 & -x_2 & & \\
& \text{s.t.}: & 7x_1 & -2x_2 & \leqslant & 14 \\
& & & x_2 & \leqslant & 3 \\
& & 2x_1 & -2x_2 & \leqslant & 3 \\
& & x_1 & & \geqslant & 3 \\
& & x \in \mathbb{Z}_+^2 & & &
\end{array}
$$

Since $R(S_2)$ is infeasible ($\bar{z}_2 = -\infty$), node S_2 is cut off.

Branch-&-Bound Application

Choosing a node

The list of active nodes has now become $L = \{S_{11}, S_{12}\}$.

- Arbitrarily we chose node S_{12}.

Branch-&-Bound Application

Bounding

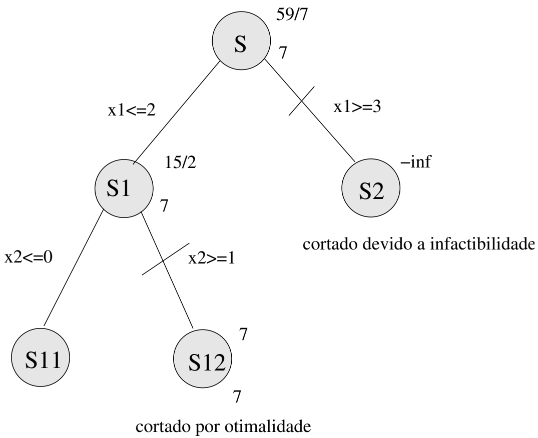
- We solve the relaxation $R(S_{12})$ with feasible space $S_{12} = S \cap \{x : x_1 \leqslant 2 \text{ and } x_2 \geqslant 1\}$.
- The solution $\overline{x}_{12} = (2, 1)$ is obtained, producing the upper bound $\overline{z}_{12} = 7$.
- This solution $R(S_{12})$ is integer, thus a lower bound was obtained:
  - The value $\underline{z}_{12} = 7$ can be propagated throughout the branch-and-bound tree.
- Consequently S_{12} is cut off by optimality.

Branch-&-Bound Application

Bounding

- We solve the relaxation $R(S_{12})$ with feasible space $S_{12} = S \cap \{x : x_1 \leqslant 2 \text{ and } x_2 \geqslant 1\}$.
- The solution $\overline{x}_{12} = (2, 1)$ is obtained, producing the upper bound $\overline{z}_{12} = 7$.
- This solution $R(S_{12})$ is integer, thus a lower bound was obtained:
  - The value $\underline{z}_{12} = 7$ can be propagated throughout the branch-and-bound tree.
- Consequently S_{12} is cut off by optimality.

Branch-&-Bound Application



Branch-&-Bound Application

Choose a node

Only S_{11} is an active node.

Branch-&-Bound Application

Bounding

- Notice that $S_{11} = S \cap \{x : x_1 \leqslant 2, x_2 = 0\}$.
- The solution to the relaxation $R(S_{11})$ is $\overline{x}_{11} = (\frac{3}{2}, 0)$, which produces an upper bound $\overline{z}_{11} = 6$.
- Since $\overline{z}_{11} = 6 < 7 = \underline{z}$, this node is cutt off.

Branch-&-Bound Application

Choosing a node

The list of active nodes is empty, thus we conclude that the solution $x^\star = (2, 1)$ is optimal with objective $z^\star = 7$.

Integer Programming: Branch-and-Bound Algorithm

- Thank you for attending this lecture!!!