Branch & Bound

Alberto Parravicini
Université libre de Bruxelles

June 1, 2017

A brief introduction...

Consider a generic optimization problem:

A brief introduction...

Consider a generic optimization problem:

$$\mathbf{max}\{\mathbf{c}(\mathbf{x}) : \mathbf{x} \in \mathbf{X}\}$$

A brief introduction...

Consider a generic optimization problem:

$$\max\{\mathbf{c}(\mathbf{x}) : \mathbf{x} \in \mathbf{X}\}$$

Often too hard to solve directly.

A brief introduction...

Consider a generic optimization problem:

$$\max\{\mathbf{c}(\mathbf{x}) : \mathbf{x} \in \mathbf{X}\}$$

Often too hard to solve directly.

Branch & Bound finds the optimal solutions thanks to:

A brief introduction...

Consider a generic optimization problem:

$$\max\{\mathbf{c}(\mathbf{x}) : \mathbf{x} \in \mathbf{X}\}$$

Often too hard to solve directly.

Branch & Bound finds the optimal solutions thanks to:

→ **Branch**: find the optimal solution by exploring smaller sub-problems.

A brief introduction...

Consider a generic optimization problem:

$$\max\{c(x) : x \in X\}$$

Often too hard to solve directly.

Branch & Bound finds the optimal solutions thanks to:

→ **Branch**: find the optimal solution by exploring smaller sub-problems.

→ **Bound**: use bounds on the objective function to implicitly enumerate all possible solutions.

Branching

Observation: if we partition the feasible region X into $\mathbf{X} = \mathbf{X_1} \cup \ldots \cup \mathbf{X_k}$ and let $\mathbf{z_i} = \mathbf{max}\{\mathbf{c}(\mathbf{x}) : \mathbf{x} \in \mathbf{X_i}\}$, then:

Branching

Observation: if we partition the feasible region X into $\mathbf{X} = \mathbf{X_1} \cup \ldots \cup \mathbf{X_k}$ and let $\mathbf{z_i} = \mathbf{max}\{\mathbf{c}(\mathbf{x}) : \mathbf{x} \in \mathbf{X_i}\}$, then:

→ $\mathbf{z} = \mathbf{max}_{\mathbf{1} \leq \mathbf{i} \leq \mathbf{z_i}}$

Branching

Observation: if we partition the feasible region X into $\mathbf{X} = \mathbf{X_1} \cup \ldots \cup \mathbf{X_k}$ and let $\mathbf{z_i} = \mathbf{max}\{\mathbf{c}(\mathbf{x}) : \mathbf{x} \in \mathbf{X_i}\}$, then:

$\rightarrow$ $\mathbf{z} = \mathbf{max}_{\mathbf{1} \leq \mathbf{i} \leq \mathbf{z_i}}$

Each subset X_i can be split even further, giving an enumeration tree.

Branching

Observation: if we partition the feasible region X into $\mathbf{X} = \mathbf{X_1} \cup \ldots \cup \mathbf{X_k}$ and let $\mathbf{z_i} = \mathbf{max}\{\mathbf{c}(\mathbf{x}) : \mathbf{x} \in \mathbf{X_i}\}$, then:

→ $\mathbf{z} = \mathbf{max}_{\mathbf{1} \leq \mathbf{i} \leq \mathbf{z_i}}$

Each subset X_i can be split even further, giving an enumeration tree.

We can't explore all possible branches though, they are too many!

Bounding

We can use *upper* and lower bounds to cut portions of the feasible region $\mathbf{X}$.

Bounding

We can use *upper* and lower bounds to cut portions of the feasible region **X**.

Observation: given $\mathbf{X} = \mathbf{X_1} \cup \ldots \cup \mathbf{X_k}$, if $\bar{z}_k$ is an *upper* bound for X_k then $max_k \bar{z}_k\}$ is an *upper* bound for X. Also, if $\underline{z}_k$ is a *lower* bound for X_k then $max_k\{\underline{z}_k\}$ is a *lower* bound for X.

Bounding

We can use *upper* and lower bounds to cut portions of the feasible region **X**.

Observation: given $\mathbf{X} = \mathbf{X_1} \cup \ldots \cup \mathbf{X_k}$, if $\bar{z}_k$ is an *upper* bound for X_k then $max_k\bar{z}_k\}$ is an *upper* bound for X. Also, if $\underline{z}_k$ is a *lower* bound for X_k then $max_k\{\underline{z}_k\}$ is a *lower* bound for X.

Lower bounds are feasible solutions, while *upper* bounds can be found by solving a relaxation.
In **ILP**, the linear relaxation will provide an upper bound $\mathbf{x}^*_{\mathbf{LP}}$.

Pruning criteria

When can we prune a node of the tree?

Pruning criteria

When can we prune a node of the tree?

→ **Optimality**: if $\bar{z}_i = \underline{z}_i$ then we have an optimal solution of X_i.

Pruning criteria

When can we prune a node of the tree?

- **Optimality**: if $\bar{z}_i = \underline{z}_i$ then we have an optimal solution of X_i.
- **Infeasibility**: if $\mathbf{X_i} = \emptyset$.

Pruning criteria

When can we prune a node of the tree?

- → **Optimality**: if $\bar{z}_i = \underline{z}_i$ then we have an optimal solution of X_i.
- → **Infeasibility**: if $\mathbf{X_i} = \emptyset$.
- → **Bounding**: if the **upper** bound $\bar{z}_i$ is $\leq$ than some other **lower** bound $\underline{z}_j$, or of the best feasible solution found so far.

In practice...

→ **How are branches defined?**

In practice...

$\rightarrow$ **How are branches defined?**

In **ILP**, pick a fractional variable $\mathbf{x_h}$ in $\mathbf{x}^*_{\mathbf{LP}}$, and define 2 sub-problems:

In practice...

→ **How are branches defined?**

In **ILP**, pick a fractional variable $\mathbf{x_h}$ in $\mathbf{x^*_{LP}}$, and define 2 sub-problems:

$z^1_{ILP} = max\{cx : x \in X, \mathbf{x_h} \leq \lfloor \mathbf{a_h} \rfloor\}$

$z^2_{ILP} = max\{cx : x \in X, \mathbf{x_h} \geq \lceil \mathbf{a_h} \rceil\}$

In practice...

→ **How are branches defined?**

In **ILP**, pick a fractional variable $\mathbf{x_h}$ in $\mathbf{x^*_{LP}}$, and define 2 sub-problems:

$z^1_{ILP} = max\{cx : x \in X, \mathbf{x_h} \leq \lfloor \mathbf{a_h} \rfloor\}$

$z^2_{ILP} = max\{cx : x \in X, \mathbf{x_h} \geq \lceil \mathbf{a_h} \rceil\}$

→ **Which variable is chosen?**

In practice...

$\rightarrow$ **How are branches defined?**

In **ILP**, pick a fractional variable $\mathbf{x_h}$ in $\mathbf{x^*_{LP}}$, and define 2 sub-problems:

$z^1_{ILP} = max\{cx : x \in X, \mathbf{x_h} \leq \lfloor \mathbf{a_h} \rfloor\}$

$z^2_{ILP} = max\{cx : x \in X, \mathbf{x_h} \geq \lceil \mathbf{a_h} \rceil\}$

$\rightarrow$ **Which variable is chosen?**

- The most fractional.

In practice...

$\rightarrow$ **How are branches defined?**

In **ILP**, pick a fractional variable $\mathbf{x_h}$ in $\mathbf{x^*_{LP}}$, and define 2 sub-problems:

$$z^1_{ILP} = max\{cx : x \in X, \mathbf{x_h} \leq \lfloor \mathbf{a_h} \rfloor\}$$
$$z^2_{ILP} = max\{cx : x \in X, \mathbf{x_h} \geq \lceil \mathbf{a_h} \rceil\}$$

$\rightarrow$ **Which variable is chosen?**

- The most fractional.
- **Strong Branching**: try all of them, keep the one with highest upper bound.

In practice...

$\rightarrow$ **How are branches defined?**

In **ILP**, pick a fractional variable $\mathbf{x_h}$ in $\mathbf{x^*_{LP}}$, and define 2 sub-problems:

$z^1_{ILP} = max\{cx : x \in X, \mathbf{x_h} \leq \lfloor \mathbf{a_h} \rfloor\}$

$z^2_{ILP} = max\{cx : x \in X, \mathbf{x_h} \geq \lceil \mathbf{a_h} \rceil\}$

$\rightarrow$ **Which variable is chosen?**

- The most fractional.
- **Strong Branching**: try all of them, keep the one with highest upper bound.
- Other (smallest sub-tour, etc...)

In practice...

→ **How are branches defined?**

In **ILP**, pick a fractional variable $\mathbf{x_h}$ in $\mathbf{x}^*_{\mathbf{LP}}$, and define 2 sub-problems:

$z^1_{ILP} = max\{cx : x \in X, \mathbf{x_h} \leq \lfloor \mathbf{a_h} \rfloor\}$
$z^2_{ILP} = max\{cx : x \in X, \mathbf{x_h} \geq \lceil \mathbf{a_h} \rceil\}$

→ **Which variable is chosen?**

- The most fractional.
- **Strong Branching**: try all of them, keep the one with highest upper bound.
- Other (smallest sub-tour, etc...)

→ **Which branch is chosen?**

In practice...

$\rightarrow$ **How are branches defined?**

In **ILP**, pick a fractional variable $\mathbf{x_h}$ in $\mathbf{x^*_{LP}}$, and define 2 sub-problems:

$z^1_{ILP} = max\{cx : x \in X, \mathbf{x_h} \leq \lfloor \mathbf{a_h} \rfloor\}$
$z^2_{ILP} = max\{cx : x \in X, \mathbf{x_h} \geq \lceil \mathbf{a_h} \rceil\}$

$\rightarrow$ **Which variable is chosen?**

- The most fractional.
- **Strong Branching**: try all of them, keep the one with highest upper bound.
- Other (smallest sub-tour, etc...)

$\rightarrow$ **Which branch is chosen?**

- **Depth First**, quickly find a feasible solution; easy to re-optimize.

In practice...

→ **How are branches defined?**

In **ILP**, pick a fractional variable $\mathbf{x_h}$ in $\mathbf{x^*_{LP}}$, and define 2 sub-problems:

$z^1_{ILP} = max\{cx : x \in X, \mathbf{x_h} \leq \lfloor \mathbf{a_h} \rfloor\}$
$z^2_{ILP} = max\{cx : x \in X, \mathbf{x_h} \geq \lceil \mathbf{a_h} \rceil\}$

→ **Which variable is chosen?**

- The most fractional.
- **Strong Branching**: try all of them, keep the one with highest upper bound.
- Other (smallest sub-tour, etc...)

→ **Which branch is chosen?**

- **Depth First**, quickly find a feasible solution; easy to re-optimize.
- **Best Bound**, find the upper bounds of the children and choose the best (slow!).

Further optimizations

→ **Generalized upper bounds**

Further optimizations

→ **Generalized upper bounds**

Used in constraints like $\sum_{i=1}^{n} x_i = 1$; fixing one $x_i = 1$ creates unbalanced tree; instead, use:

Further optimizations

→ **Generalized upper bounds**

Used in constraints like $\sum_{i=1}^{n} x_i = 1$; fixing one $x_i = 1$ creates unbalanced tree; instead, use:

$X_1 = \mathbf{x : x_i = 0, i = 0, \ldots, r}$

$X_2 = \mathbf{x : x_i = 0, i = r + 1, \ldots, n}$

$\mathbf{r = min\{t : \sum_{i=1}^{t} x_i >= \frac{1}{2}\}}$

Further optimizations

- **Generalized upper bounds**
  Used in constraints like $\sum_{i=1}^{n} x_i = 1$; fixing one $x_i = 1$ creates unbalanced tree; instead, use:
  $X_1 = \mathbf{x : x_i = 0, i = 0, \ldots, r}$
  $X_2 = \mathbf{x : x_i = 0, i = r + 1, \ldots, n}$
  $\mathbf{r = min\{t : \sum_{i=1}^{t} x_i >= \frac{1}{2}\}}$
- **Constraints preprocessing**

Further optimizations

- **Generalized upper bounds**

  Used in constraints like $\sum_{i=1}^{n} x_i = 1$; fixing one $x_i = 1$ creates unbalanced tree; instead, use:

  $X_1 = \mathbf{x} : \mathbf{x_i} = \mathbf{0}, \mathbf{i} = \mathbf{0}, \ldots, \mathbf{r}$

  $X_2 = \mathbf{x} : \mathbf{x_i} = \mathbf{0}, \mathbf{i} = \mathbf{r} + \mathbf{1}, \ldots, \mathbf{n}$

  $\mathbf{r} = \mathbf{min}\{\mathbf{t} : \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{t}} \mathbf{x_i} >= \frac{\mathbf{1}}{\mathbf{2}}\}$
- **Constraints preprocessing**
- **Branch & Cut**

Further optimizations

$\rightarrow$ **Generalized upper bounds**

Used in constraints like $\sum_{i=1}^{n} x_i = 1$; fixing one $x_i = 1$ creates unbalanced tree; instead, use:

$X_1 = \mathbf{x : x_i = 0, i = 0}, \ldots, \mathbf{r}$

$X_2 = \mathbf{x : x_i = 0, i = r + 1}, \ldots, \mathbf{n}$

$\mathbf{r = min\{t : \sum_{i=1}^{t} x_i >= \frac{1}{2}\}}$

$\rightarrow$ **Constraints preprocessing**

$\rightarrow$ **Branch & Cut**

- Combine **Gomory cuts** with **Branch & Bound**.

Further optimizations

→ **Generalized upper bounds**

Used in constraints like $\sum_{i=1}^{n} x_i = 1$; fixing one $x_i = 1$ creates unbalanced tree; instead, use:
$X_1 = \mathbf{x : x_i = 0, i = 0, \ldots, r}$
$X_2 = \mathbf{x : x_i = 0, i = r + 1, \ldots, n}$
$\mathbf{r = min\{t : \sum_{i=1}^{t} x_i >= \frac{1}{2}\}}$

→ **Constraints preprocessing**

→ **Branch & Cut**

- Combine **Gomory cuts** with **Branch & Bound**.
- At each node, add **cuts** to the relaxation of the sub-problem.

Further optimizations

→ **Generalized upper bounds**

Used in constraints like $\sum_{i=1}^{n} x_i = 1$; fixing one $x_i = 1$ creates unbalanced tree; instead, use:

$X_1 = \mathbf{x : x_i = 0, i = 0, \ldots, r}$

$X_2 = \mathbf{x : x_i = 0, i = r + 1, \ldots, n}$

$\mathbf{r = min\{t : \sum_{i=1}^{t} x_i >= \frac{1}{2}\}}$

→ **Constraints preprocessing**

→ **Branch & Cut**

- Combine **Gomory cuts** with **Branch & Bound**.
- At each node, add **cuts** to the relaxation of the sub-problem.
- Branch only when adding more cuts stops being effective.

Further optimizations

→ **Generalized upper bounds**

Used in constraints like $\sum_{i=1}^{n} x_i = 1$; fixing one $x_i = 1$ creates unbalanced tree; instead, use:

$X_1 = \mathbf{x : x_i = 0, i = 0, \ldots, r}$

$X_2 = \mathbf{x : x_i = 0, i = r + 1, \ldots, n}$

$\mathbf{r = min\{t : \sum_{i=1}^{t} x_i >= \frac{1}{2}\}}$

→ **Constraints preprocessing**

→ **Branch & Cut**

- Combine **Gomory cuts** with **Branch & Bound**.
- At each node, add **cuts** to the relaxation of the sub-problem.
- Branch only when adding more cuts stops being effective.
- Computational tradeoff, as cuts are valid only for a node and its children. Very effective in practice, however.

THANK YOU!