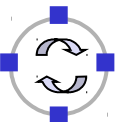


Algorithmic Aspects of Communication Networks

Chapter 3 General Optimization Methods for Network Design

Part 2



Branch and Bound – the basic algorithm

□ The idea of branch and bound methods

- Consider a (hard to solve) optimization problem

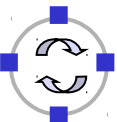
$$\text{minimize } f(x) \quad \text{subject to } x \in M \quad (1)$$

- Associate a *relaxed* optimization problem

$$\text{minimize } g(x) \quad \text{subject to } x \in R \quad (2)$$

such that

- $R \supseteq M$,
- if $x \in R$, then $g(x) = f(x)$, and
- (2) can be solved efficiently.



Branch and Bound – the basic algorithm

□ Recall:

An *optimal solution* for (1) is an element x^* of M such that $f(x) \geq f(x^*)$ for all x in M (likewise for (2)).

□ Claim 1:

(a) If y^* is an optimal solution for (2), then $f(x) \geq f(y^*)$ for all x in M .

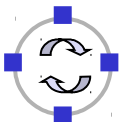
(b) If y^* is an optimal solution for (2) and $y^* \in M$, then y^* is an optimal solution for (1) too.

□ Claim 2:

Let $M = M_1 \cup M_2$ be a partition of M , and let x_k^* be an optimal solution for the OP

$$\text{minimize } f(x) \quad \text{subject to } x \in M_k \quad (1_k)$$

($k=1,2$). Then $\operatorname{argmin}_x \{f(x_1^*), f(x_2^*)\}$ is an optimal solution for (1).



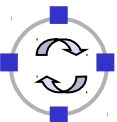
Branch and Bound – the basic algorithm

- Let R' be a nonempty subset of R , and consider the optimization problem

$$\text{minimize } g(y) \quad \text{subject to } y \in R' \quad (R').$$

- If (R') has a solution, then $\text{SOLVE}(R')$ returns a pair $(y^*, g(y^*))$ consisting of an optimal solution y^* of (R') and the corresponding value $g(y^*)$, otherwise $\text{SOLVE}(R')$ returns (n, n) .
- If $\text{SOLVE}(R')$ yields an optimal solution $y^* \notin M$, then $\text{BRANCH}(R', y^*)$ returns two disjoint subsets R'_1, R'_2 of R' such that

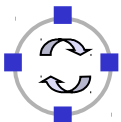
$$M \cap R' = (M \cap R'_1) \cup (M \cap R'_2).$$



Branch and Bound – the basic algorithm

The idea of the algorithm

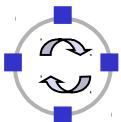
- Initialization $L \leftarrow \{R\}; best \leftarrow \infty;$
- While $L \neq \emptyset$ do
 - begin
 - choose $B \in L;$ % according to a specific rule %
 - $(y^*, g(y^*)) \leftarrow \text{SOLVE}(B);$
 - if $y^* \in M$ and $g(y^*) < best$ then
 - begin
 - $best \leftarrow g(y^*);$
 - $y_{best} \leftarrow y^*;$
 - remove B from $L;$
 - end;
 - if $g(y^*) \geq best$ then remove B from $L;$ % bounding %
 - else
 - begin % branching %
 - $(B_1, B_2) \leftarrow \text{BRANCH}(B, y^*);$
 - $L \leftarrow L \cup \{B_1, B_2\};$
 - end;
 - end



Branch and Bound – the basic algorithm

A more concrete recursive realization of Branch and Bound

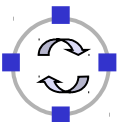
- Initialization $A \leftarrow M, B \leftarrow R, best \leftarrow \infty$
- procedure $BB(A, B, f, g)$
 - begin
 - $(y^*, g(y^*)) \leftarrow SOLVE(B)$
 - if $y^* \in A$ then
 - if $g(y^*) < best$ then
 - begin $best \leftarrow g(y^*)$; return $(y^*, g(y^*))$
 - else
 - if $g(y^*) \geq best$ then return % bounding %
 - else % branching %
 - begin
 - $(B_1, B_2) \leftarrow BRANCH(B, y^*)$;
 - $BB(A, B_1, f, g)$;
 - $BB(A, B_2, f, g)$;
 - end;
 - end



Branch and Bound – the basic algorithm

□ Remarks:

- Branch and Bound yields an (exact) optimal solution provided there is a constant K (depending on the input) such that all subproblems obtained after at most K repetitions of BRANCH either have no optimal solution or their optimal solutions are in M .
- Branch and Bound is not (necessarily) efficient.



Branch and Bound for MIP

- Consider the MIP (1)

minimize $f(\mathbf{x}) = \mathbf{c} \mathbf{x}$

subject to $\mathbf{Ax} = \mathbf{b}$,
 $\mathbf{x} \geq \mathbf{0}$, and
 x_j is integer for all $k \in I$

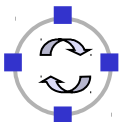
where $\mathbf{x} = (x_1, \dots, x_n)$, $\mathbf{b} \geq \mathbf{0}$, \mathbf{A} has rank $m \leq n$, and $I \subseteq \{1, \dots, n\}$

- Choose as relaxed problem the LOP (2) (called *LP-relaxation*)

minimize $f(\mathbf{x}) = \mathbf{c} \mathbf{x}$

subject to $\mathbf{Ax} = \mathbf{b}$,
 $\mathbf{x} \geq \mathbf{0}$

where $\mathbf{x} = (x_1, \dots, x_n)$, $\mathbf{b} \geq \mathbf{0}$, and \mathbf{A} has rank $m \leq n$

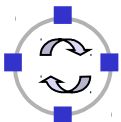


Branch and Bound for MIP

- SOLVE can be any method to solve LOPs (e.g. the simplex algorithm).
- If $(y^*, g(y^*))$ is an optimal solution where y_k^* is not an integer, then $\text{BRANCH}(B, y^*)$ 'adds' new inequalities to B , i.e.

$$\begin{aligned}\text{BRANCH}(B, y^*) &= (B_1, B_2) \text{ where} \\ B_1 &= \{y \in B \mid y_k \leq \lfloor y_k^* \rfloor\}, \\ B_2 &= \{y \in B \mid y_k \geq \lceil y_k^* \rceil\}.\end{aligned}$$

(Notation: $\lfloor y_k^* \rfloor$ is the smallest integer not smaller than y_k^* and $\lceil y_k^* \rceil$ is the greatest integer not greater than y_k^* .)



Cutting planes for MIP

- Consider the MIP (1)

$$\text{minimize} \quad f(\mathbf{x}) = \mathbf{c} \mathbf{x}$$

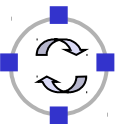
$$\text{subject to} \quad \mathbf{x} \in M = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}, \text{ and } x_j \text{ is integer for all } k \in I\}$$

$$\text{where} \quad \mathbf{x} = (x_1, \dots, x_n) \text{ and } I \subseteq \{1, \dots, n\},$$

and its LP-relaxation (2)

$$\text{minimize} \quad f(\mathbf{x}) = \mathbf{c} \mathbf{x}$$

$$\text{subject to} \quad \mathbf{x} \in R = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}.$$



Cutting planes for MIP

- A *valid cut* for (1) is an equality $\mathbf{d} \mathbf{x} \geq q$ such that
 - $\{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}, \mathbf{d} \mathbf{x} \geq q\} \neq R$ and
 - $\{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}, \text{ and } x_j \text{ is integer for all } k \in I, \mathbf{d} \mathbf{x} \geq q\} = M$.

- Outline of a cutting plane algorithm

begin

$B \leftarrow R$;

$(y^*, g(y^*)) \leftarrow \text{SOLVE}(B)$;

while $y^* \notin M$ do

begin

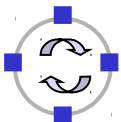
compute a valid cut $\mathbf{d} \mathbf{x} \geq q$ for (B) ;

$B \leftarrow \{\mathbf{x} \in B \mid \mathbf{d} \mathbf{x} \geq q\}$;

$(y^*, g(y^*)) \leftarrow \text{SOLVE}(B)$;

end;

end



Cutting planes for MIP

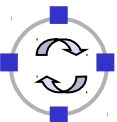
- There are two kinds of cuts:
 - problem specific ones (e.g. for the knap sack problem), and
 - general purpose ones (Gomory cuts).

- Gomory cuts
 - Let $\mathbf{x} = (x_1, \dots, x_n)$ be a basic solution of the LP-relaxation such that x_i is not an integer.
 - Then x_i is a basic variable and from the simplex tableau we know that

$$x_i + \sum a_{ij} x_j = b_i \quad (a)$$

- Equation (a) implies

$$x_i + \sum [a_{ij}] x_j - [b_i] = b_i - [b_i] - \sum (a_{ij} - [a_{ij}]) x_j \quad (b)$$



Cutting planes for MIP

- For any point $\mathbf{x} \in M$ the right hand side of (b) is less than 1, and the left hand side is an integer.
- Consequently,

$$b_i - [b_i] - \sum (a_{ij} - [a_{ij}]) x_i \leq 0 \quad (c)$$

for any point $\mathbf{x} \in M$.

- Furthermore, for the original basic solution $\mathbf{x} = (x_1, \dots, x_n)$ is

$$b_i - [b_i] - \sum (a_{ij} - [a_{ij}]) x_i = b_i - [b_i]$$

(because all non basic variables =0) not an integer, and therefore

$$b_i - [b_i] - \sum (a_{ij} - [a_{ij}]) x_i = b_i - [b_i] > 0,$$

i.e. (c) excludes \mathbf{x} .

- Hence (c) is a valid cut. The inequality (c) is called a Gomory cut.

