Short Course on Column Generation - Part I

Ana Flávia U. S. Macambira
ana.macambira@academico.ufpb.br

Alain Faye
alain.faye@ensiie.fr

Departamento de Estatística
Universidade Federal da Paraíba
École nationale supérieure d'informatique pour l'industrie et l'entreprise

Overview

1. Linear Programming - revision
   - Feasible Region
   - Simplex Method
   - Two Phase Simplex Method
   - Dual Problem
   - Complementary Slackness Theorem
   - Minkowski's Representation Theorem
2. Dantzig-Wolfe Decomposition
   - Dantzig-Wolfe Decomposition - Example
3. Column Generation in Linear Programming
   - Column Generation - Example
4. Integer Programming - revision

Linear Programming - Revision - Feasible region

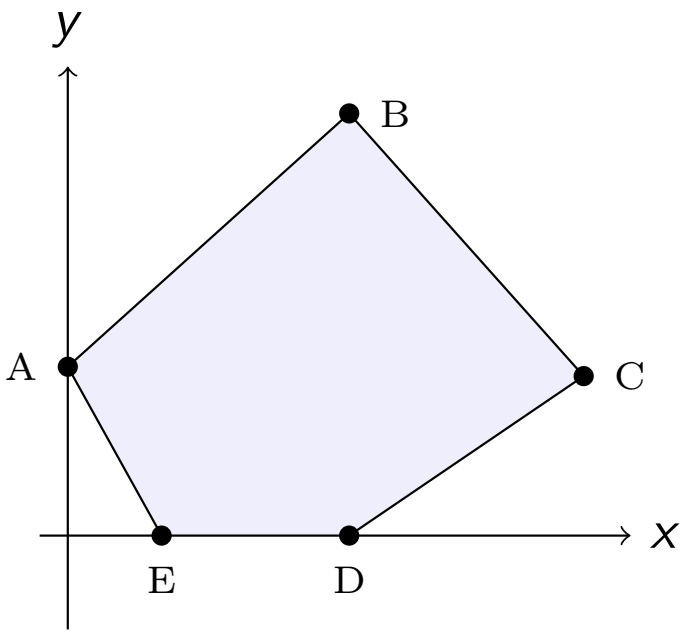
Given a linear programming problem (LP),

$$
\begin{aligned}
(LP): \text{maximize} \quad & z = c^{\top} x && (1)\\
\text{subject to:} \quad & Ax \leq b && (2)\\
& x \geq 0 && (3)
\end{aligned}
$$

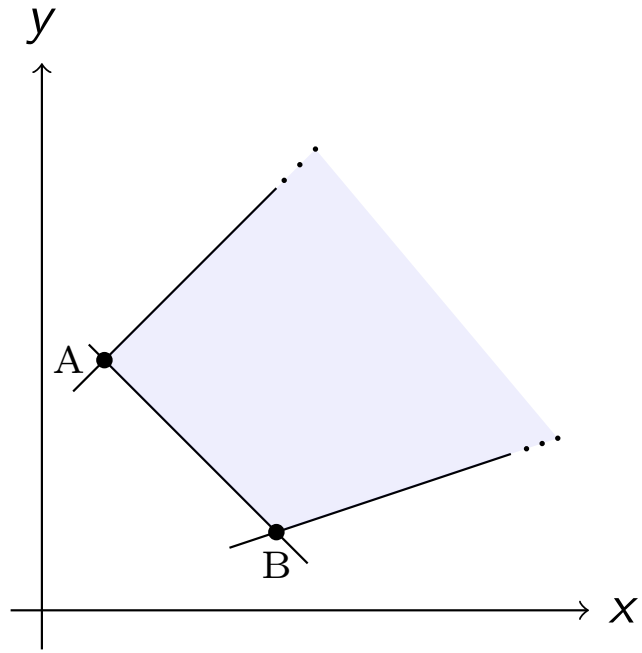
the feasible region defined by constraints (2), (3) is a convex set. This convex set can be limited or not.

Linear Programming - Revision - Feasible region

Examples



(a) Limited region



(b) Unlimited region

Linear Programming - Revision - Simplex Method

Again, consider a linear programming problem (LP),

$$\begin{aligned} \text{maximize} \quad & z = c^\top x \\ \text{subject to:} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

$A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^{m \times 1}, c \in \mathbb{R}^{n \times 1}, x \in \mathbb{R}^{n \times 1}$.
Suppose that rank$(A) = m$, thus, A has at least one square non singular submatrix of order m. We can split matrix A as $A = (B \quad N)$, where $B \in \mathbb{R}^{m \times m}$, $N \in \mathbb{R}^{m \times n-m}$ and we can suppose that B is non singular. The columns in B we call them basic columns and the columns in N, non basic columns.
We also can split vectors $x = \begin{pmatrix} x_B \\ x_N \end{pmatrix}$ and $c = \begin{pmatrix} c_B \\ c_N \end{pmatrix}$.

Linear Programming - Revision - Simplex Method

$$\begin{aligned} \text{maximize} \quad & z = c^\top x \\ \text{subject to:} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

$A = [B \;\; N]$, $x = \begin{pmatrix} x_B \\ x_N \end{pmatrix}$ and $c = \begin{pmatrix} c_B \\ c_N \end{pmatrix}$.

$$\begin{aligned} \text{maximize} \quad & z = c_B x_B + c_N x_N \\ \text{subject to:} \quad & Bx_B + Nx_N = b \\ & x_B \geq 0, x_N \geq 0 \end{aligned}$$

Linear Programming - Revision - Simplex Method

$$Bx_B + Nx_N = b$$

$$Bx_B = b - Nx_N$$

$$x_B = B^{-1}b - B^{-1}Nx_N$$

$$x_B = B^{-1}b - B^{-1}Nx_N$$

In order to start Simplex Algorithm, we need to have an initial solution, given by an initial basis, which is B. So, as the non basic variables are not participating of this solution, they assume zero value, and we get:

$$\bar{x}_B = B^{-1}b.$$

Remember that $B^{-1} \in \mathbb{R}^{m \times m}$, $b \in \mathbb{R}^{m \times 1}$. So, we find the value $\bar{x}_B$ of each variable currently at the basis by multiplying $B^{-1}b$.

Linear Programming - Revision - Simplex Method

$$\begin{aligned} \text{maximize} \quad & z = c_B x_B + c_N x_N \\ \text{subject to:} \quad & Bx_B + Nx_N = b \\ & x_B \geq 0, x_N \geq 0 \end{aligned}$$

The current solution is:

$$\bar{x}_B = B^{-1}b, \ \bar{x}_N = \mathbf{0}$$

So, the current objective value is:

$$\bar{z} = c_B \bar{x}_B = c_B B^{-1} b$$

Question: is this the current solution the optimal one? In order to answer that, we have to see if there is at least one variable which can improve the value of the objective function.

$$Bx_B + Nx_N = b$$

$$Bx_B = b - Nx_N$$

$$x_B = B^{-1}b - B^{-1}Nx_N$$

$$z = c_B x_B + c_N x_N$$

$$z = c_B(B^{-1}b - B^{-1}Nx_N) + c_N x_N$$

$$z = c_B B^{-1}b - c_B B^{-1}Nx_N + c_N x_N$$

$$z = \underbrace{c_B B^{-1}b}_{\bar{z}} + \underbrace{(c_N - c_B B^{-1}N)}_{\text{gain}} x_N$$

where $\bar{z}$ is the value of the current solution.

Taking a look at the expression of the value of gain, we have that

$$(c_N - \underbrace{c_B B^{-1}}_{\text{dual variable}} N)x_N$$

Suppose we call μ the dual variable and suppose that j are the indexes in the set N of non basic variables. So we have

$$(c_j - \pi a_j)x_j.$$

Returning to the previous equation, we have:

$$z = \underbrace{c_B B^{-1} b}_{\bar{z}} + \underbrace{(c_N - \pi N)}_{\text{gain}} x_N$$

So, if we have a minimization problem, we try to find a variable $x_j \in N$ which gives a negative reduced cost, which decreases the value of the objective function.
On the other hand, to a maximization problem we look for a variable x_j, $j \in N$ that provides a positive gain, which increases the value of the objective function.

Linear Programming - Revision - Simplex Method

- So, once we have a basic feasible solution for the problem which we want to solve, the intuitive question appears: is this the optimal solution?
- In order to answer this question, we have to look, among all non-basic variables and if there is one that improves the value of the objective function, given a criteria, which can be the variable that gives the most negative reduced cost (for minimizing problems) or the most positive gain (for maximizing problems);
- If there is no variable to enter the basis, then the current solution is optimal and the problem is solved.
- If we have found a variable to enter the basis, we look for one variable to leave the basis and the new variable will enter the basis in the same position of the one that left it, and we start another iteration of the method.

Linear Programming - Revision - Two Phase Simplex Method

- If we don't have an initial solution to start Simplex Method, we can use two phase method in order to find a basic feasible solution to the problem;
- the first phase of the method consists in inserting artificial variables to the problem and solve one problem derived from the original one but with an objective function minimizing the sum of all artificial variables we inserted at the problem. So, at the end of the first phase we find all artificial variables with value zero and as a result we have a basic feasible solution for the original problem;
- if we cannot find all artificial variables with value zero at the end of the first phase, it means that the original problem is not feasible;
- if the first phase was successful, the second phase is to solve the original problem we had.

Linear Programming - Revision - Two Phase Simplex Method

$$
\begin{aligned}
\text{maximize} \quad & 6x_1 - x_2 \\
\text{subject to:} \quad & 4x_1 + x_2 \leq 21 \\
& 2x_1 + 3x_2 \geq 13 \\
& x_1 - x_2 = -1 \\
& x_1, x_2 \geq 0
\end{aligned}
$$

Inserting the slack variables and artificial variables, we get:

$$
\begin{aligned}
\text{maximize} \quad & 6x_1 - x_2 \\
\text{subject to:} \quad & 4x_1 + x_2 + x_3 = 21 \\
& 2x_1 + 3x_2 - x_4 + x_1^a = 13 \\
& -x_1 + x_2 + x_2^a = 1 \\
& x_1, x_2, x_3, x_4, x_1^a, x_2^a \geq 0
\end{aligned}
$$

Linear Programming - Revision - Two Phase Simplex Method

Variables x_3, x_1^a e x_2^a will form the initial basis and will give the first feasible solution. So, the problem that we have to solve at the first phase is:

$$\begin{aligned}
\text{minimize} \quad & x_1^a + x_2^a \\
\text{subject to:} \quad & 4x_1 + x_2 + x_3 = 21 \\
& 2x_1 + 3x_2 - x_4 + x_1^a = 13 \\
& -x_1 + x_2 + x_2^a = 1 \\
& x_1, x_2, x_3, x_4, x_1^a, x_2^a \geq 0
\end{aligned}$$

Solving this problem we find $x_3 = 10,\ x_1 = 2,\ x_2 = 3,\ x_1^a = x_4 = x_2^a = 0$. We reached the end of the first phase, once the artificial variables have left the basis and we have a basis composed only by the variables of the original problem which gives us a feasible solution. From this point on we only have to apply Simplex Method to the original problem starting with this base and we will find the optimal solution $x_1^* = 4,\ x_2^* = 5,\ x_3^* = 0,\ x_4^* = 10$ e $z^* = 19$.

Linear Programming - Revision - Dual Problem

Given a linear programming problem, we can always find its dual problem. Suppose you have a maximization problem,

$$\begin{aligned} \text{maximize} \quad & z = c^\top x \\ \text{subject to:} \quad & Ax \leq b \\ & x \geq 0. \end{aligned}$$

We call the problem above a primal problem and its dual problem is given by

$$\begin{aligned} \text{minimize} \quad & d = b^\top \pi \\ \text{subject to:} \quad & A^\top \pi \geq c \\ & \pi \geq 0. \end{aligned}$$

So, for each primal problem, there is always a dual problem related to it, so we say that there is a pair of Primal-Dual problems.

Linear Programming - Revision - Dual Problem

Given the primal problem below, we are going to find its dual problem.

$$\begin{aligned} \text{maximize} \quad & 3x_1 + 4x_2 \\ \text{subject to:} \quad & x_1 + 2x_2 \leq 6 \\ & 4x_1 + 3x_2 \leq 12 \\ & x_1, x_2 \geq 0. \end{aligned}$$

First we assign dual variables to each constraint. Here we are going to use π_i to denote the dual variables.

$$\begin{aligned} \text{maximize} \quad & 3x_1 + 4x_2 \\ \text{subject to:} \quad & x_1 + 2x_2 \leq 6 \quad (\pi_1) \\ & 4x_1 + 3x_2 \leq 12 \quad (\pi_2) \\ & x_1, x_2 \geq 0. \end{aligned}$$

The objective function of the dual problem is: minimize $6\pi_1 + 12\pi_2$.

Linear Programming - Revision - Dual Problem

Then we transpose matrix A,

$$A = \begin{pmatrix} 1 & 2 \\ 4 & 3 \end{pmatrix}, \quad A^\top = \begin{pmatrix} 1 & 4 \\ 2 & 3 \end{pmatrix}$$

So, $A^\top \pi_i$

$$\begin{pmatrix} 1 & 4 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} \pi_1 \\ \pi_2 \end{pmatrix} = \begin{cases} \pi_1 + 4\pi_2 \\ 2\pi_1 + 3\pi_2 \end{cases}$$

Linear Programming - Revision - Dual Problem

Primal

$$\begin{aligned} \text{maximize} \quad & 3x_1 + 4x_2 \\ \text{subject to:} \quad & x_1 + 2x_2 \leq 6 \\ & 4x_1 + 3x_2 \leq 12 \\ & x_1, x_2 \geq 0. \end{aligned}$$

Dual

$$\begin{aligned} \text{minimize} \quad & 6\pi_1 + 12\pi_2 \\ \text{subject to:} \quad & \pi_1 + 4\pi_2 \geq 3 \\ & 2\pi_1 + 3\pi_2 \geq 4 \\ & \pi_1, \ \pi_2 \geq 0. \end{aligned}$$

We can observe that all coefficients of the first constraint of the dual problem are related to primal variable x_1 and the same thing happens to the second constraint of the dual problem related to x_2.

Linear Programming - Revision - Dual Problem

Consider the pair of primal-dual problems:

Primal

$$\begin{aligned} \text{maximize} \quad & z = c^\top x \\ \text{subject to:} \quad & Ax \leq b \\ & x \geq 0. \end{aligned}$$

Dual

$$\begin{aligned} \text{minimize} \quad & d = b^\top \pi \\ \text{subject to:} \quad & A^\top \pi \geq c \\ & \pi \geq 0. \end{aligned}$$

Weak duality property

If $\bar{x}_k,\ k = 1, ..., n$ is a feasible solution to the primal problem and $\bar{\pi}_i,\ i = 1, ..., m$ is a feasible solution to the dual problem, then

$$\sum_{k=1}^{n} c_k^\top \bar{x}_k \leq \sum_{i=1}^{m} b_k^\top \bar{\pi}_i.$$

Linear Programming - Revision - Dual Problem

Strong duality property.
If the primal problem has a finite optimal solution z^*, then the dual also has a finite optimal solution d^* and the values of the respective solutions are equal, $z^* = d^*$.

Primal

$$\begin{aligned} \text{maximize} \quad & z = c^\top x \\ \text{subject to:} \quad & Ax \leq b \\ & x \geq 0. \end{aligned}$$

Dual

$$\begin{aligned} \text{minimize} \quad & d = b^\top \pi \\ \text{subject to:} \quad & A^\top \pi \geq c \\ & \pi \geq 0. \end{aligned}$$

Complementary Slackness Theorem

Theorem

Let x^* and π^* be any feasible solutions to the primal and dual problems in the canonical form. Then they are respectively optimal if and only if

$$(\pi^* A - c)x^* = 0$$

and

$$\pi^*(Ax^* - b) = 0, \ i = 1, ..., m.$$

This theorem is very important in linear programming. We can see that at least one of the two terms of the multiplication must be zero in each expression, since the theorem states that the product is equal to zero.

Complementary Slackness Theorem - Example

Primal

$$
\begin{aligned}
\max \quad & 2x_1 + 3x_2 + 5x_3 + 2x_4 + 3x_5 && (4)\\
\text{s. to:} \quad & x_1 + x_2 + 2x_3 + x_4 + 3x_5 \geq 4 && (5)\\
& 2x_1 - 2x_2 + 3x_3 + x_4 + x_5 \geq 3 && (6)\\
& x_1, x_2, x_3, x_4, x_5 \geq 0.
\end{aligned}
$$

Dual

$$
\begin{aligned}
\max \quad & 4\pi_1 + 3\pi_2 && (7)\\
\text{s. to:} \quad & \pi_1 + 2\pi_2 \leq 2 && (8)\\
& \pi_1 - 2\pi_2 \leq 3 && (9)\\
& 2\pi_1 + 3\pi_2 \leq 5 && (10)\\
& \pi_1 + \pi_2 \leq 2 && (11)\\
& 3\pi_1 + \pi_2 \leq 3 && (12)\\
& \pi_1,\ \pi_2 \geq 0.
\end{aligned}
$$

The solution to the primal problem is $x_1^* = x_5^* = 1$, $x_2^* = x_3^* = x_4^* = 0$. and we are going to use the complementary slackness theorem to find the optimal value of the dual problem.

Complementary Slackness Theorem - Example

- Using equation $(\pi^* a_j - c_j)x_j^* = 0$ of the theorem and considering that we have $x_1^* \neq 0$, then we must have $\pi_1 + 2\pi_2 - 2 = 0$ which means that $\pi_1 + 2\pi_2 = 2$;
- Using the same equation and as $x_5^* \neq 0$, we have that the fifth constraint in the dual problem, constraint (12), $3\pi_1 + \pi_2 = 3$.
- Solving the system

$$\begin{cases} \pi_1 + 2\pi_2 = 2 \\ 3\pi_1 + \pi_2 = 3 \end{cases}$$

we have $\pi_1^* = \frac{4}{5}$ and $\pi_2^* = \frac{3}{5}$.

Linear Programming - Revision

Consider $X = \{x \in \mathbb{R}^n | Ax \leq b,\ x \geq 0\}, A \in \mathbb{R}^{m \times n}$, rank$(A) = m$. Let $v_1, v_2, ..., v_p$ be the vertices of X and $r_1, r_2, ..., r_q$ be the extreme radii of X.

Theorem

(Minkowski's Representation Theorem)
A point x belongs to X if and only if there exists $\lambda_j \geq 0,\ j = 1, 2, ..., p$ with $\sum_{j=1}^{p} \lambda_j = 1$ and $\mu_i \geq 0,\ i = 1, 2, ..., q$ such that

$$x = \sum_{j=1}^{p} \lambda_j v_j + \sum_{i=1}^{q} \mu_i r_i.$$

The demonstration can be seen at [1].

Dantzig-Wolfe decomposition

Consider (LP) again.

$$\begin{aligned} (LP): \text{maximize} \quad & z = c^\top x \\ \text{subject to:} \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

$c \in \mathbb{R}^{n \times 1}$, $x \in \mathbb{R}^{n \times 1}$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^{m \times 1}$.

Dantzig-Wolfe Decomposition

We can split A and b as

$$A = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix}, \; b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}.$$

$A_1 \in \mathbb{R}^{m_1 \times n}, \; A_2 \in \mathbb{R}^{m_2 \times n}, \; b_1 \in \mathbb{R}^{m_1 \times 1}, \; b_2 \in \mathbb{R}^{m_2 \times 1}$

$$\begin{aligned} (LP): \text{maximize} \quad & z = c^\top x && (13) \\ \text{subject to:} \quad & A_1 x \leq b_1 && (14) \\ & A_2 x \leq b_2 && (15) \\ & x \geq 0 && (16) \end{aligned}$$

Dantzig-Wolfe Decomposition

Suppose $X = \{x \in \mathbb{R}^n | A_2 x \leq b_2,\ x \geq 0\}$.
Accordingly to Theorem 2 we can rewrite x as:

$$x = \sum_{j=1}^{p} \lambda_j v_j + \sum_{i=1}^{q} \mu_i r_i,$$

with $\sum_{j=1}^{p} \lambda_j = 1$, $\lambda_j \geq 0,\ j = 1, 2, ..., p$ and $\mu_i \geq 0,\ i = 1, 2, ..., q$. and we can rewrite problem (LP) presented in (13)-(16) as follows.

Dantzig-Wolfe Decomposition

$$(LP1): \text{maximize} \quad z = c^\top \left(\sum_{j=1}^{p} \lambda_j v_j + \sum_{i=1}^{q} \mu_i r_i\right) \tag{17}$$

$$\text{subject to:} \quad A_1 \left(\sum_{j=1}^{p} \lambda_j v_j + \sum_{i=1}^{q} \mu_i r_i\right) \leq b_1 \tag{18}$$

$$\sum_{j=1}^{p} \lambda_j = 1 \tag{19}$$

$$\lambda_j \geq 0,\ j = 1, 2, ..., p,\ \ \mu_i \geq 0,\ \ i = 1, 2, ..., q \tag{20}$$

which can also be rewritten as follows.

Dantzig-Wolfe Decomposition

$$(LP1): \text{maximize} \quad z = \sum_{j=1}^{p} (c^\top v_j)\lambda_j + \sum_{i=1}^{q} (c^\top r_i)\mu_i \tag{21}$$

$$\text{subject to:} \quad \sum_{j=1}^{p} (A_1 v_j)\lambda_j + \sum_{i=1}^{q} (A_1 r_i)\mu_i \leq b_1 \tag{22}$$

$$\sum_{j=1}^{p} \lambda_j = 1 \tag{23}$$

$$\lambda_j \geq 0,\ j = 1, 2, ..., p,\ \ \mu_i \geq 0,\ \ i = 1, 2, ..., q \tag{24}$$

This is the Dantzig-Wolfe reformulation for problem (LP) (13)-(16).

Column Generation in Linear Programming

- Problem (LP) (13)-(16) is solved using Simplex method. It means that, at each iteration you look for the variable which will improve the value of your objective function until there is no variable to improve it;
- The problem (LP1), (21)-(24) is solved also by using Simplex method but now you are aiming at finding the best vertex. To look for the best vertex means you are now trying to find a new column, that is, a new combination of variables' values which will improve your solution at each Simplex iteration.
- Since you moved from analysing n variables at each iteration to analysing combinations of a subset of the n variables (which form the columns), Column Generation works for Integer Programming problems. We are going to see an example of Column Generation for linear programming only to understand how it works.

Dantzig-Wolfe Decomposition

There is a complete example with vertices and radii in reference [1]. The constraints of the example that we are going to see form a limited region, therefore this example only has vertices and can be found at reference [2].

Example

$$
\begin{aligned}
\text{maximize} \quad & x_1 + x_2 \\
\text{subject to:} \quad & x_1 - x_2 \leq 2 && (25) \\
& 4x_1 + 9x_2 \leq 18 && (26) \\
& -2x_1 + 4x_2 \leq 4 && (27) \\
& x_1, x_2 \geq 0.
\end{aligned}
$$

$A_1 x \leq b_1$ is represented by constraint (25) and $A_2 x \leq b_2$ is represented by constraints (26) and (27).

Example

The extreme points of the polytope $\{x \in \mathbb{R}^2_+ \mid A_2 x \leq b_2,\ x \geq 0\}$ are:

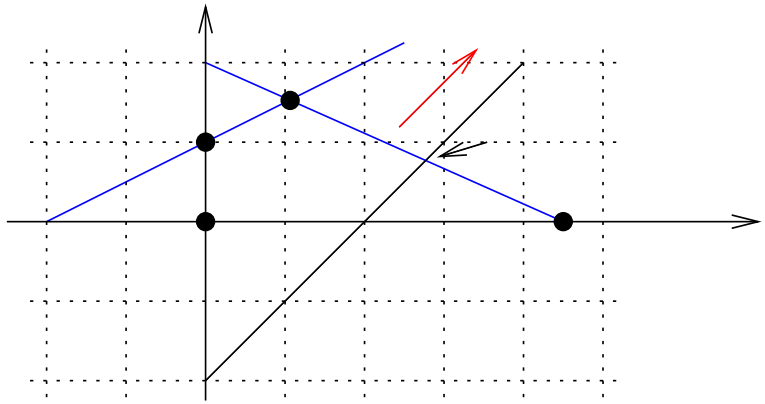
$$v_1 = (0,0)^\top, \quad v_2 = (\frac{9}{2}, 0)^\top, v_3 = (\frac{18}{17}, \frac{26}{17})^\top \text{ and } v_4 = (0,1)^\top.$$

So, every point of the polytope given by $A_2 x \leq b_2,\ x \geq 0$ can be written as a function of its vertices as:

$$(x_1, x_2) = (0,0)^\top \lambda_1 + (\frac{9}{2}, 0)^\top \lambda_2 + (\frac{18}{17}, \frac{26}{17})^\top \lambda_3 + (0,1)^\top \lambda_4$$

$\lambda_i \geq 0$ for all $i \in \{1,2,3,4\}$ and also $\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1$

Example



- the black points indicate the vertices v_1, v_2, v_3, v_4 of the polytope given by $A_2x \leq b_2,\ x \geq 0$;
- the black line is associated to $A_1x = b_1,\ x \geq 0$;
- the black arrow indicates the optimal point with coordinates $(\frac{36}{13}, \frac{10}{13})$;
- the optimal point is at the intersection of $x_1 - x_2 = 2$ and the polytope given by $A_2x \leq b_2,\ x \geq 0$.

Example

Remembering that $A_1 x \leq b_1$ is represented by constraint $x_1 - x_2 \leq 2$, so $A_1 = (1 \quad -1)$. Thus,

- $A_1 v_1 = (1 \quad -1) \begin{pmatrix} 0 \\ 0 \end{pmatrix} = 0;$
- $A_1 v_2 = (1 \quad -1) \begin{pmatrix} \frac{9}{2} \\ 0 \end{pmatrix} = \frac{9}{2};$
- $A_1 v_3 = (1 \quad -1) \begin{pmatrix} \frac{18}{17} \\ \frac{26}{17} \end{pmatrix} = \frac{-8}{17};$
- $A_1 v_4 = (1 \quad -1) \begin{pmatrix} 0 \\ 1 \end{pmatrix} = -1.$

Example

Rewriting the objective function, we have:

- $cv_1 = (1 \;\; 1)^\top \begin{pmatrix} 0 \\ 0 \end{pmatrix} = 0;$
- $cv_2 = (1 \;\; 1)^\top \begin{pmatrix} \frac{9}{2} \\ 0 \end{pmatrix} = \frac{9}{2};$
- $cv_3 = (1 \;\; 1)^\top \begin{pmatrix} \frac{18}{17} \\ \frac{26}{17} \end{pmatrix} = \frac{44}{17};$
- $cv_4 = (1 \;\; 1)^\top \begin{pmatrix} 0 \\ 1 \end{pmatrix} = 1.$

Example

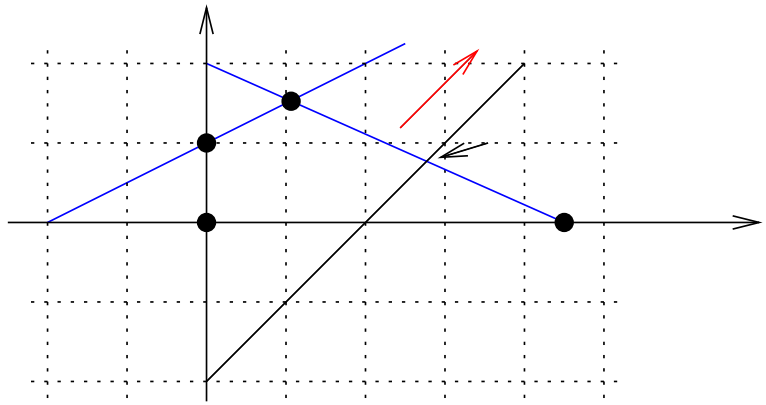
As a result, we have the following master problem:

$$
\begin{aligned}
\text{maximize} \quad & 0\lambda_1 + \frac{9}{2}\lambda_2 + \frac{44}{17}\lambda_3 + \lambda_4 \\
\text{subject to:} \quad & 0\lambda_1 + \frac{9}{2}\lambda_2 - \frac{8}{17}\lambda_3 - \lambda_4 \leq 2 \\
& \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1 \\
& \lambda_1, \lambda_2, \lambda_3, \lambda_4 \geq 0
\end{aligned}
$$

- This master problem can be solved directly using Simplex method, and we obtain $\lambda_1 = \lambda_4 = 0, \lambda_2 = 0.497$ and $\lambda_3 = 0.503$;
- Remember that $(x_1, x_2) = (0,0)\lambda_1 + (\frac{9}{2}, 0)\lambda_2 + (\frac{18}{17}, \frac{26}{17})\lambda_3 + (0,1)\lambda_4$;
- $(x_1, x_2) = (0,0) \times 0 + (\frac{9}{2}, 0) \times 0.497 + (\frac{18}{17}, \frac{26}{17}) \times 0.503 + (0,1) \times 0$;
- $(x_1, x_2) = (2.2365, 0) + (0.5326, 0.769) = (2.7691, 0.769) = (\frac{36}{13}, \frac{10}{13})$.

Example

- You can see that the optimal solution $(x_1, x_2) = (\frac{36}{13}, \frac{10}{13})$, indicated by the black arrow, is not a vertex of the polytope $A_2 x \leq b_2,\ x \geq 0$ but it still can be written as a convex combination of its vertices $v_2 = (\frac{9}{2}, 0)$ and $v_3 = (\frac{18}{17}, \frac{26}{17})$.



- This observation tells us that we don't need to know all vertices of the problem to solve when we use Dantzig-Wolfe decomposition.

Dantzig-Wolfe Decomposition

- It is a method for solving linear problems that have a block structure;
- We split the original problem in one pricing problem for each block structure and a restricted master problem.

Column Generation

- Now, imagine you have a problem with a huge number of variables and we had rewritten this problem in function of vertices or vertices and rays;
- If we use Simplex method to solve this, the step related to the computation of the reduced cost/gain would take much effort, once the number of variables at the reformulated problem is much larger than at the original problem (because it is equal to the number of vertices of the polytope);
- Therefore, the best way to solve this kind of problem is to use the automatic column generation;
- Automatic column generation in linear programming were first presented at [3], [4] and [5].

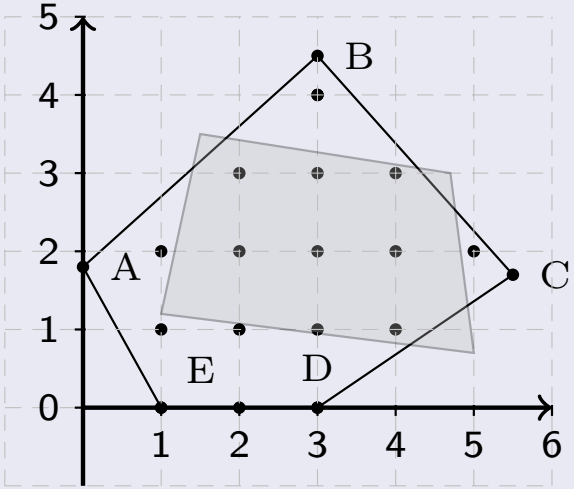
Column Generation

- For the automatic column generation, instead looking among all variables of the problem in order to find the ones which will enter the basis, we do an implicit search for a new column to be added to the master problem;
- the implicit search for a new column is the role of the Auxiliary problem, also called pricing problem;
- If there isn't any column to improve the current solution, it means that we reached the optimal solution of the original problem.

Column Generation

Suppose a problem with a limited feasible region

$$
\begin{aligned}
(PL): \text{maximize} \quad & z = c^\top x \\
\text{subject to:} \quad & A_1 x \leq b_1 \\
& A_2 x \leq b_2 \\
& x \geq 0
\end{aligned}
$$



A Dantzig-Wolfe decomposition of (PL)

$$
\begin{aligned}
(PLDW): \text{maximize} \quad & z = \sum_{v_j \in \mathcal{V}} (c^\top v_j)\lambda_j \\
\text{subject to:} \quad & \sum_{v_j \in \mathcal{V}} (A_1 v_j)\lambda_j \leq b_1 \\
& \sum_{j=1}^{p} \lambda_j = 1 \\
& \lambda_j \geq 0, \quad j = 1, 2, ..., p
\end{aligned}
$$

where $\mathcal{V}$ is the set of vertices of the polytope given by constraints $A_2 x \leq b_2$.

Column Generation

Consider $\mathcal{V}^{'} \subset \mathcal{V}$ a subset of columns (vertices) of the original problem, which includes at least a set of basic variables related to one basic feasible solution of the original problem. According to that we have the Restricted Master Problem (RMP)

$$(RMP) : \text{maximize} \quad \overline{z} = \sum_{v_j \in \mathcal{V}'} (c^{\top} v_j)\lambda_j \tag{28}$$

$$\text{subject to:} \quad \sum_{v_j \in \mathcal{V}'} (A_1 v_j)\lambda_j \leq b_1 \tag{29}$$

$$\sum_{j : v_j \in \mathcal{V}'} \lambda_j = 1 \tag{30}$$

$$\lambda_j \geq 0, \quad v_j \in \mathcal{V}^{'}. \tag{31}$$

Column Generation

- If we don't have an initial basic feasible solution, we can use artificial columns or heuristics, among other methods.
- It's important to point out that, in this case (maximization), (RMP) gives a lower bound (LB) of value $\overline{z}$ to the optimal solution of the original problem, so z^*, logo $\overline{z} \leq z^*$.

Column Generation

- When we solve the auxiliary problem (AP), we are trying to find a new column $v \in \mathcal{V} \setminus \mathcal{V}'$ which will improve the value of the objective function.
- Let π be the dual variable associated to constraints $\sum_{v_j \in \mathcal{V}'} (A_1 v_j)\lambda_j \leq b_1$ of (RMP);
- Let ν the dual variable associated to the constraint $\sum_{j:v_j \in \mathcal{V}'} \lambda_j = 1$ of (RMP).

$$
\begin{aligned}
(AP): \text{maximize} \quad & (c^\top - \pi A_1)v - \nu && (32)\\
\text{subject to:} \quad & A_2 v \leq b_2 && (33)\\
& v \geq 0 && (34)
\end{aligned}
$$

Observation

$$(AP): \text{maximize} \quad (c^\top - \pi A_1)v - \nu \tag{35}$$
$$\text{subject to:} \quad A_2 v \leq b_2 \tag{36}$$
$$v \geq 0 \tag{37}$$

The auxiliary problem rates the vertices of feasible region of the subproblem accordingly to the master problem. This is the role of the dual variables of the master problem, which are present here, at the auxiliary problem. And that's why we have always a different vertex generated when solving the auxiliary problem.

Solving the master problem using of Column Generation

First step

Use a small subset of variables containing at least one column that produces a feasible solution for the master problem;

Second step

Solve the current (RMP), also obtaining the values of the dual variables.

Third step

Solve the auxiliary problem. If we find a positive value for the objective function of the auxiliary problem, we go to Fourth step. If we find zero as a value for the objective function of the auxiliary problem, we conclude that there is no column to be added to the master problem and thus the current solution is optimal.

Fourth step

If the auxiliary problem has returned a new column, insert this column to the master problem and solve it. Go to Second step.

Example

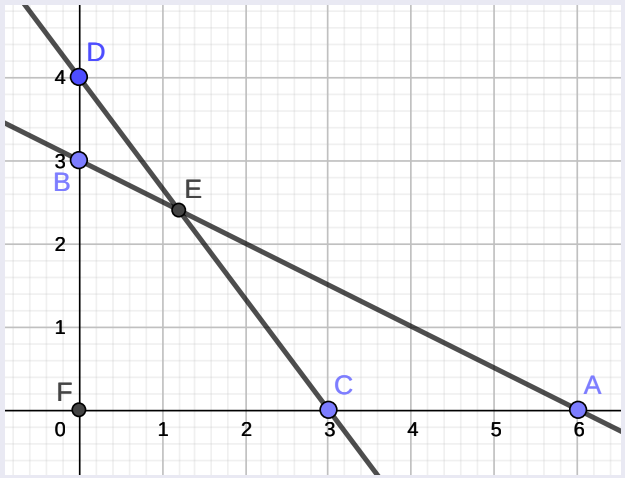
$$
\begin{aligned}
\text{maximize} \quad & 3x_1 + 4x_2 + 3x_3 + 2x_4 \\
\text{subject to:} \quad & x_1 + 2x_2 \leq 6 && (38)\\
& 4x_1 + 3x_2 \leq 12 && (39)\\
& 2x_3 + x_4 \leq 8 && (40)\\
& x_3 + 5x_4 \leq 10 && (41)\\
& x_1 + x_2 + x_3 + x_4 \leq 7 && (42)\\
& 2x_1 + x_2 + x_3 + 3x_4 \leq 17 && (43)\\
& x_1, x_2, x_3, x_4 \geq 0.
\end{aligned}
$$

- Constraints (38) and (39) are in function of x_1 and x_2 and (40) and (41) are in function of x_3 and x_4, which we can see as auxiliary problem 1 and auxiliary problem 2;
- The master problem is given by constraints (42) and (43).

Example - Subproblem 1

$$
\begin{aligned}
\text{maximize} \quad & 3x_1 + 4x_2 \\
\text{subject to:} \quad & x_1 + 2x_2 \leq 6 \\
& 4x_1 + 3x_2 \leq 12 \\
& x_1, x_2 \geq 0.
\end{aligned}
$$

Extreme points:
$F = (0, 0)$
$B = (0, 3)$
$E = (1.2, 2.4)$
$C = (3, 0)$.



Example - original problem

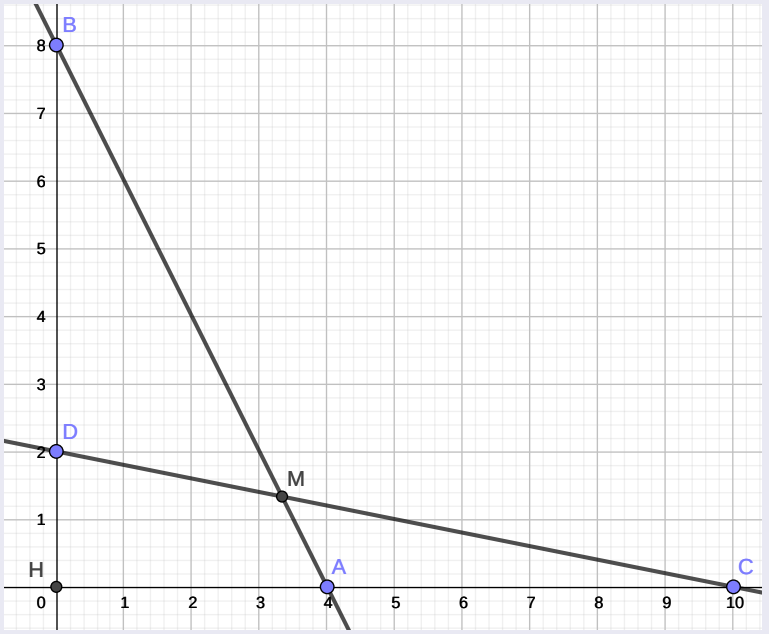
$$
\begin{aligned}
\text{maximize} \quad & 3x_1 + 4x_2 + 3x_3 + 2x_4 \\
\text{subject to:} \quad & x_1 + 2x_2 \leq 6 \\
& 4x_1 + 3x_2 \leq 12 \\
& 2x_3 + x_4 \leq 8 \\
& x_3 + 5x_4 \leq 10 \\
& x_1 + x_2 + x_3 + x_4 \leq 7 \\
& 2x_1 + x_2 + x_3 + 3x_4 \leq 17 \\
& x_1, x_2, x_3, x_4 \geq 0.
\end{aligned}
$$

Here is the original problem just to remember.

Example - Subproblem 2

$$\begin{aligned} \text{maximize} \quad & 3x_3 + 2x_4 \\ \text{subject to:} \quad & 2x_3 + x_4 \leq 8 \\ & x_3 + 5x_4 \leq 10 \\ & x_3, x_4 \geq 0. \end{aligned}$$

Extreme points:
$H = (0, 0)$
$D = (0, 2)$
$M = (3.3333.., 1.3333...)$
$A = (4, 0)$.



Example

- In general the problems are huge and the block structure allows us to choose between entering one column at each iteration or as many columns as are the number of subproblems;
- We are going to see the solution of this example first adding one column at each iteration and then, adding two columns (one for each subproblem) at each iteration.

Example - first iteration master problem - one column

Let's begin with the column $(0, 0, 0, 2)$, which means $x_1, x_2, x_3 = 0$ and $x_4 = 2$.
Objective function:

$$(3 \quad 4 \quad 3 \quad 2) \begin{pmatrix} 0 \\ 0 \\ 0 \\ 2 \end{pmatrix} \lambda_1.$$

Constraints:

$$(1 \quad 1 \quad 1 \quad 1) \begin{pmatrix} 0 \\ 0 \\ 0 \\ 2 \end{pmatrix} \lambda_1 \leq 7.$$

$$(2 \quad 1 \quad 1 \quad 3) \begin{pmatrix} 0 \\ 0 \\ 0 \\ 2 \end{pmatrix} \lambda_1 \leq 17.$$

First iteration master problem - one column

$$
\begin{aligned}
\text{maximize} \quad & 4\lambda_1 \\
\text{subject to:} \quad & 2\lambda_1 \leq 7 \\
& 6\lambda_1 \leq 17 \\
& \lambda_1 = 1 \\
& \lambda_1 \geq 0.
\end{aligned}
$$

$\lambda_1^* = 1, \bar{z} = 4, \pi_1^* = \pi_2^* = 0, \nu_1^* = 4.$

Example

- Once we have solved the master problem, the intuitive question is: is this the optimal solution?
- The auxiliary problem aims at finding a column which gives a positive reduced cost (in our case here), as we have seen at the revision.
- As we are inserting one column at each iteration, we are going to have one auxiliary problem;
- If the auxiliary problem doesn't return any column, so the solution found for the master problem is optimal.

Example - First iteration auxiliary problem - one column

$$
\begin{aligned}
\text{maximize} \quad & (3-\pi_1-2\pi_2)x_1+(4-\pi_1-\pi_2)x_2+(3-\pi_1-\pi_2)x_3+(2-\pi_1-3\pi_2)x_4-\nu_1 \\
\text{subject to:} \quad & x_1+2x_2 \leq 6 \\
& 4x_1+3x_2 \leq 12 \\
& 2x_3+x_4 \leq 8 \\
& x_3+5x_4 \leq 10 \\
& x_1,x_2,x_3,x_4 \geq 0.
\end{aligned}
$$

Remember that $\pi_1=\pi_2=0, \nu_1=4$.

$$
\begin{aligned}
\text{maximize} \quad & 3x_1+4x_2+3x_3+2x_4-4 \\
\text{subject to:} \quad & x_1+2x_2 \leq 6 \\
& 4x_1+3x_2 \leq 12 \\
& 2x_3+x_4 \leq 8 \\
& x_3+5x_4 \leq 10 \\
& x_1,x_2,x_3,x_4 \geq 0.
\end{aligned}
$$

$x_1^*=1.2,\ x_2^*=2.4,\ x_3^*=3.33333,\ x_4^*=1.33333\ OF=21.8666667.$

Example - Second iteration master problem - one column

Objective function:

$$\begin{pmatrix} 3 & 4 & 3 & 2 \end{pmatrix} \begin{pmatrix} 1.2 \\ 2.4 \\ 3.33333 \\ 1.33333 \end{pmatrix} \lambda_2.$$

Constraints:

$$\begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1.2 \\ 2.4 \\ 3.33333 \\ 1.33333 \end{pmatrix} \lambda_2 \leq 7.$$

$$\begin{pmatrix} 2 & 1 & 1 & 3 \end{pmatrix} \begin{pmatrix} 1.2 \\ 2.4 \\ 3.33333 \\ 1.33333 \end{pmatrix} \lambda_2 \leq 17.$$

Second iteration master problem - one column

$$
\begin{aligned}
\text{maximize} \quad & 4\lambda_1 + 25.8666665\lambda_2 \\
\text{subject to:} \quad & 2\lambda_1 + 8.266666\lambda_2 \leq 7 \\
& 6\lambda_1 + 12.1333332\lambda_2 \leq 17 \\
& \lambda_1 + \lambda_2 = 1 \\
& \lambda_1,\ \lambda_2 \geq 0.
\end{aligned}
$$

$\lambda_1^* = 0.20213,\ \lambda_2^* = 0.79787,\ \bar{z} = 21.44680,$

$\pi_1^* = 3.1290,\ \pi_2^* = 0,\ \nu_1^* = 0.$

Second iteration auxiliary problem - one column

$$
\begin{aligned}
\text{maximize} \quad & (3-\pi_1-2\pi_2)x_1+(4-\pi_1-\pi_2)x_2+(3-\pi_1-\pi_2)x_3+(2-\pi_1-3\pi_2)x_4-\nu_1 \\
\text{subject to:} \quad & x_1+2x_2 \leq 6 \\
& 4x_1+3x_2 \leq 12 \\
& 2x_3+x_4 \leq 8 \\
& x_3+5x_4 \leq 10 \\
& x_1,x_2,x_3,x_4 \geq 0.
\end{aligned}
$$

Remember that $\pi_1^* = 3.1290,\ \pi_2^* = 0,\ \nu_1^* = 0.$

$$
\begin{aligned}
\text{maximize} \quad & -0.129x_1+0.871x_2-0.129x_3-1.129x_4 \\
\text{subject to:} \quad & x_1+2x_2 \leq 6 \\
& 4x_1+3x_2 \leq 12 \\
& 2x_3+x_4 \leq 8 \\
& x_3+5x_4 \leq 10 \\
& x_1,x_2,x_3,x_4 \geq 0.
\end{aligned}
$$

$x_1^* = 0,\ x_2^* = 3,\ x_3^* = x_4^* = 0\ OF = 2.613.$

Example - Third iteration master problem - one column

Objective function:

$$\begin{pmatrix} 3 & 4 & 3 & 2 \end{pmatrix} \begin{pmatrix} 0 \\ 3 \\ 0 \\ 0 \end{pmatrix} \lambda_3.$$

Constraints:

$$\begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 3 \\ 0 \\ 0 \end{pmatrix} \lambda_3 \leq 7.$$

$$\begin{pmatrix} 2 & 1 & 1 & 3 \end{pmatrix} \begin{pmatrix} 0 \\ 3 \\ 0 \\ 0 \end{pmatrix} \lambda_3 \leq 17.$$

Third iteration master problem - one column

$$
\begin{aligned}
\text{maximize} \quad & 4\lambda_1 + 25.8666665\lambda_2 + 12\lambda_3 \\
\text{subject to:} \quad & 2\lambda_1 + 8.266666\lambda_2 + 3\lambda_3 \leq 7 \\
& 6\lambda_1 + 12.1333332\lambda_2 + 3\lambda_3 \leq 17 \\
& \lambda_1 + \lambda_2 + \lambda_3 = 1 \\
& \lambda_1,\ \lambda_2,\ \lambda_3 \geq 0.
\end{aligned}
$$

$\lambda_1^* = 0,\ \lambda_2^* = 0.759493,\ \lambda_3 = 0.24050,\ \bar{z} = 22.5316,$

$\pi_1^* = 2.6329,\ \pi_2^* = 0,\ \nu_1^* = 4.1012.$

Third iteration auxiliary problem - one column

$$
\begin{aligned}
\text{maximize} \quad & (3-\pi_1-2\pi_2)x_1+(4-\pi_1-\pi_2)x_2+(3-\pi_1-\pi_2)x_3+(2-\pi_1-3\pi_2)x_4-\nu_1 \\
\text{subject to:} \quad & x_1+2x_2 \leq 6 \\
& 4x_1+3x_2 \leq 12 \\
& 2x_3+x_4 \leq 8 \\
& x_3+5x_4 \leq 10 \\
& x_1,x_2,x_3,x_4 \geq 0.
\end{aligned}
$$

Remember that $\pi_1^* = 2.6329,\ \pi_2^* = 0,\ \nu_1^* = 4.1012.$

$$
\begin{aligned}
\text{maximize} \quad & 0.3671x_1+1.3671x_2+0.3671x_3-0.6329x_4-4.1012 \\
\text{subject to:} \quad & x_1+2x_2 \leq 6 \\
& 4x_1+3x_2 \leq 12 \\
& 2x_3+x_4 \leq 8 \\
& x_3+5x_4 \leq 10 \\
& x_1,x_2,x_3,x_4 \geq 0.
\end{aligned}
$$

$x_1^* = 0,\ x_2^* = 3,\ x_3^* = 4,\ x_4^* = 0\ OF = 1.4685.$

Example - Fourth iteration master problem - one column

Objective function:

$$(3 \quad 4 \quad 3 \quad 2) \begin{pmatrix} 0 \\ 3 \\ 4 \\ 0 \end{pmatrix} \lambda_4.$$

Constraints:

$$(1 \quad 1 \quad 1 \quad 1) \begin{pmatrix} 0 \\ 3 \\ 4 \\ 0 \end{pmatrix} \lambda_4 \leq 7.$$

$$(2 \quad 1 \quad 1 \quad 3) \begin{pmatrix} 0 \\ 3 \\ 4 \\ 0 \end{pmatrix} \lambda_4 \leq 17.$$

Fourth iteration master problem - one column

$$
\begin{aligned}
\text{maximize} \quad & 4\lambda_1 + 25.8666665\lambda_2 + 12\lambda_3 + 24\lambda_4 \\
\text{subject to:} \quad & 2\lambda_1 + 8.266666\lambda_2 + 3\lambda_3 + 7\lambda_4 \leq 7 \\
& 6\lambda_1 + 12.1333332\lambda_2 + 3\lambda_3 + 7\lambda_4 \leq 17 \\
& \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1 \\
& \lambda_1,\ \lambda_2,\ \lambda_3,\ \lambda_4 \geq 0.
\end{aligned}
$$

$\lambda_1^* = \lambda_2^* = \lambda_3 = 0,\ \lambda_4 = 1,\ \bar{z} = 24,$

$\pi_1^* = 3,\ \pi_2^* = 0,\ \nu_1^* = 3.$

Fourth iteration auxiliary problem - one column

$$
\begin{aligned}
\text{maximize} \quad & (3-\pi_1-2\pi_2)x_1+(4-\pi_1-\pi_2)x_2+(3-\pi_1-\pi_2)x_3+(2-\pi_1-3\pi_2)x_4-\nu_1 \\
\text{subject to:} \quad & x_1+2x_2 \leq 6 \\
& 4x_1+3x_2 \leq 12 \\
& 2x_3+x_4 \leq 8 \\
& x_3+5x_4 \leq 10 \\
& x_1,x_2,x_3,x_4 \geq 0.
\end{aligned}
$$

Remember that $\pi_1^* = 3,\ \pi_2^* = 0,\ v_1^* = 3.$

$$
\begin{aligned}
\text{maximize} \quad & x_2-x_4-3 \\
\text{subject to:} \quad & x_1+2x_2 \leq 6 \\
& 4x_1+3x_2 \leq 12 \\
& 2x_3+x_4 \leq 8 \\
& x_3+5x_4 \leq 10 \\
& x_1,x_2,x_3,x_4 \geq 0.
\end{aligned}
$$

$x_1^* = 0,\ x_2^* = 3,\ x_3^* = 0,\ x_4^* = 0\ FO = 0$. We can see that there is no column which improves the value of the objective function of the master problem. The conclusion is that we already reached the optimal solution.

Example - optimal solution

The optimal solution is:

$$x_1^* = 0,\ x_2^* = 3,\ x_3^* = 4,\ x_4^* = 0$$

$$\lambda_1^* = \lambda_2^* = \lambda_3 = 0,\ \lambda_4 = 1,\ \bar{z} = 24$$

and we can see that it is a vertex from subproblems 1 and 2, because it's not a combination of vertices, since we have only $\lambda_4 = 1$, otherwise we would have more than one λ_i with positive value.

Example

- Now we are going to solve the same problem but adding two columns at each iteration;
- it's a choice between solve one big auxiliary problem or solve many smaller problems of it;
- as the subproblems are idependent of each other, we can parallelize their execution (it will be very clear to see at the example);
- we are going to start at the same vertex we did previously.

Example - original problem

$$
\begin{aligned}
\text{maximize} \quad & 3x_1 + 4x_2 + 3x_3 + 2x_4 \\
\text{subject to:} \quad & x_1 + 2x_2 \leq 6 \\
& 4x_1 + 3x_2 \leq 12 \\
& 2x_3 + x_4 \leq 8 \\
& x_3 + 5x_4 \leq 10 \\
& x_1 + x_2 + x_3 + x_4 \leq 7 \\
& 2x_1 + x_2 + x_3 + 3x_4 \leq 17 \\
& x_1, x_2, x_3, x_4 \geq 0.
\end{aligned}
$$

Example - First iteration master problem

Let's begin with the columns $(0,0)$ and $(0,2)$ which means $x_1, x_2, x_3 = 0$ and $x_4 = 2$. Below we calculate the coefficients of the master problem using the two first columns. We are going to use λ_{i12} for variables 1 and 2 given by the first column if the i-th iteration and λ_{i34}.
Objective function:

$$(3 \quad 4)\begin{pmatrix}0\\0\end{pmatrix}\lambda_{112} + (3 \quad 2)\begin{pmatrix}0\\2\end{pmatrix}\lambda_{134} = 4\lambda_{134}.$$

Constraints:

$$(1 \quad 1)\begin{pmatrix}0\\0\end{pmatrix}\lambda_{112} + (1 \quad 1)\begin{pmatrix}0\\2\end{pmatrix}\lambda_{134} \leq 7.$$

$$(2 \quad 1)\begin{pmatrix}0\\0\end{pmatrix}\lambda_{112} + (1 \quad 3)\begin{pmatrix}0\\2\end{pmatrix}\lambda_{134} \leq 17.$$

First iteration master problem

$$\begin{aligned} \text{maximize} \quad & 4\lambda_{134} \\ \text{subject to:} \quad & 2\lambda_{134} \leq 7 \\ & 6\lambda_{134} \leq 17 \\ & \lambda_{112} = 1 \\ & \lambda_{134} = 1 \\ & \lambda_{112}, \lambda_{134} \geq 0. \end{aligned} \tag{44}$$

$\lambda^*_{112} = 1, \lambda^*_{134} = 1, \bar{z} = 4, \pi^*_1 = \pi^*_2 = \nu^*_1 = 0, \nu^*_2 = 4.$

Example - First iteration auxiliary problem 1

$$\begin{aligned} \text{maximize} \quad & (3-\pi_1-2\pi_2)x_1+(4-\pi_1-\pi_2)x_2-\nu_1 \\ \text{subject to:} \quad & x_1+2x_2 \leq 6 \\ & 4x_1+3x_2 \leq 12 \\ & x_1, x_2 \geq 0. \end{aligned}$$

Remember that $\pi_1 = \pi_2 = \nu_1 = 0, \nu_2 = 4$.

$$\begin{aligned} \text{maximize} \quad & 3x_1+4x_2 \\ \text{subject to:} \quad & x_1+2x_2 \leq 6 \\ & 4x_1+3x_2 \leq 12 \\ & x_1, x_2 \geq 0. \end{aligned}$$

$x_1^* = 1.2, x_2^* = 2.4, OF = 13.2$.

Example - First iteration auxiliary problem 2

$$\begin{aligned} \text{maximize} \quad & (3 - \pi_1 - \pi_2)x_3 + (2 - \pi_1 - 3\pi_2)x_4 - \nu_2 \\ \text{subject to:} \quad & 2x_3 + x_4 \leq 8 \\ & x_3 + 5x_4 \leq 10 \\ & x_3, x_4 \geq 0. \end{aligned}$$

Remember that $\pi_1 = \pi_2 = \nu_1 = 0, \nu_2 = 4$.

$$\begin{aligned} \text{maximize} \quad & 3x_3 + 2x_4 - 4 \\ \text{subject to:} \quad & 2x_3 + x_4 \leq 8 \\ & x_3 + 5x_4 \leq 10 \\ & x_3, x_4 \geq 0. \end{aligned}$$

$x_3^* = 3.33333..., x_4^* = 1.33333..., OF = 8.666667....$

Example - second iteration master problem

Solving auxiliary problems 1 and 2, two columns were generated:
column 1 = (1.2 2.4) and column 2 = (3.33333 1.33333).
Calculating the coefficients of the master problem using these two columns.
Objective function:

$$(3 \;\; 4)\begin{pmatrix}1.2\\2.4\end{pmatrix}\lambda_{212} + (3 \;\; 2)\begin{pmatrix}3.33333\\1.33333\end{pmatrix}\lambda_{234} = 13.2\lambda_{212} + 12.666665\lambda_{234}.$$

Constraints:

$$(1 \;\; 1)\begin{pmatrix}1.2\\2.4\end{pmatrix}\lambda_{212} + (1 \;\; 1)\begin{pmatrix}3.33333\\1.33333\end{pmatrix}\lambda_{234} \leq 7.$$

$$(2 \;\; 1)\begin{pmatrix}1.2\\2.4\end{pmatrix}\lambda_{212} + (1 \;\; 3)\begin{pmatrix}3.33333\\1.33333\end{pmatrix}\lambda_{234} \leq 17.$$

Second iteration master problem

$$
\begin{aligned}
\text{maximize} \quad & 4\lambda_{134} + 13.2\lambda_{212} + 12.666665\lambda_{234} \\
\text{subject to:} \quad & 2\lambda_{134} + 3.6\lambda_{212} + 4.66666\lambda_{234} \leq 7 \\
& 6\lambda_{134} + 4.8\lambda_{212} + 7.333332\lambda_{234} \leq 17 \\
& \lambda_{112} + \lambda_{212} = 1 \\
& \lambda_{134} + \lambda_{234} = 1
\end{aligned}
$$

$\lambda^*_{112} = 0,\ \lambda^*_{134} = 0.47499,\ \lambda^*_{212} = 1,\ \lambda^*_{234} = 0.525,\ \bar{z} = 21.75,$

$\pi^*_1 = 2.71428,\ \pi^*_2 = \nu^*_2 = 0, \nu^*_1 = 3.42855.$

Second iteration auxiliary problem 1

$$\begin{aligned} \text{maximize} \quad & (3 - \pi_1 - 2\pi_2)x_1 + (4 - \pi_1 - \pi_2)x_2 - \nu_1 \\ \text{subject to:} \quad & x_1 + 2x_2 \leq 6 \\ & 4x_1 + 3x_2 \leq 12 \\ & x_1, x_2 \geq 0. \end{aligned}$$

Remember that $\pi_1^* = 2.71428,\ \pi_2^* = \nu_2^* = 0, \nu_1^* = 3.42855.$

$$\begin{aligned} \text{maximize} \quad & 0.28572x_1 + 1.28572x_2 - 3.42855 \\ \text{subject to:} \quad & x_1 + 2x_2 \leq 6 \\ & 4x_1 + 3x_2 \leq 12 \\ & x_1, x_2 \geq 0. \end{aligned}$$

$x_1^* = 0\ , x_2^* = 3\ , OF = 0.42861.$

Example - Second iteration auxiliary problem 2

$$\begin{aligned} \text{maximize} \quad & (3 - \pi_1 - \pi_2)x_3 + (2 - \pi_1 - 3\pi_2)x_4 - \nu_2 \\ \text{subject to:} \quad & 2x_3 + x_4 \leq 8 \\ & x_3 + 5x_4 \leq 10 \\ & x_3, x_4 \geq 0. \end{aligned}$$

Remember that $\pi_1^* = 2.71428,\ \pi_2^* = \nu_2^* = 0, \nu_1^* = 3.42855.$

$$\begin{aligned} \text{maximize} \quad & 0.28572x_3 - 0.71428x_4 \\ \text{subject to:} \quad & 2x_3 + x_4 \leq 8 \\ & x_3 + 5x_4 \leq 10 \\ & x_3, x_4 \geq 0. \end{aligned}$$

$x_3^* = 4,\ x_4^* = 0,\ OF = 1.14288$

Example - third iteration master problem

Solving auxiliary problems 1 and 2, two columns were generated:
column 1 = (0 3) and column 2 = (4 0).
Calculating the coefficients of the master problem using these two columns.
Objective function:

$$(3 \quad 4)\begin{pmatrix}0\\3\end{pmatrix}\lambda_{312} + (3 \quad 2)\begin{pmatrix}4\\0\end{pmatrix}\lambda_{334} = 12\lambda_{312} + 12\lambda_{334}.$$

Constraints:

$$(1 \quad 1)\begin{pmatrix}0\\3\end{pmatrix}\lambda_{312} + (1 \quad 1)\begin{pmatrix}4\\0\end{pmatrix}\lambda_{334} \leq 7.$$

$$(2 \quad 1)\begin{pmatrix}0\\3\end{pmatrix}\lambda_{312} + (1 \quad 3)\begin{pmatrix}4\\0\end{pmatrix}\lambda_{334} \leq 17.$$

Third iteration master problem

$$
\begin{aligned}
\text{maximize} \quad & 4\lambda_{134} + 13.2\lambda_{212} + 12.666665\lambda_{234} + 12\lambda_{312} + 12\lambda_{334} \\
\text{subject to:} \quad & 2\lambda_{134} + 3.6\lambda_{212} + 4.66666\lambda_{234} + 3\lambda_{312} + 4\lambda_{334} \leq 7 \\
& 6\lambda_{134} + 4.8\lambda_{212} + 7.333332\lambda_{234} + 3\lambda_{312} + 4\lambda_{334} \leq 17 \\
& \lambda_{112} + \lambda_{212} + \lambda_{312} = 1 \\
& \lambda_{134} + \lambda_{234} + \lambda_{334} = 1
\end{aligned}
$$

$\lambda^*_{112} = \lambda^*_{212} = \lambda^*_{134} = \lambda^*_{234} = 0,\ \lambda^*_{312} = \lambda^*_{334} = 1,\ \bar{z} = 24,$

$\pi^*_1 = 3,\ \pi^*_2 = \nu^*_2 = 0, \nu^*_1 = 3.$

Third iteration auxiliary problem 1

$$\begin{aligned} \text{maximize} \quad & (3 - \pi_1 - 2\pi_2)x_1 + (4 - \pi_1 - \pi_2)x_2 - \nu_1 \\ \text{subject to:} \quad & x_1 + 2x_2 \leq 6 \\ & 4x_1 + 3x_2 \leq 12 \\ & x_1, x_2 \geq 0. \end{aligned}$$

Remember that $\pi_1^* = 3,\ \pi_2^* = \nu_2^* = 0, \nu_1^* = 3.$

$$\begin{aligned} \text{maximize} \quad & x_2 - 3 \\ \text{subject to:} \quad & x_1 + 2x_2 \leq 6 \\ & 4x_1 + 3x_2 \leq 12 \\ & x_1, x_2 \geq 0. \end{aligned}$$

$x_1^* = 0\ , x_2^* = 3\ , OF = 0.$

Example - Third iteration auxiliary problem 2

$$\begin{aligned} \text{maximize} \quad & (3-\pi_1-\pi_2)x_3+(2-\pi_1-3\pi_2)x_4-\nu_2 \\ \text{subject to:} \quad & 2x_3+x_4 \leq 8 \\ & x_3+5x_4 \leq 10 \\ & x_3, x_4 \geq 0. \end{aligned}$$

Remember that $\pi_1^* = 2.71428,\ \pi_2^* = \nu_2^* = 0, \nu_1^* = 3.42855.$

$$\begin{aligned} \text{maximize} \quad & -x_4 \\ \text{subject to:} \quad & 2x_3+x_4 \leq 8 \\ & x_3+5x_4 \leq 10 \\ & x_3, x_4 \geq 0. \end{aligned}$$

$x_3^* = x_4^* = 0,\ OF = 0$

Example

- Neither the auxiliary problem 1 nor the auxiliary problem 2 produced columns which would increase the value of the objective function of the master problem, therefore the current solution is optimal;
- Optimal solution: $x_1^* = 0,\ x_2^* = 3,\ x_3^* = 4,\ x_4^* = 0,\ \lambda_{312} = \lambda_{334} = 1$;
- We can see that the optimal solution is one vertex of the feasible region of the subproblems and not a vertex in the intersection with the master problem, because in this case, it would be a linear combination of vertices of the subproblems.

Exercise

You have 30 minutes to complete exercise 1.

Column Generation - important observations

- The solution of the Auxiliary Problem (AP) will always be a vertex of its feasible region;
- the optimum value of (AP) is be given by $[(c^\top - \pi A_k)v_j - \nu]$;
- If we call $\bar{z}$ the optimal value of the objective function of RMP at each iteration OF the optimal value of the auxiliary problem (AP) at each iteration, we have:
  - $\bar{z}$ gives a lower bound for RMP at each iteration;
  - $\bar{z} + OF$ gives an upper bound for RMP at each iteration;
- So, at each iteration we have $\bar{z} \leq \text{RMP value} \leq \bar{z} + OF$
- The problem is: there is no guarantee that the upper bound of iteration j is greater than the upper bound calculated at iteration $j + 1$, which means that the upper bound doesn't have a monotone behaviour throughout the iterations.

Integer programming problems

- Both Dantzig-Wolfe decomposition and column generation are being used nowadays to solve integer programming problems (IP);

$$\begin{aligned} (IP): \text{maximize} \quad & z = c^\top x \\ \text{subject to:} \quad & Ax \leq b \\ & x \geq 0 \text{ and integer.} \end{aligned}$$

- In general, integer programming problems are solved relaxing the integer constraint, which means solving the its linear relaxation, and then applying a Branch and Bound technique;
- In column generation we relax the Master problem and look for an integer column at the Auxiliary problem;
- Before starting column generation for integer programming problems, we are going to revise some important aspects of Integer programming.

Integer Programming - revision

Definition

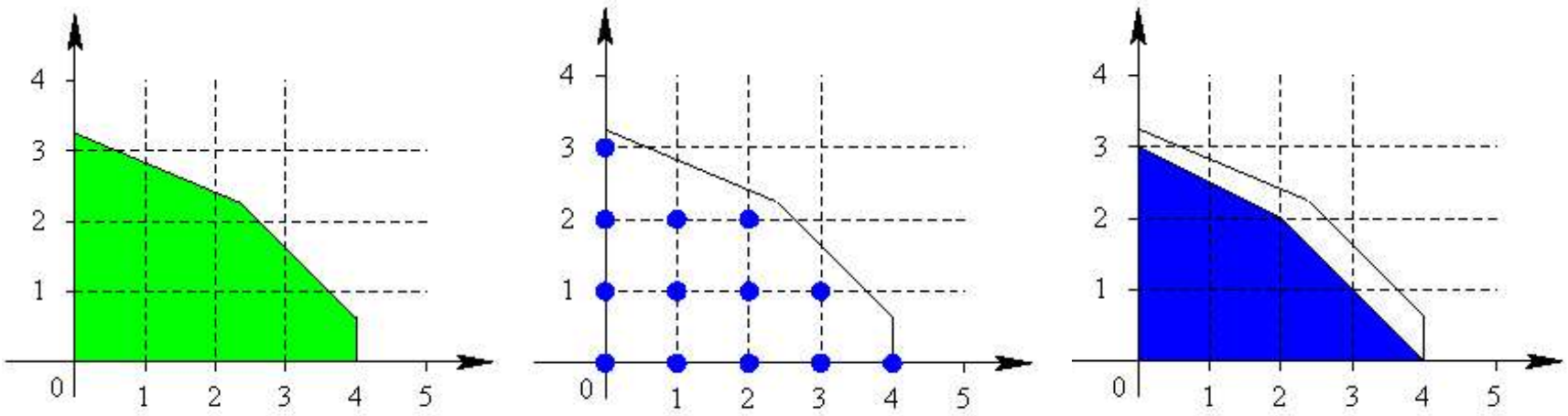
A subset of $\mathbb{R}^n$ described by a finite set of linear constraints $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ is a polyhedron.

Definition

A polyhedron $P \subseteq \mathbb{R}^{n+p}$ is a formulation for a set $X \subseteq \mathbb{Z}^n \times \mathbb{R}^p$ if and only if $X = P \cap (\mathbb{Z}^n \times \mathbb{R}^p)$.

Integer Programming - revision

- If we had the convex hull of every integer problem, it would be easy to solve them;



Integer Programming - revision

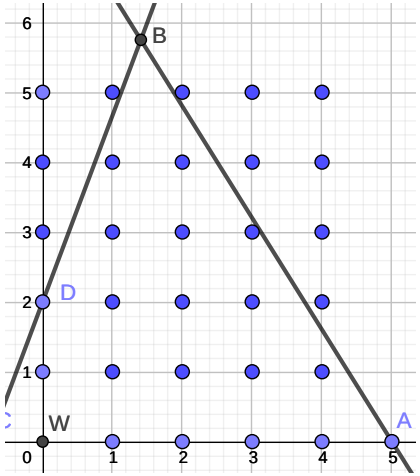
- As we don't have this information, we work with the relaxed problem and then look for the integer solution using Branch and Bound;
- We already know that rounding the solution found solving the linear relaxation isn't the right thing to do.

Integer Programming - revision

As we can see at this example of [6]:

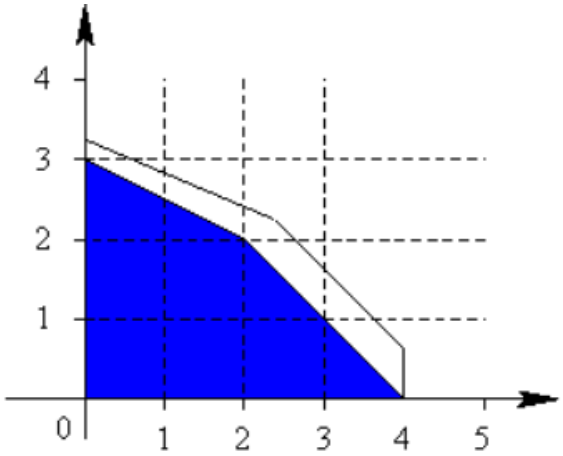
$$
\begin{aligned}
(IP): \text{maximize} \quad & z = x_1 + 0.64x_2 \\
\text{subject to:} \quad & 50x_1 + 31x_2 \leq 250 \\
& 3x_1 - 2x_2 \geq -4 \qquad\qquad x_1, x_2 \geq 0 \text{ and integer.}
\end{aligned}
$$

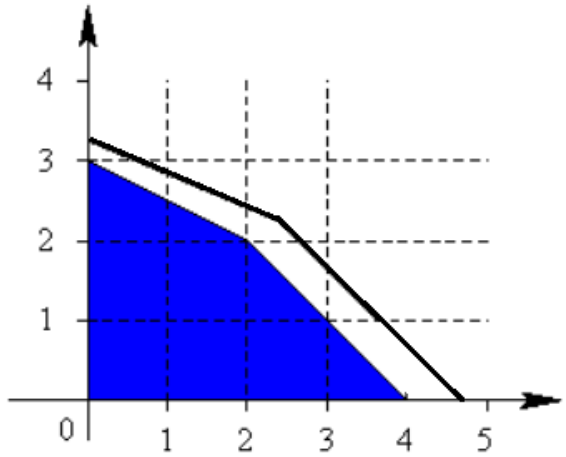
The solution of the relaxed problem is at $B = (\frac{376}{193}, \frac{950}{193})$ and the integer solution is at $A = (5, 0)$.



Integer Programming - revision

- A good formulation is an important step in the direction of successfully solve an integer problem.
- We can see at the figure below a blue region. It is the convex hull of the integer points of a generic problem.
- The polytope formed with a thin line is a representation of a better formulation comparing to the polytope formed with a thick line.





Integer Programming - revision

- Better formulations for the relaxed problems have feasible regions "nearest" to the convex hull of its integer points;
- "Tighest" feasible regions of relaxed problems produces smaller gaps, which means that the gap between the relaxed solution and the integer solution of a good formulation is smaller than in a not very good formulation.
- The tree created by Branch and Bound technique is smaller when the gap between the relaxed solution and the integer solution is small.

Integer Programming - revision

One example of study of better formulation is the subtour elimination constraint for the Minimum Spanning Tree formulation.

- Let $G = (V, E)$ be an undirected graph with vertex set V where $|V| = n$ and edge set E, with $|E| = m$.
- Remember that an edge in an undirected graph is an unordered pair $e = \{i, j\},\ i, j \in V,\quad i \neq j$.
- For every $e \in E$ there is a cost c_e.
- Let $E' \subseteq E$. If $G' = (V, E')$ is a tree and all vertices are reached by at least one edge, so G' is called a spanning tree.
- The cost of the tree is the sum of the costs of its edges.
- The Minimum Spanning Tree (MST) problem consists of finding a spanning tree of minimum cost.

Integer programming - revision

Minimum Spanning Tree formulation

- Let $x_{ij} = 1$ if the edge ij is at the tree T;
- We need one constraint to ensure that there are $n-1$ edges in T;

$$\sum_{ij \in E} x_{ij} = n - 1$$

- We need another constraint to guarantee that there is no cycles in T. Subtour elimination constraint. Any subset of k vertices must have at most $k-1$ edges contained in that subset.

$$\sum_{ij \in E : i \in S, j \in S} x_{ij} \leq |S| - 1 \quad \forall S \subseteq V,\ S \neq \emptyset$$

MST formulation 1 - Subtour elimination formulation

$$\begin{aligned}
\text{minimize} \quad & \sum_{ij \in E} c_{ij} x_{ij} \\
\text{subject to:} \quad & \sum_{ij \in E} x_{ij} = n - 1 \\
& \sum_{ij \in E : i \in S, j \in S} x_{ij} \leq |S| - 1 \quad \forall S \subseteq V,\ S \neq \emptyset \\
& x_{ij} \in \{0, 1\} \quad \forall ij \in E.
\end{aligned}$$

Integer Programming - revision

- The definition of tree used by MST formulation 1 is a subgraph containing $n-1$ edges and no cycles;
- Using the definition of a tree as a connected graph containing $n-1$ edges, we can have another formulation;
- In order to guarantee that the tree is connected, given a subset S of V, we define the cutset $\delta(S)$

$$\delta(S) = \{ij \in E \mid i \in S,\ j \notin S\}$$

- We can see that $\delta(i)$ is the set of edges which incides on vertex i. The conectivity of the tree can be described by the constraints:

$$\sum_{ij \in \delta(S)} x_{ij} \geq 1, \qquad S \subseteq E,\ S \neq \emptyset$$

MST formulation 2 - Cutset formulation

$$\begin{aligned}
\text{minimize} \quad & \sum_{ij \in E} c_{ij} x_{ij} \\
\text{subject to:} \quad & \sum_{ij \in E} x_{ij} = n - 1 \\
& \sum_{ij \in \delta(S)} x_{ij} \geq 1, \qquad S \subseteq V,\ S \neq \emptyset \\
& x_{ij} \in \{0, 1\} \qquad \forall ij \in E.
\end{aligned}$$

Integer Programming - revision

- Formulations 1 and 2 have an exponential number of constraints;
- It can be proved that the subtour elimination formulation is stronger than the cutset formulation;
- I suggest reading [7].

Integer Programming - revision - Branch and Bound

- Branch and Bound uses the Simplex Method combined to a divide and conquer strategy;
- the divide and conquer strategy appears after the relaxed problem is solved;
- if the relaxed problem doesn't give an integer solution, which is the expected scenario, subproblems are generated with additional constraints on these variables.

Integer Programming - revision - Branch and Bound

- But what happens if all variables of the relaxed problem are continuous and we want them to be integer, we are going to subdivide each one of them in two?
- If we run a complete enumeration, the method becomes impracticable;
- Thus we make an implicit enumeration.

Integer Programming - revision - Branch and Bound

- Implicit enumeration is based on a pruning mechanism and the non-promising branches are not examined;
- We use the information about bounds of the current solution and this information helps us to prune several nodes.

Integer Programming - revision - Branch example

Integer Problem

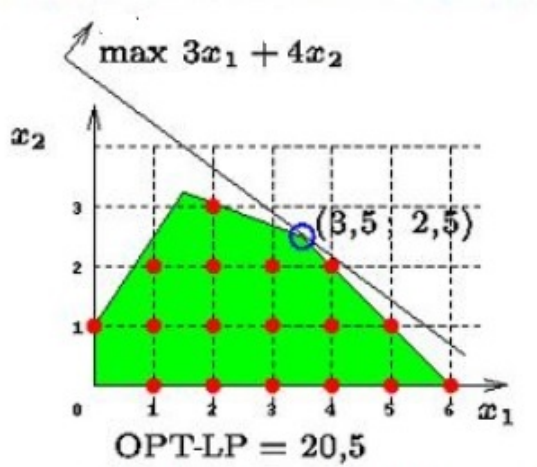
$$
\begin{aligned}
\text{maximize} \quad & 3x_1 + 4x_2 \\
\text{subject to:} \quad & -3x_1 + 2x_2 \leq 2 \\
& x_1 + 3x_2 \leq 11 \\
& x_1 + x_2 \leq 6 \\
& x_1, x_2 \geq 0 \text{ and integer.}
\end{aligned}
$$

Relaxed Problem

$$
\begin{aligned}
\text{maximize} \quad & 3x_1 + 4x_2 \\
\text{subject to:} \quad & -3x_1 + 2x_2 \leq 2 \\
& x_1 + 3x_2 \leq 11 \\
& x_1 + x_2 \leq 6 \\
& x_1, x_2 \geq 0
\end{aligned}
$$

Integer Programming - revision - Branch example

The figure below shows the graphical representation of the relaxed problem and its integer points.



We can see that the solution of the relaxed problem is (3.5, 2.5).

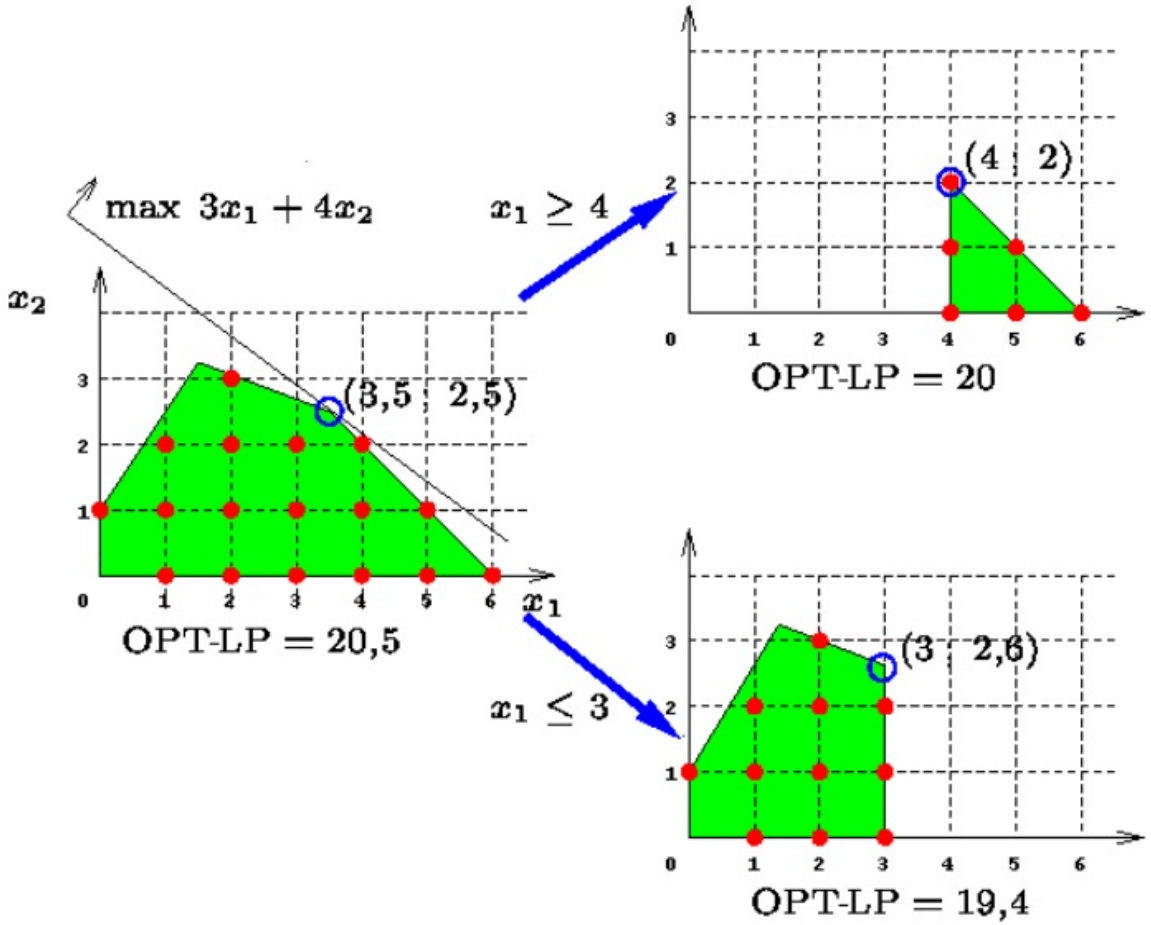
Integer Programming - revision - Branch example

Choosing variable x_1 to branch, we have:

$$
\begin{aligned}
\text{maximize} \quad & 3x_1 + 4x_2 \\
\text{subject to:} \quad & -3x_1 + 2x_2 \leq 2 \\
& x_1 + 3x_2 \leq 11 \\
& x_1 + x_2 \leq 6 \\
& x_1 \geq 4 \\
& x_1, x_2 \geq 0.
\end{aligned}
$$

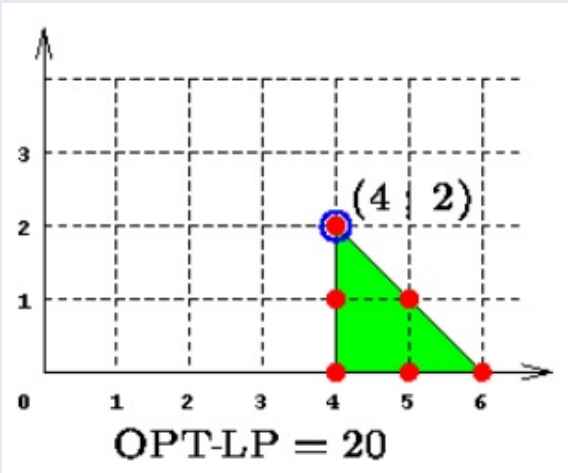
$$
\begin{aligned}
\text{maximize} \quad & 3x_1 + 4x_2 \\
\text{subject to:} \quad & -3x_1 + 2x_2 \leq 2 \\
& x_1 + 3x_2 \leq 11 \\
& x_1 + x_2 \leq 6 \\
& x_1 \leq 3 \\
& x_1, x_2 \geq 0
\end{aligned}
$$

Integer Programming - revision - Branch example

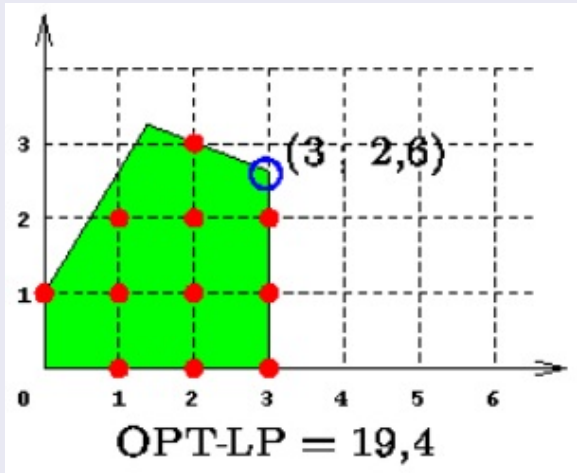


Integer Programming - revision - Branch example

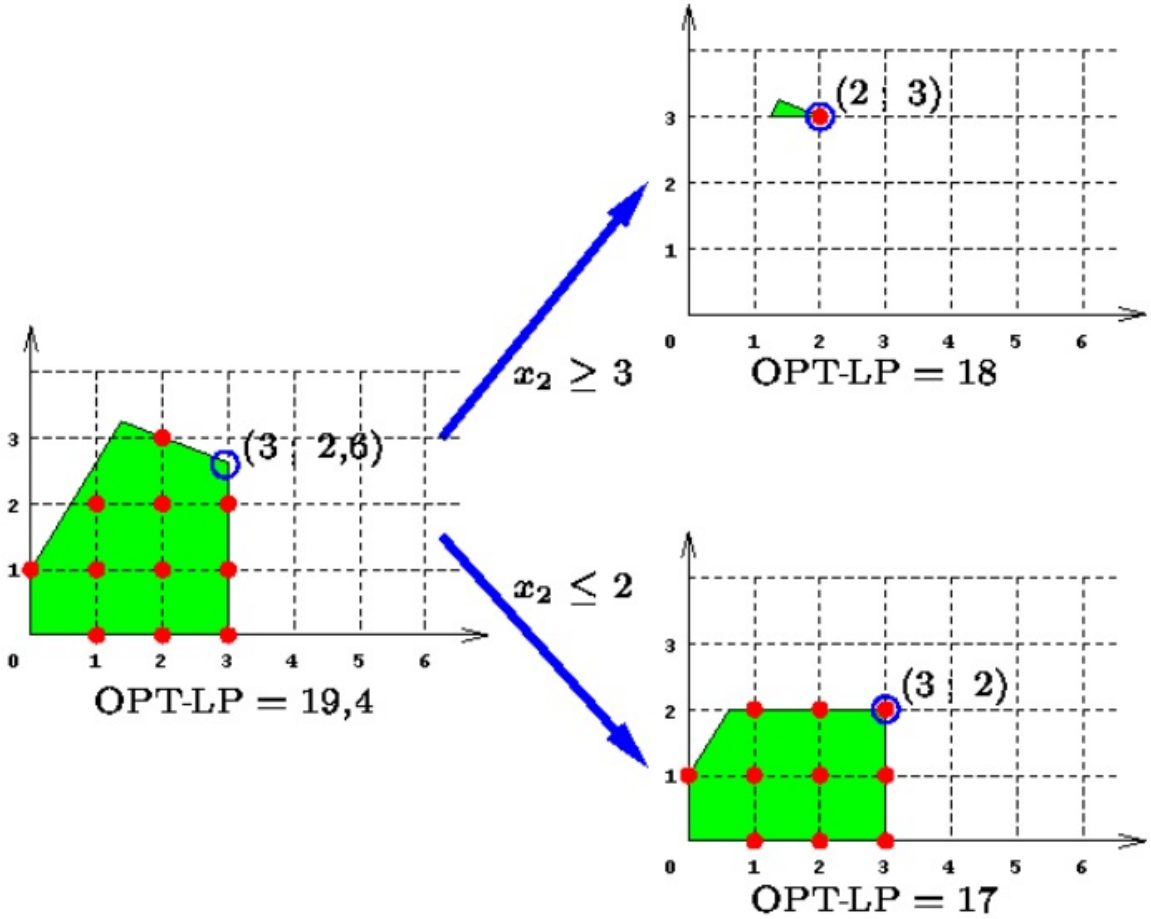
$$
\begin{aligned}
\text{maximize} \quad & 3x_1 + 4x_2 \\
\text{subject to:} \quad & -3x_1 + 2x_2 \leq 2 \\
& x_1 + 3x_2 \leq 11 \\
& x_1 + x_2 \leq 6 \\
& x_1 \geq 4 \\
& x_1, x_2 \geq 0.
\end{aligned}
$$



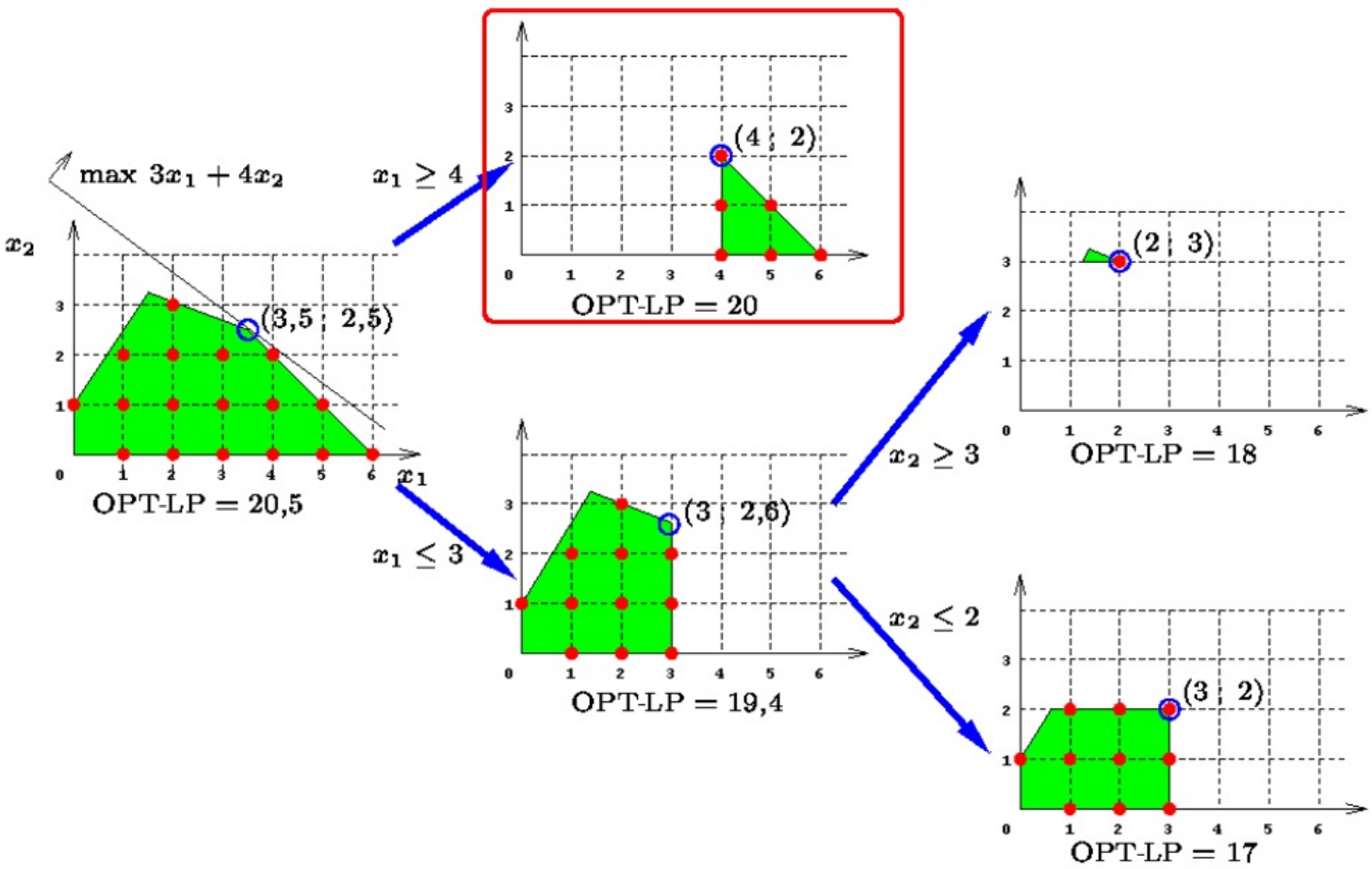
$$
\begin{aligned}
\text{maximize} \quad & 3x_1 + 4x_2 \\
\text{subject to:} \quad & -3x_1 + 2x_2 \leq 2 \\
& x_1 + 3x_2 \leq 11 \\
& x_1 + x_2 \leq 6 \\
& x_1 \leq 3 \\
& x_1, x_2 \geq 0
\end{aligned}
$$



Integer Programming - revision - Branch example



Integer Programming - revision - Branch example



Integer Programming - revision

- We only pruned one node in the example we saw, and this prune was made by optimality;
- It's important to observe that the tree can get very big;
- In Branch and Bound technique, we prune the nodes based on some aspects as bound, feasibility and optimality;
- Solving the relaxed maximization problem gives us an upper bound u on the objective value of the original problem;
- The value of the objective function of a feasible solution of a maximization problem gives a lower bound l on the objective value of the original problem.

Integer Programming - revision

- The **incumbent** is a feasible solution for the IP problem and it is the best solution so far in the B&B search.
- We say a node is **active** if it hasn't been pruned and if its (LP) has not been solved yet.
- We can prune the active node k if its optimal objective function value is lesser or equal the value of the objective function given by the **incumbent**.

Integer Programming - revision - Branch and Bound example

$$
\begin{aligned}
\text{maximize} \quad & 4x_1 - x_2 \\
\text{subject to:} \quad & 3x_1 - 2x_2 \leq 14 \\
& 2x_1 - 2x_2 \leq 3 \\
& x_2 \leq 3 \\
& x_1, x_2 \geq 0 \text{ and integer.}
\end{aligned}
$$

$$
\begin{aligned}
\text{maximize} \quad & 4x_1 - x_2 \\
\text{subject to:} \quad & 3x_1 - 2x_2 + x_3 = 14 \\
& 2x_1 - 2x_2 + x_4 = 3 \\
& x_2 + x_5 = 3 \\
& x_1, ..., x_5 \geq 0 \\
& x_1, ..., x_5 \text{ integer.}
\end{aligned}
$$

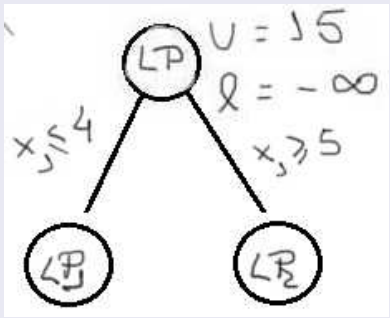
Integer Programming - revision - Branch and Bound example

Solve the relaxed problem

$$
\begin{aligned}
\text{maximize} \quad & 4x_1 - x_2 \\
\text{subject to:} \quad & 3x_1 - 2x_2 + x_3 = 14 \\
& 2x_1 - 2x_2 + x_4 = 3 \\
& x_2 + x_5 = 3 \\
& x_1, x_2, x_3, x_4, x_5 \geq 0
\end{aligned}
$$

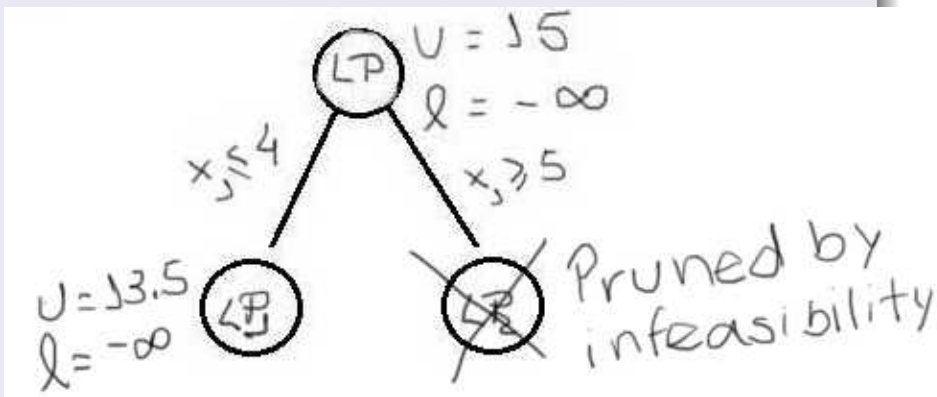
$x^* = (4.5, 3, 6.5, 0, 0)$, OF $= 15$.

Choosing x_1 to branch, we have:



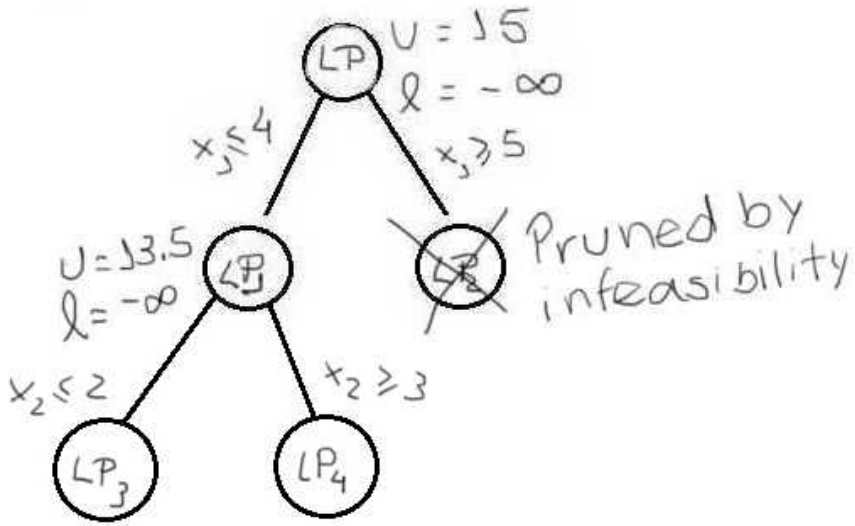
Integer Programming - revision - Branch and Bound example

- Solving LP_1 we obtain $x^* = (4, 2.5, 7, 4, 0)$ and $OF = 13.5$, thus $u = 13.5$;
- As the solution of LP_1 isn't feasible for the integer problem, our lower bound l continues $l = -\infty$;
- LP_2 is infeasible, so we prune this node by infeasibility.



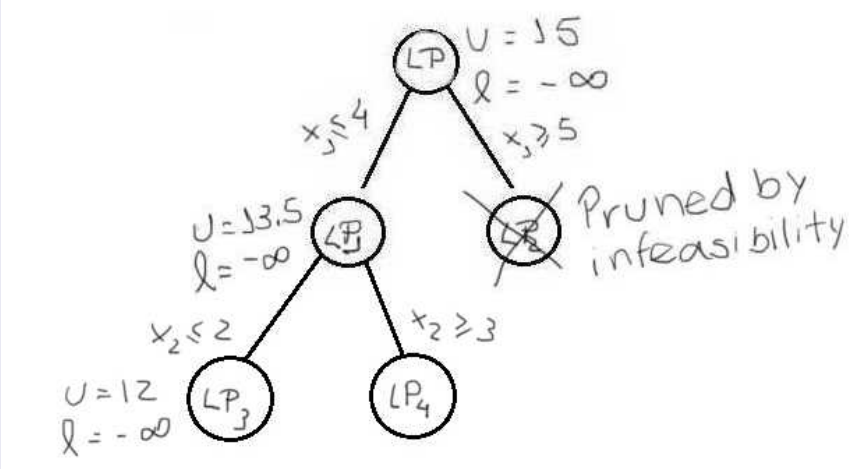
Integer Programming - revision - Branch and Bound example

The only fractionary variable is x_2, so we branch on it.

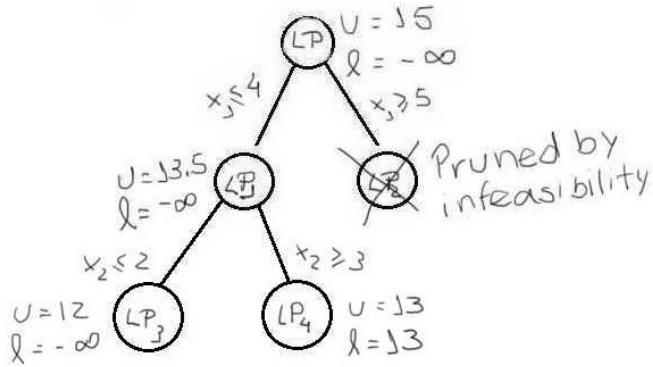


Integer Programming - revision - Branch and Bound example

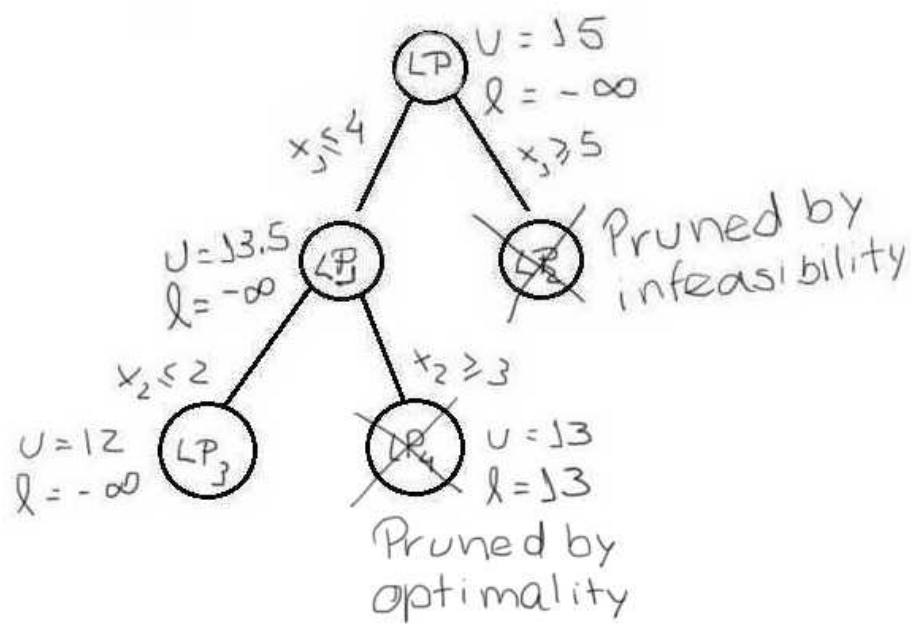
Solving LP_3 we have $x^* = (3.5, 2, 7.5, 1, 0)$ and $OF = 12$, which means $u = 12$.



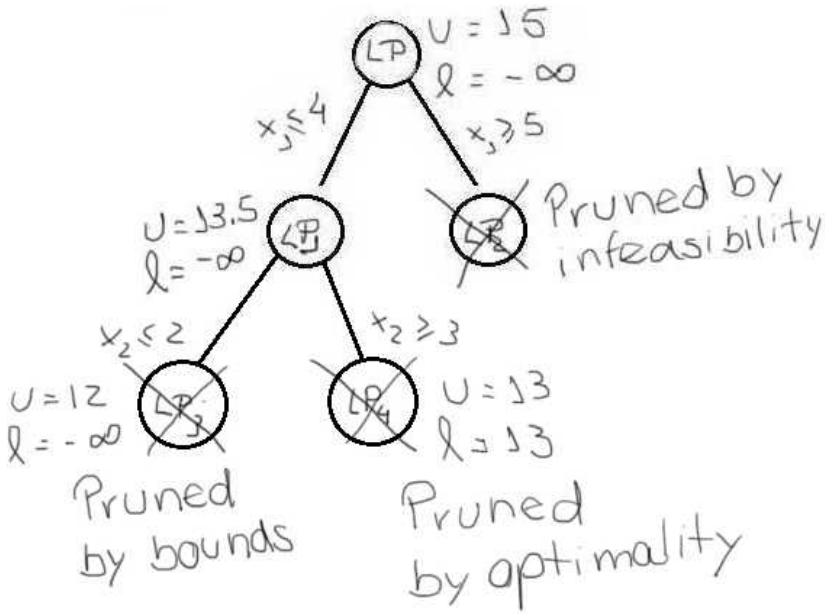
Solving LP_4 we have $x^* = (4, 3, 8, 1, 0)$, which means $u = 13$. and $l = 13$. As this solution is feasible for the integer problem, we have $l = 13$, so we reached the optimal solution in this node. This is the incumbent so far.



Integer Programming - revision - Branch and Bound example



Integer Programming - revision - Branch and Bound example



Bibliography I

[1] N. MACULAN e M. Fampa, *Otimização linear*. EdUnB, 2006, ISBN: 9788523009274. endereço: https://books.google.com.br/books?id=s02VPgAACAAJ.

[2] L. Simonetti, L. Patuzzi, A. F. U. S. Macambira e P. H. González, "Geração de Colunas em Programação Inteira," em *Tópicos em Programação Inteira*, A. F. U. S. Macambira, L. Simonetti, R. F. Rodrigues e N. Maculan, ed., In Press, UFRJ - Rio de Janeiro - Brasil: Editora da UFRJ, 2021, cap. 5, pp. 81–90.

[3] G. Dantzig e P. Wolfe, "The decomposition algorithm for linear programming," *Operations Research*, v. 8, pp. 101–111, 1960.

[4] P. Gilmore e R. Gomory, "A linear programming approach to the cutting stock problem," *Operations Research*, v. 9, pp. 849–859, 1961.

[5] ——,"A linear programming approach to the cutting stock problem – Part 2," *Operations Research*, v. 11, pp. 863–888, 1963.

[6] L. Wolsey, *Integer Programming*, sér. Wiley Series in Discrete Mathematics and Optimization. Wiley, 1998, ISBN: 9780471283669. endereço: https://books.google.com.br/books?id=x7RvQgAACAAJ.

Bibliography II

[7] C. Feremans, M. Labbé e G. Laporte, "A Comparative Analysis of Several Formulations for the Generalized Minimum Spanning Tree Problem," *Networks*, v. 39, pp. 29–34, jan. de 2002. DOI: 10.1002/net.10009.