ELEN0445-1 - Microgrids

Introduction to mathematical programming (v2)

Based on slides from Pr. Quentin Louveaux

Bertrand Cornélusse

ULiège - Institut Montefiore

2021

Lecture overview

- The goal of this lecture is to introduce the concept of mathematical programming, and in particular of linear and mixed integer programming.
- We also quickly review solution methods for these problems.
- These tools can then be used for real-time dispatch, operational planning, or sizing of a microgrid.

Outline

Mathematical programming

Linear programming

Integer and Mixed-Integer programming

Modeling techniques

Cutting planes

Branch and bound

Mathematical programming

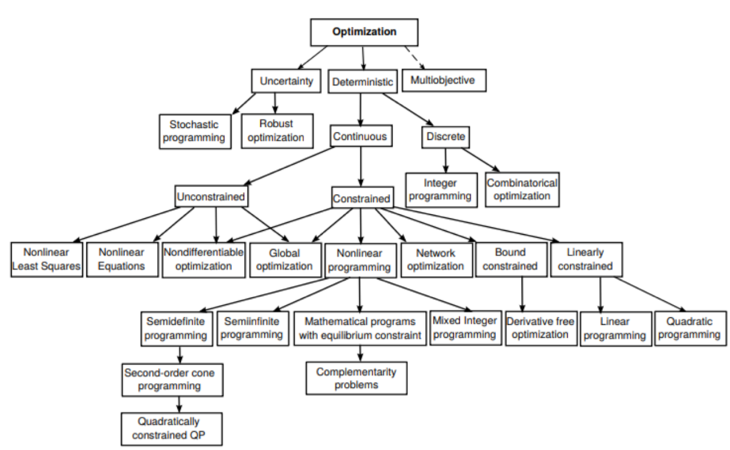
Mathematical programming is a field of applied mathematics that deals with the solution of optimization problems.

More precisely, it provides a framework and solution methods for computing the decisions of an optimization problem, given an objective function to minimize or maximize, and (optionally) constraints on the decisions variables.

Mathematical programming relies on a model of the problem to solve.

There is a great variety of mathematical programming problem types, depending on the characteristics of the objective function and of the constraints, and of the restrictions that apply to variables.

Categories of mathematical programs



https://neos-guide.org/content/optimization-taxonomy

Categories of mathematical programs

General mathematical program

A general mathematical program can be stated as follows:

$$\begin{aligned} \min\ & f(x) \\ s.t.\ & g(x) \leq 0 \\ & Ax = 0 \\ & x \in X \end{aligned}$$

It is very hard to solve, especially when

- objective and constraints are non-linear or even worse non-convex
- variables are discrete

Linear program

$$\begin{aligned} \min\ & c^T x \\ s.t.\ & Ax = b \\ & x \in \mathbb{R}^n_+ \end{aligned}$$

Easy to solve even for large problems.

Mixed-Integer Linear program

$$\begin{aligned} \min\ & c^T x \\ s.t.\ & Ax = b \\ & x \in \mathbb{R}^{n_1}_+ * \mathbb{Z}^{n_2}_+ \end{aligned}$$

Hard problem, but feasible for moderately sized instances.

Outline

Mathematical programming

Linear programming

Integer and Mixed-Integer programming

Modeling techniques

Cutting planes

Branch and bound

Linear programming

If the objective is linear and the constraints are linear, we talk about linear programming (LP) or linear optimization.

LP in standard form

$$\begin{aligned} \min\ & c^T x \\ \text{s.t. } & Ax = b \\ & x \in \mathbb{R}^n_+ \end{aligned}$$

Definition

A polyhedron is a set $\{x \in \mathbb{R}^n | Ax \geq b\}$

A set of the form $Ax \leq b$ is also a polyhedron.
A set $\{x \in \mathbb{R}^n | Ax = b, x \geq 0\}$ is a polyhedron in standard form.

Graphic representation

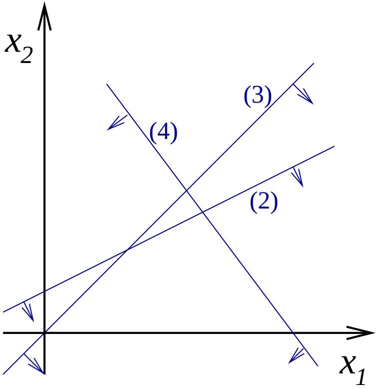
We can represent a problem in two dimensions graphically.

Example:

$$
\begin{aligned}
\max\ x_1 + 2x_2 & \qquad (1)\\
-x_1 + 2x_2 \leq 1 & \qquad (2)\\
-x_1 + x_2 \leq 0 & \qquad (3)\\
4x_1 + 3x_2 \leq 12 & \qquad (4)\\
x_1, \quad x_2 \geq 0 & \qquad (5)
\end{aligned}
$$

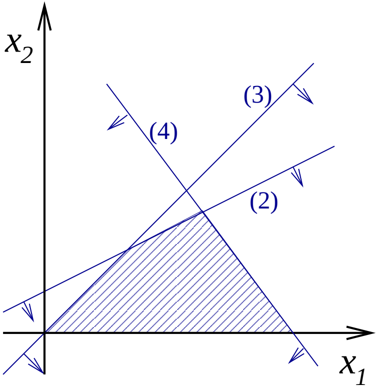
Graphic representation

$$\max \; x_1 + 2x_2 \tag{1}$$
$$-x_1 + 2x_2 \leq 1 \tag{2}$$
$$-x_1 + x_2 \leq 0 \tag{3}$$
$$4x_1 + 3x_2 \leq 12 \tag{4}$$
$$x_1, \quad x_2 \geq 0 \tag{5}$$



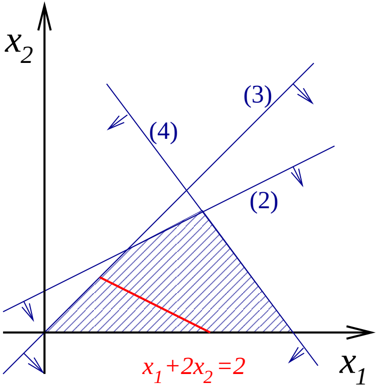
Graphic representation

$$\max\ x_1 + 2x_2 \tag{1}$$
$$-x_1 + 2x_2 \leq 1 \tag{2}$$
$$-x_1 + x_2 \leq 0 \tag{3}$$
$$4x_1 + 3x_2 \leq 12 \tag{4}$$
$$x_1,\quad x_2 \geq 0 \tag{5}$$



Graphic representation

$$\max\ x_1 + 2x_2 \tag{1}$$
$$-x_1 + 2x_2 \leq 1 \tag{2}$$
$$-x_1 + x_2 \leq 0 \tag{3}$$
$$4x_1 + 3x_2 \leq 12 \tag{4}$$
$$x_1,\quad x_2 \geq 0 \tag{5}$$



Graphic representation

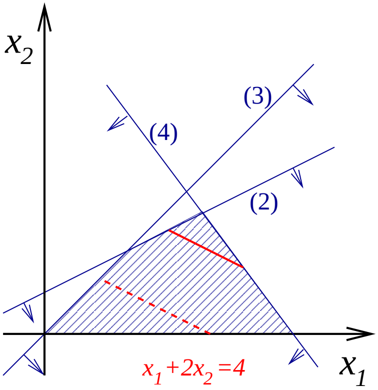
$$\max \; x_1 + 2x_2 \quad (1)$$

$$-x_1 + 2x_2 \leq 1 \quad (2)$$

$$-x_1 + x_2 \leq 0 \quad (3)$$

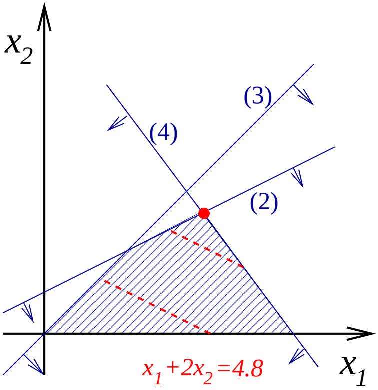
$$4x_1 + 3x_2 \leq 12 \quad (4)$$

$$x_1, \quad x_2 \geq 0 \quad (5)$$



Graphic representation

$$\max \; x_1 + 2x_2 \tag{1}$$
$$-x_1 + 2x_2 \leq 1 \tag{2}$$
$$-x_1 + x_2 \leq 0 \tag{3}$$
$$4x_1 + 3x_2 \leq 12 \tag{4}$$
$$x_1, \quad x_2 \geq 0 \tag{5}$$



Extreme points and vertices

Definition

Let P be a polyhedron. A point $x \in P$ is an extreme point of P if there do not exist two points $y, z \in P$ such that x is a convex combination of y and z.

Definition

Let P be a polyhedron. A point $x \in P$ is a vertex of P if there exists $c \in \mathbb{R}^n$ such that $c^T x < c^T y$ for all $y \in P$ and $y \neq x$.

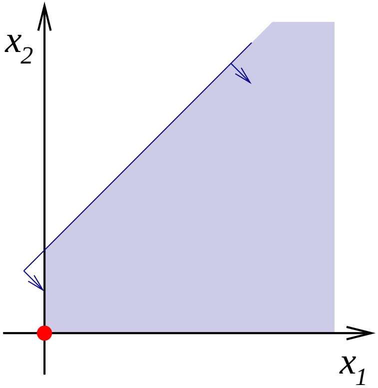
Degenerate cases

In the example we had a unique solution at a vertex of the polyhedron.
Some degenerate cases can lead to different solutions.

Degenerate cases

In the example we had a unique solution at a vertex of the polyhedron.
Some degenerate cases can lead to different solutions.

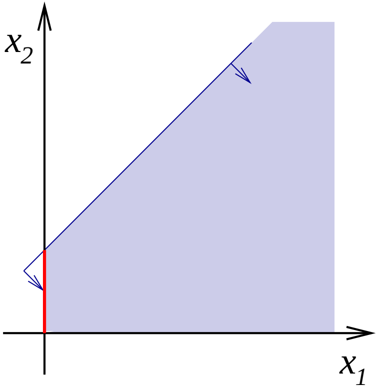
$$\begin{aligned} &\min\ x_1 + x_2 \\ &\text{s.t.}\ \ -x_1 + x_2 \leq 1 \\ &\qquad x_1, x_2 \geq 0 \end{aligned}$$



Degenerate cases

In the example we had a unique solution at a vertex of the polyhedron.
Some degenerate cases can lead to different solutions.

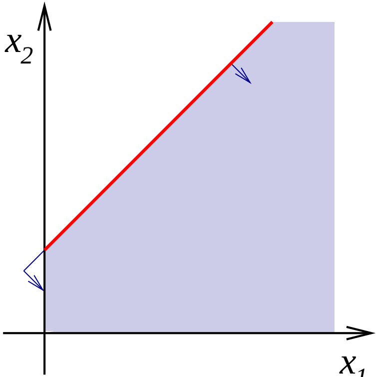
$$\min x_1$$
$$\text{s.t.} \quad -x_1 + x_2 \leq 1$$
$$x_1, x_2 \geq 0$$



Degenerate cases

In the example we had a unique solution at a vertex of the polyhedron.
Some degenerate cases can lead to different solutions.

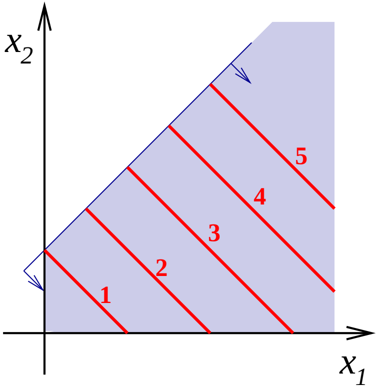
$$\begin{aligned} \max \quad & -x_1 + x_2 \\ \text{s.t.} \quad & -x_1 + x_2 \leq 1 \\ & x_1, x_2 \geq 0 \end{aligned}$$



Degenerate cases

In the example we had a unique solution at a vertex of the polyhedron.
Some degenerate cases can lead to different solutions.

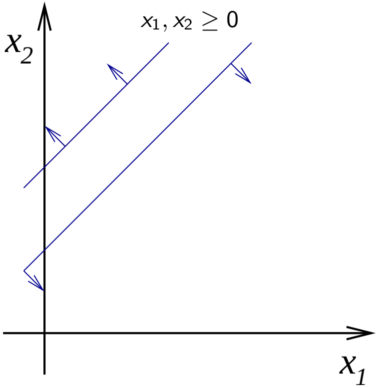
$$\max x_1 + x_2$$
$$\text{s.t.} \quad -x_1 + x_2 \leq 1$$
$$x_1, x_2 \geq 0$$



Degenerate cases

In the example we had a unique solution at a vertex of the polyhedron.
Some degenerate cases can lead to different solutions.

$$
\begin{aligned}
\max\ & x_1 + 2x_2 \\
\text{s.t.}\ & -x_1 + x_2 \leq 1 \\
& -x_1 + x_2 \geq 2 \\
& x_1, x_2 \geq 0
\end{aligned}
$$



Bases of a polyhedron

We subdivide the equalities and inequalities into three categories:

$$
\begin{aligned}
a_i^T x &\geq b_i \qquad & i \in M_{\geq} \\
a_i^T x &\leq b_i \qquad & i \in M_{\leq} \\
a_i^T x &= b_i \qquad & i \in M_{=}
\end{aligned}
$$

Definition

Let $\bar{x}$ be a point satisfying $a_i^T \bar{x} = b_i$ for some $i \in M_{\geq}, M_{\leq}$ or $M_{=}$. The constraint i is said to be active or tight.

Bases of a polyhedron

Definition

Let P be a polyhedron and let $\bar{x} \in \mathbb{R}^n$.

(a) $\bar{x}$ is a basic solution if

- all equalities ($i \in M_=$) are active
- among the active constraints, there are n linearly independent

(b) if $\bar{x}$ is a basic solution that satisfies all constraints, then $\bar{x}$ is a feasible basic solution.

Theorem

Let P be a polyhedron and let $\bar{x} \in P$. The three following statements are equivalent.

(i) $\bar{x}$ is a vertex

(ii) $\bar{x}$ is an extreme point

(iii) $\bar{x}$ is a basic feasible solution

Linear programming algorithms

There are two main types of algorithms used in practice.

Simplex methods

- moves from one vertex (extreme point) of the feasible domain to another until objective stops decreasing
- very efficient in practice but can be exponential on some special problems
- can keep information of one solution to quickly compute a solution to a perturbed problem (useful in a B&B setting), dual simplex, ...

Interior point methods

- iteratively penalizes the objective with a function of constraints, to force successive points to lie within the feasible domain
- polynomial time, very efficient especially for large sparse systems
- but no extremal solution hence crossover required in a B&B setting

More advanced topics

- Duality
- Shadow prices
- Complementary slackness
- Sensitivity analyses
- ...

Outline

Modeling a discrete problem

Consider the problem

$$
\begin{aligned}
\min\ & c(x) \\
\text{s.t. } & f(x) \leq b \\
& g(x) = 0 \\
& x \in X.
\end{aligned}
$$

When

- c, f, g are linear
- $X = \mathbb{Z}^n_+$

This is called Integer (Linear) Programming (IP).
Remarks:

- **Mixed** Integer Programming (MIP) when some variables have a continuous domain.
- If c is nonlinear, very little has been done (except the quadratic case)
- If f or g is nonlinear: even less

Why is that so complicated?

- After all, there is a finite number of solutions
- In particular, $n!$ possible permutations
- Imagine we can check 10^{12} possibilities per second
  That is already a pretty amazing machine!

10!	0 sec
20!	28 days
30!	8400 billion years
40!	5 quadrillions times the age of the Earth...

- Let us not dare to continue...

Example: Uncapacitated Lot Sizing (ULS)

You are producing bikes and you know in advance the demand d_t for T time steps ahead. Producing at time t induces a fixed cost f_t, and the variable cost per bike produced is c_t. There is no storage cost. Formulate the MIP that allows you to compute the production plan that minimizes the total production cost to satisfy the demand.

Example: Uncapacitated Lot Sizing (ULS)

You are producing bikes and you know in advance the demand d_t for T time steps ahead. Producing at time t induces a fixed cost f_t, and the variable cost per bike produced is c_t. There is no storage cost. Formulate the MIP that allows you to compute the production plan that minimizes the total production cost to satisfy the demand.

Formulation as a MIP:

$$\min_{x,y} \sum_{t=1}^{T} f_t x_t + \sum_{t=1}^{T} c_t y_t$$

$$\textbf{s.t.} \sum_{u=1}^{t} y_u \geq \sum_{u=1}^{t} d_u, \ \forall t \in \mathcal{T}$$

$$y_t \leq \left(\sum_{u=t}^{T} d_u \right) x_t, \ \forall t \in \mathcal{T}$$

$$y_t \geq 0, \ \forall t \in \mathcal{T}$$

$$x_t \in \{0, 1\}, \ \forall t \in \mathcal{T}$$

Outline

Binary choice

A choice between 2 alternatives is modeled through a $0,1$-variable.

Example: the knapsack problem

$$
\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{n} c_i x_i \\
\text{subject to} \quad & \sum_{i=1}^{n} a_i x_i \leq b \\
& x_i \in \{0,1\} \text{ for all } i = 1, \ldots, n.
\end{aligned}
$$

Forcing constraints

If decision A is made then decision B must be made as well.

$x = 1$	if decision A is taken	$y = 1$	if decision B is taken
$x = 0$	otherwise	$y = 0$	otherwise

The constraint reads

$$x \leq y$$

Example: Facility Location problem

- m clients ($i = 1, 2, \ldots, m$) to satisfy (demand $= 1$)
- n potential locations for facilities ($j = 1, 2, \ldots, n$)
- Can serve client i from facility j only if facility j is open:

$$x_{ij} \leq y_j$$

  - x_{ij} fraction of demand of client i served by facility j
  - $y_j \in \{0, 1\}$, 1 if facility is open.

Restricted range of values

Suppose we want to formulate $x \in \{a_1, a_2, \ldots, a_m\}$.

We introduce m binary variables y_j.

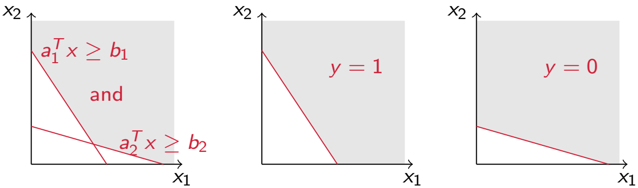
$$x = \sum_{j=1}^{m} a_j y_j, \quad \sum_{j=1}^{m} y_j = 1, \quad y_j \in \{0, 1\}$$

Disjunctive constraints

- Consider a variable $x \geq 0$,
- we want to model that either $a_1^T x \geq b_1$ or $a_2^T x \geq b_2$,
- and $a_1 \geq 0$, $a_2 \geq 0$.

We introduce a variable $y \in \{0, 1\}$ that represents whether constraint 1 ($y = 1$) or constraint 2 is satisfied, and replace both constraints by

$$a_1^T x \geq y b_1 \quad \text{and} \quad a_2^T x \geq (1 - y) b_2.$$



Exercise:

- extend to N disjunctive constraints;
- what if you want that exactly k of the N constraints satisfied simultaneously?

Disjunctive constraints (...)

- Now consider $0 \leq x \leq U$,
- we want to express either $a_1^T x \leq b_1$ or $a_2^T x \leq b_2$,
- without restriction on a_1 nor a_2.

Again, introduce variable $y \in \{0, 1\}$ and parameter M defined as

$$M = \max_{m, 0 \leq x \leq U} \left\{ m : m \geq a_i^T x - b_i, \quad i = 1, 2 \right\},$$

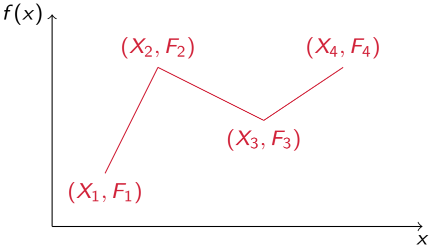
then

$$a_1^T x - b_1 \leq My \text{ and } a_2^T x - b_2 \leq M(1 - y).$$

Example: scheduling problem

- Two tasks, starting time $t_1, t_2 \geq 0$, duration $P_1, P_2 \geq 0$
- either task 1 is performed before task 2, or the opposite
- hence either $t_1 \geq t_2 + P_2$, or $t_2 \geq t_1 + P_1$

Arbitrary piecewise linear cost functions



Introduce variables $b_i \in \{0, 1\}$ such that

$$b_i = 1 \quad \text{if } x \in [X_i, X_{i+1}]$$
$$b_i = 0 \quad \text{if } x \notin [X_i, X_{i+1}]$$

Formulation 1:

- $\sum_i b_i = 1$
- $x_i \leq b_i$
- $f = \sum_i (F_i b_i + x_i(F_{i+1} - F_i))$

Formulation 2:

- $\sum_i b_i = 1$
- $\lambda_i \leq b_{i-1} + b_i,\ i = 2, \ldots, n-1$
- $\lambda_1 \leq b_1,\ \lambda_n \leq b_{n-1}$
- $\sum_i \lambda_i = 1$
- $f = \sum_i \lambda_i F_i$

Exercise: what if $f(x)$ is convex and we want to solve $\min_x \{f(x) \text{ s.t. } x \in X\}$?

Outline

The linear (continuous) relaxation

Definition

Given the Mixed Interger Program:

$$
\begin{aligned}
z_{MIP} = \min\ & c^T x + d^T y \\
\text{s.t.}\ \ & Ax + By = b \\
& x, y \geq 0 \\
& y \in \mathbb{R}^n \\
& x \in \mathbb{Z}^n,
\end{aligned}
$$

its linear relaxation is defined as

$$
\begin{aligned}
z_{LP} = \min\ & c^T x + d^T y \\
\text{s.t.}\ \ & Ax + By = b \\
& x, y \geq 0 \\
& y \in \mathbb{R}^n \\
& x \in \mathbb{R}^n.
\end{aligned}
$$

- The linear relaxation gives important information about the optimal value of an integer program:

$$z_{LP} \leq z_{MIP},$$

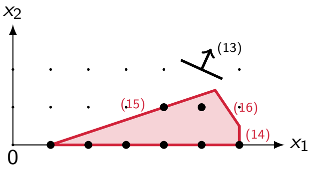
- hence, it is easy to obtain a lower bound (solving the relaxation is “easy”),
- but in general hard to obtain an integer solution from the solution of the relaxation without elaborated techniques.

Relaxation strength

- Alternative formulations of a problem may lead to different linear relaxations.
- If the formulation is ideal, that is, the LP relaxation defines the convex hull of the feasible set of the Integer Program, we need nothing else than Linear Programming algorithms. This often requires an exponential number of constraints.
- Here, we consider a different approach: **automatically** derive **valid inequalities** from the original constraints of the model in order to approximate the convex hull of the feasible points of the IP.

Illustration

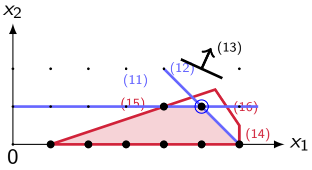
$$
\begin{aligned}
\max \quad & 5x_1 + 11x_2 && (6)\\
\text{s.t.} \quad & x_1 \le 6 && (7)\\
& x_1 - 3x_2 \ge 1 && (8)\\
& 3x_1 + 2x_2 \le 19 && (9)\\
& x_1, x_2 \in \mathbb{Z}_+ && (10)
\end{aligned}
$$



Illustration

$$\frac{(14)-(15)}{3} \text{ and } (17) \Rightarrow x_2 \leq 1 \quad (11)$$

$$\frac{(11)+(16)}{3} \text{ and } (17) \Rightarrow x_1 + x_2 \leq 6 \quad (12)$$



Valid inequalities

Definition

Let $P \subseteq \mathbb{R}^n$. An inequality $\sum_{j=1}^n a_j x_j \leq b$ is valid if it is satisfied by all points $x \in P$.

Typically,

- we want to derive valid inequalities for the set of integral solutions
- and at the same time cut off some part of the linear programming relaxation.

Outline

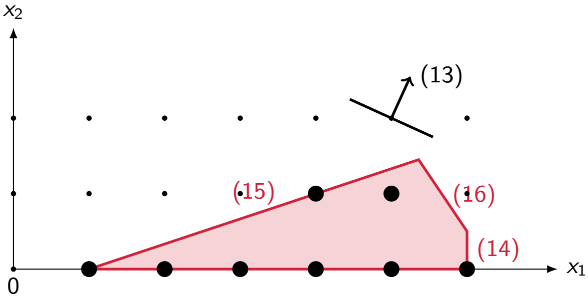
Introduction

- Here, we see an algorithm for solving to optimality an Integer Program when its formulation is not ideal: **branch-and-bound**.
- Other algorithms such as cutting-planes, are almost always used in conjunction with branch-and-bound (leading to the well known branch-and-cut algorithm).

Consider the following problem

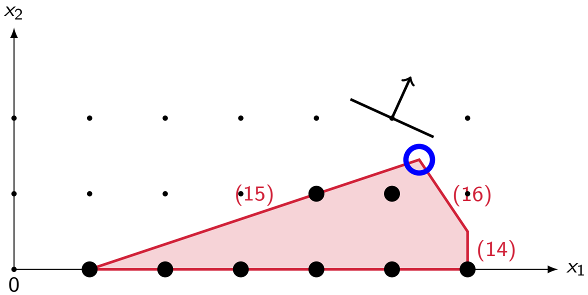
$$\begin{aligned}
\max \quad & 5x_1 + 11x_2 && (13)\\
\text{s.t.} \quad & x_1 \leq 6 && (14)\\
& x_1 - 3x_2 \geq 1 && (15)\\
& 3x_1 + 2x_2 \leq 19 && (16)\\
& x_1, x_2 \in \mathbb{Z}_+ && (17)
\end{aligned}$$

Geometrical view



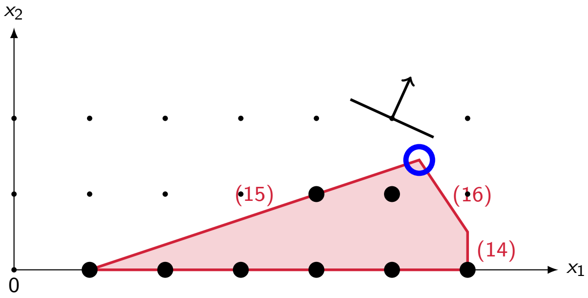
What information does the LP relaxation yield?

- Objective: $z^{\star,0} \approx 42.82$
- Solution: $x^{\star,0} \approx (5.36, 1.45)$



What information does the LP relaxation yield?

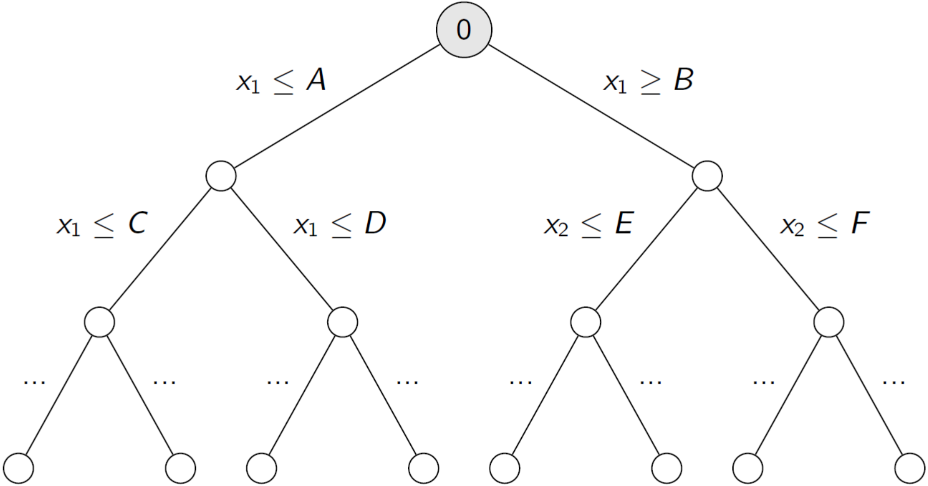
- Objective: $z^{\star,0} \approx 42.82$
- Solution: $x^{\star,0} \approx (5.36, 1.45)$



- Idea: enumerate, i.e. iteratively restrict the domain of x, but using the information of the linear relaxation.
- The enumeration yields a search tree.
- The root node of this tree is called the *root relaxation* (node 0 in the sequel).

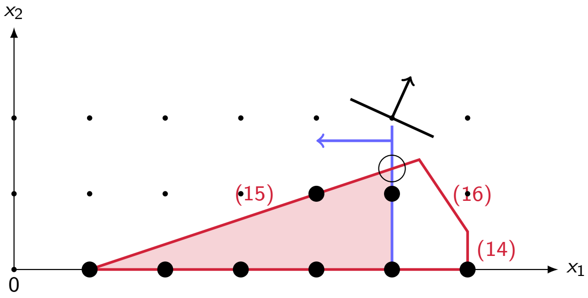
Use information of the relaxation to ...

- decide on which variables to branch
- set the thresholds
- prune parts of the search tree
- ...



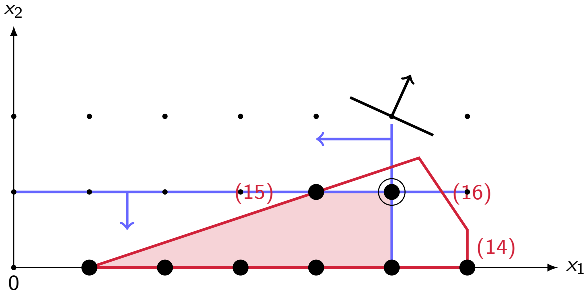
Back to our example, branch on x_1: $x_1 \leq 5$ (Node 1)

- $z^{\star,1} \approx 39.67$
- $x^{\star,1} = (5, 4/3)$



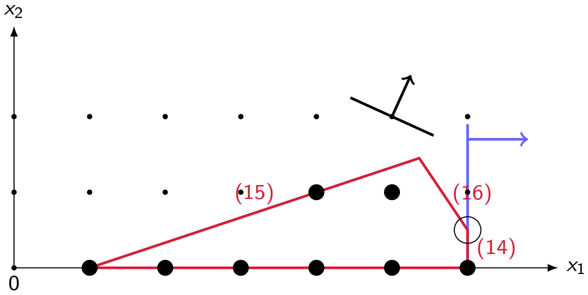
Node 2: from node 1, branch on x_2: $x_2 \leq 1$

- $z^{\star,2} = 36$
- $x^{\star,2} = (5, 1)$

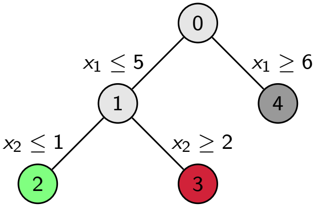


Node 4: the second alternative from the root node: $x_1 \geq 6$

- $z^{\star,4} = 35.5$
- $x^{\star,4} = (6, 1/2)$



Branch and bound tree



Remark: node index = order of exploration ! = order of creation.

Node	Nodes left	Objective	Nb integer infeasible variables	Best Integer	Best Bound	Gap	Decision
0	2	42,82	2	/	42,82	Infinite	Branch
1	2	39,67	1	/	42,82	Infinite	Branch
2	2	36	0	36	42,82	15,93%	prune by optimality
3	1	"-infinity"	/	36	42,82	15,93%	prune by infeasibility
4	0	35,5	1	36	35,5	0,00%	prune by bound

Remarks

- Opportunities to prune the search:
  - by bound,
  - by optimality,
  - by infeasibility
- Need of a good primal bound in the beginning
- Different strategies for the node selection:
  - depth-first-search (good to find quickly primal solutions)
  - breadth-first-search (good to increase the dual bound)
- Different strategies for variable selection:
  - Most fractional variable or least fractional variable
  - Take advantage of the history of branching
  - Look ahead for best improvement in the bound: strong branching