Lecture

Mathematical Optimization and Polyhedral Approaches

Section 1 : Solving combinatorial optimization problems using mathematical programming

Pierre Fouilhoux and Lucas Létocart

Université Sorbonne Paris Nord - LIPN CNRS

2026, January

1. Linear Programming
   - (Continuous) Linear Programming
   - Integer Linear Programming
   - Branch&Bound
2. List of MIP formulations
3. Compact formulation tricks

A first example from a College book

A yoghurt manufacturer produces 2 types A and B of strawberry yoghurt from strawberries, milk and sugar.
Each yoghurt must comply with the following proportions of raw materials.

	A	B
Strawberry	2	1
Milk	1	2
Sugar	0	1

Raw materials in limited quantities :
Strawberry : 800kg, Milk : 700kg
Sugar : 300kg.
The profit from the yogurt sales :
A : 4€ /kg et B : 5€ /kg

Modélisons

x_A quantity in kg of type A to be produced
x_B quantity in kg of type B to be produced

$$\begin{array}{llcll} \text{Max} & 4x_A & + & 5x_b & \\ & \frac{2}{3}x_A & + & \frac{1}{4}x_B & \leq 800 \\ & \frac{1}{3}x_A & + & \frac{1}{2}x_B & \leq 700 \\ & & & \frac{1}{4}x_B & \leq 300 \\ & x_A \geq 0 & & & \\ & & & x_B \geq 0 & \end{array}$$

Its a **(continuous) Linear Program**

A first example from a College book

A yoghurt manufacturer produces 2 types A and B of strawberry yoghurt from strawberries, milk and sugar.
Each yoghurt must comply with the following proportions of raw materials.

	A	B
Strawberry	2	1
Milk	1	2
Sugar	0	1

Raw materials in limited quantities :
Strawberry : 800kg, Milk : 700kg
Sugar : 300kg.
The profit from the yogurt sales :
A : 4€ /kg et B : 5€ /kg

Modélisons

x_A quantity in kg of type A to be produced
x_B quantity in kg of type B to be produced

$$\begin{array}{llcll} \text{Max} & 4x_A & + & 5x_b & \\ & \frac{2}{3}x_A & + & \frac{1}{4}x_B & \leq 800 \\ & \frac{1}{3}x_A & + & \frac{1}{2}x_B & \leq 700 \\ & & & \frac{1}{4}x_B & \leq 300 \\ & x_A \geq 0 & & & \\ & & & x_B \geq 0 & \end{array}$$

Its a **(continuous) Linear Program**

A first example from a College book

A yoghurt manufacturer produces 2 types A and B of strawberry yoghurt from strawberries, milk and sugar.
Each yoghurt must comply with the following proportions of raw materials.

	A	B
Strawberry	2	1
Milk	1	2
Sugar	0	1

Raw materials in limited quantities :
Strawberry : 800kg, Milk : 700kg
Sugar : 300kg.
The profit from the yogurt sales :
A : 4€ /kg et B : 5€ /kg

Modélisons

x_A quantity in kg of type A to be produced
x_B quantity in kg of type B to be produced

$$\begin{array}{llcll}
\text{Max} & 4x_A & + & 5x_b & \\
 & \frac{2}{3}x_A & + & \frac{1}{4}x_B & \leq 800 \\
 & \frac{1}{3}x_A & + & \frac{1}{2}x_B & \leq 700 \\
 & & & \frac{1}{4}x_B & \leq 300 \\
 & x_A \geq 0 & & & \\
 & & & x_B \geq 0 &
\end{array}$$

Its a **(continuous) Linear Program**

A first example from a College book

A yoghurt manufacturer produces 2 types A and B of strawberry yoghurt from strawberries, milk and sugar.
Each yoghurt must comply with the following proportions of raw materials.

	A	B
Strawberry	2	1
Milk	1	2
Sugar	0	1

Raw materials in limited quantities :
Strawberry : 800kg, Milk : 700kg
Sugar : 300kg.
The profit from the yogurt sales :
A : 4€ /kg et B : 5€ /kg

Modélisons

x_A quantity in kg of type A to be produced
x_B quantity in kg of type B to be produced

$$
\begin{array}{lcccccc}
\text{Max} & 4x_A & + & 5x_b & & \\
& \frac{2}{3}x_A & + & \frac{1}{4}x_B & \leq & 800 \\
& \frac{1}{3}x_A & + & \frac{1}{2}x_B & \leq & 700 \\
& & & \frac{1}{4}x_B & \leq & 300 \\
& x_A \geq 0 & & & & \\
& & & x_B \geq 0 & &
\end{array}
$$

Its a **(continuous) Linear Program**

A first example from a College book

A yoghurt manufacturer produces 2 types A and B of strawberry yoghurt from strawberries, milk and sugar.
Each yoghurt must comply with the following proportions of raw materials.

	A	B
Strawberry	2	1
Milk	1	2
Sugar	0	1

Raw materials in limited quantities :
Strawberry : 800kg, Milk : 700kg
Sugar : 300kg.
The profit from the yogurt sales :
A : 4€ /kg et B : 5€ /kg

Modélisons

x_A quantity in kg of type A to be produced
x_B quantity in kg of type B to be produced

$$\begin{array}{lllll} \text{Max} & 4x_A & + & 5x_b & \\ & \frac{2}{3}x_A & + & \frac{1}{4}x_B & \leq 800 \\ & \frac{1}{3}x_A & + & \frac{1}{2}x_B & \leq 700 \\ & & & \frac{1}{4}x_B & \leq 300 \\ & x_A \geq 0 & & & \\ & & & x_B \geq 0 & \end{array}$$

Its a **(continuous) Linear Program**

(Continuous) Linear Programming

Linear program :
Optimizing a linear function
with respect to
linear inequalities.

$$
\begin{array}{lccccc}
\text{Max} & 4x_A & + & 5x_b & & \\
& \frac{2}{3}x_A & + & \frac{1}{4}x_B & \leq & 800 \\
& \frac{1}{3}x_A & + & \frac{1}{2}x_B & \leq & 700 \\
& & & \frac{1}{4}x_B & \leq & 300 \\
& x_A \geq 0 & & x_B \geq 0 & &
\end{array}
$$

It's a continuous linear program...

But, we can also see it as a Combinatorial Optimization problem !

(Continuous) Linear Programming

Linear program :
Optimizing a linear function
with respect to
linear inequalities.

$$
\begin{array}{llllll}
\text{Max} & 4x_A & + & 5x_b & & \\
& \frac{2}{3}x_A & + & \frac{1}{4}x_B & \leq & 800 \\
& \frac{1}{3}x_A & + & \frac{1}{2}x_B & \leq & 700 \\
& & & \frac{1}{4}x_B & \leq & 300 \\
& x_A \geq 0 & & x_B \geq 0 & &
\end{array}
$$

It's a continuous linear program...

But, we can also see it
as a Combinatorial Optimization
problem !

(Continuous) Linear Programming

Linear program :
Optimizing a linear function
with respect to
linear inequalities.

$$
\begin{array}{llllll}
\text{Max} & 4x_A & + & 5x_b & & \\
& \frac{2}{3}x_A & + & \frac{1}{4}x_B & \leq & 800 \\
& \frac{1}{3}x_A & + & \frac{1}{2}x_B & \leq & 700 \\
& & & \frac{1}{4}x_B & \leq & 300 \\
& x_A \geq 0 & & x_B \geq 0 & &
\end{array}
$$

It's a continuous linear program...

But, we can also see it as a Combinatorial Optimization problem !

Graphical representation

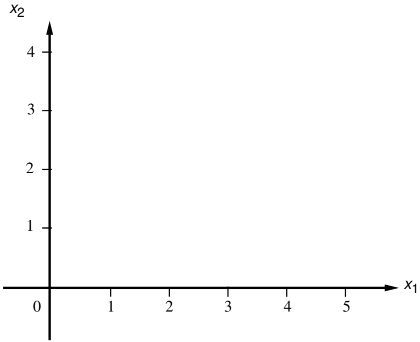
A linear program with 2 variables
can be embedded on a 2-dimensional space.

$$\begin{aligned} \text{Max } z = \quad & 2x_1 + x_2 \\ & x_1 - 4x_2 \leq 0 \\ & 3x_1 + 4x_2 \leq 15 \\ & x_1 \geq 0 \\ & x_2 \geq 0 \end{aligned}$$

Graphical representation

A linear program with 2 variables
can be embedded on a 2-dimensional space.

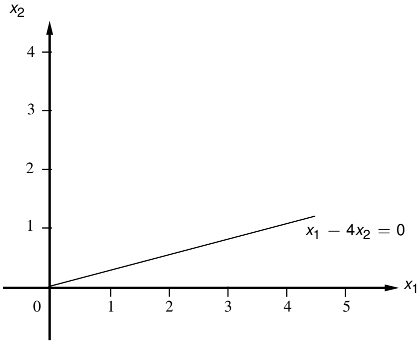
$$
\begin{aligned}
\text{Max } z = \quad & 2x_1 + x_2 \\
& x_1 \; - 4x_2 \leq 0 \\
& 3x_1 + 4x_2 \leq 15 \\
& x_1 \geq 0 \\
& x_2 \geq 0
\end{aligned}
$$



Graphical representation

A linear program with 2 variables
can be embedded on a 2-dimensional space.

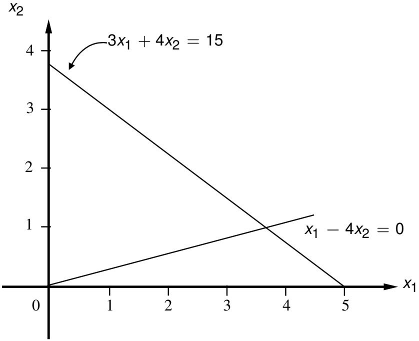
$$
\begin{aligned}
\text{Max } z = \quad & 2x_1 + x_2 \\
& x_1 - 4x_2 \leq 0 \\
& 3x_1 + 4x_2 \leq 15 \\
& x_1 \geq 0 \\
& x_2 \geq 0
\end{aligned}
$$



Graphical representation

A linear program with 2 variables
can be embedded on a 2-dimensional space.

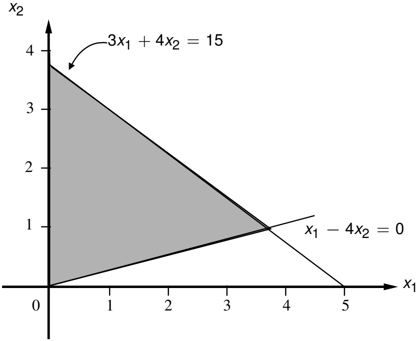
$$\begin{aligned} \text{Max } z = \quad & 2x_1 + x_2 \\ & x_1 - 4x_2 \leq 0 \\ & 3x_1 + 4x_2 \leq 15 \\ & x_1 \geq 0 \\ & x_2 \geq 0 \end{aligned}$$



Graphical representation

A linear program with 2 variables
can be embedded on a 2-dimensional space.

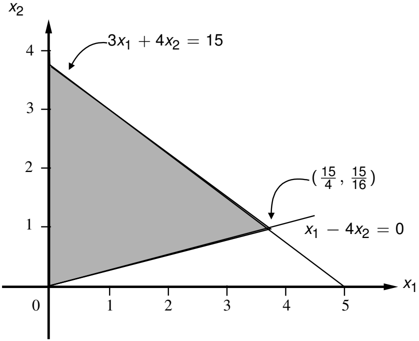
$$\begin{aligned} \text{Max } z = \quad & 2x_1 + x_2 \\ & x_1 - 4x_2 \leq 0 \\ & 3x_1 + 4x_2 \leq 15 \\ & x_1 \geq 0 \\ & x_2 \geq 0 \end{aligned}$$



Graphical representation

A linear program with 2 variables
can be embedded on a 2-dimensional space.

$$\begin{aligned} \text{Max } z = \quad & 2x_1 + x_2 \\ & x_1 - 4x_2 \leq 0 \\ & 3x_1 + 4x_2 \leq 15 \\ & x_1 \geq 0 \\ & x_2 \geq 0 \end{aligned}$$



Optimal solution
$x_1 = \frac{15}{4}$ $x_2 = \frac{15}{16}$

A combinatorial optimization problem ?

A linear program describes a set of solutions which is a **polyhedron**.

For 2 variables,
a polygon.

for 3 variables,
a “3D” mathematical figure.

For *n* variables...

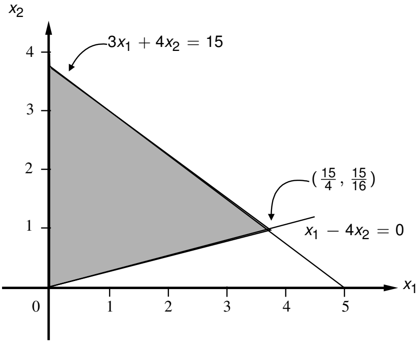
A combinatorial optimization problem ?

A linear program describes a set of solutions which is a **polyhedron**.

For 2 variables,
a polygon.

for 3 variables,
a “3D” mathematical figure.

For n variables...



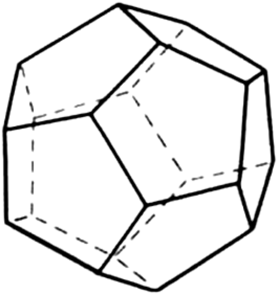
A combinatorial optimization problem ?

A linear program describes a set of solutions which is a **polyhedron**.

For 2 variables,
a polygon.

for 3 variables,
a "3D" mathematical figure.

For n variables...



A combinatorial optimization problem ?

A linear program describes a set of solutions which is a **polyhedron**.

For 2 variables,
a polygon.

...

for 3 variables,
a "3D" mathematical figure.

For n variables...

A combinatorial optimization problem ?

The **optimal** solution
of a (continous) linera programm
can be chosen
among the **extrem points**
(i.e. the intersection
of n facets of the polyhedron.

For 2 variables,
intersection of 2 straight lines

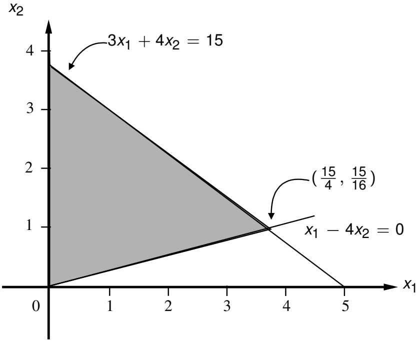
For 3 variables,
intersection of 3 facets

A combinatorial optimization problem ?

The **optimal** solution
of a (continous) linera programm
can be chosen
among the **extrem points**
(i.e. the intersection
of n facets of the polyhedron.

For 2 variables,
intersection of 2 straight lines

For 3 variables,
intersection of 3 facets

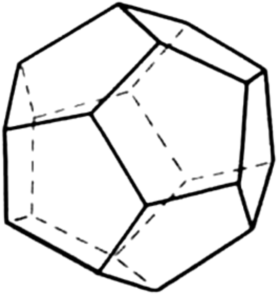


A combinatorial optimization problem ?

The **optimal** solution
of a (continous) linera programm
can be chosen
among the **extrem points**
(i.e. the intersection
of *n* facets of the polyhedron.

For 2 variables,
intersection of 2 straight lines

For 3 variables,
intersection of 3 facets



A combinatorial optimization problem ?

The **optimal** solution
of a (continous) linera programm
can be chosen
among the **extrem points**
(i.e. the intersection
of n facets of the polyhedron.

For a linear program
of n variables
and m linear inéqualities
and n inequalities $x_i \geq 0$

How many potential optimal solutions ?

A most as many as the way to take
n inequalities among $n + m$:

$$C_n^{n+m} = \frac{(n+m)!}{n!m!}$$

It's an exponential number of solutions !

Its a combinatorial Optimization problem !

A combinatorial optimization problem ?

The **optimal** solution
of a (continous) linera programm
can be chosen
among the **extrem points**
(i.e. the intersection
of n facets of the polyhedron.

For a linear program
of n variables
and m linear inéqualities
and n inequalities $x_i \geq 0$

How many potential optimal solutions ?

A most as many as the way to take
n inequalities among $n + m$:

$$C_n^{n+m} = \frac{(n+m)!}{n!m!}$$

It's an exponential number of solutions !

Its a combinatorial Optimization problem !

A combinatorial optimization problem ?

The **optimal** solution
of a (continous) linera programm
can be chosen
among the **extrem points**
(i.e. the intersection
of n facets of the polyhedron.

For a linear program
of n variables
and m linear inéqualities
and n inequalities $x_i \geq 0$

How many potential optimal solutions ?

A most as many as the way to take
n inequalities among $n+m$:

$$\mathcal{C}_n^{n+m} = \frac{(n+m)!}{n!m!}$$

It's an exponential number of solutions !

Its a combinatorial Optimization problem !

A combinatorial optimization problem ?

The **optimal** solution
of a (continous) linera programm
can be chosen
among the **extrem points**
(i.e. the intersection
of n facets of the polyhedron.

For a linear program
of n variables
and m linear inéqualities
and n inequalities $x_i \geq 0$

How many potential optimal solutions ?

A most as many as the way to take
n inequalities among $n+m$:

$$\mathcal{C}_n^{n+m} = \frac{(n+m)!}{n!m!}$$

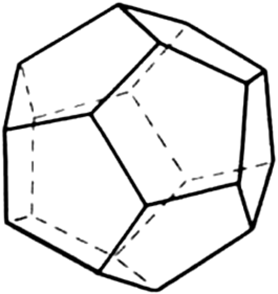
It's an exponential number of solutions !

Its a combinatorial Optimization problem !

Simplex algorithm (G. Dantzig (1947))

The main ideas are

- to represent each extrem points trought very simple algebraic notations
- to start from a known exrem points and go to another following the edges of the polyhedron.
- at each step the next solution is better than or equal the previous one
- each iteration takes a few millisecond even for huge LP
- a linear time optimality test says whether the optimal solution is reached.



Simplex algorithm (G. Dantzig (1947)

- **Ending** of the simplex algorithm :
  - With each iteration, the objective value increases (in the broadest sense).
  - The number of iterations is bounded by the number of extruded points of the polyhedron.

How many iterations they are in the worst case ?

Klee et Minty have exhibited this LP

$$\begin{array}{lll} \text{Max} & \sum_{j=1}^{n} 10^{n-j} x_j & \\ & \left(2 \sum_{j=1}^{i-1} x_j\right) + x_i \leq 100^{i-1} & \forall i \in \{1, \ldots, n\} \\ & x_j \geq 0 & \forall j \in \{1, \ldots, n\} \end{array}$$

This LP corresponds to $2^n - 1$ extrem points that the Simplex algorithm explores one after the other :
Ainsi The simplex algorithm is exponential from the worst-case analysis.

Simplex algorithm (G. Dantzig (1947))

- **Ending** of the simplex algorithm :
  - With each iteration, the objective value increases (in the broadest sense).
  - The number of iterations is bounded by the number of extruded points of the polyhedron.

How many iterations they are in the worst case ?

Klee et Minty have exhibited this LP

$$\begin{array}{lll} \text{Max} & \sum_{j=1}^{n} 10^{n-j} x_j & \\ & \left(2 \sum_{j=1}^{i-1} x_j\right) + x_i \leq 100^{i-1} & \forall i \in \{1, \ldots, n\} \\ & x_j \geq 0 & \forall j \in \{1, \ldots, n\} \end{array}$$

This LP corresponds to $2^n - 1$ extrem points that the Simplex algorithm explores one after the other :
Ainsi The simplex algorithm is exponential from the worst-case analysis.

Simplex algorithm (G. Dantzig (1947))

- **Ending** of the simplex algorithm :
  - With each iteration, the objective value increases (in the broadest sense).
  - The number of iterations is bounded by the number of extruded points of the polyhedron.

How many iterations they are in the worst case ?

Klee et Minty have exhibited this LP

$$\begin{array}{lll} \text{Max} & \sum_{j=1}^{n} 10^{n-j} x_j & \\ & \left(2 \sum_{j=1}^{i-1} x_j\right) + x_i \leq 100^{i-1} & \forall i \in \{1, \ldots, n\} \\ & x_j \geq 0 & \forall j \in \{1, \ldots, n\} \end{array}$$

This LP corresponds to $2^n - 1$ extrem points that the Simplex algorithm explores one after the other :
Ainsi The simplex algorithm is exponential from the worst-case analysis.

“In practice”

Even if the Simplex Algorithm has an exponential capacity in a worst case,
Worst case appears very very rarely **“in pratice”**
with a very few iterations each times !

But what is the signification of “in pratice” :

- LP coming from real optimization probems
- with rationnal coefficients
- these coefficients have values far one from the other
- ...

It is difficult to describe this “in pratice”, but it’s the reality of the daily use of Simplex algorithm. Perhaps that worst case LP have **a rare combinatorial structure** far from the “practice”...
And somehow this famous behavior of Simplex Algorithm let think that perhaps the question ($P? = NP$) is not so important “in practice” !

“In practice”

Even if the Simplex Algorithm has an exponential capacity in a worst case,
Worst case appears very very rarely **“in pratice”**
with a very few iterations each times !

But what is the signification of “in pratice” :

- LP coming from real optimization probems
- with rationnal coefficients
- these coefficients have values far one from the other
- ...

It is difficult to describe this “in pratice”, but it’s the reality of the daily use of Simplex algorithm. Perhaps that worst case LP have **a rare combinatorial structure** far from the “practice”...
And somehow this famous behavior of Simplex Algorithm let think that perhaps the question ($P? = NP$) is not so important “in practice” !

“In practice”

Even if the Simplex Algorithm has an exponential capacity in a worst case,
Worst case appears very very rarely **“in pratice”**
with a very few iterations each times !

But what is the signification of “in pratice” :

- LP coming from real optimization probems
- with rationnal coefficients
- these coefficients have values far one from the other
- ...

It is difficult to describe this “in pratice”, but it’s the reality of the daily use of Simplex algorithm. Perhaps that worst case LP have **a rare combinatorial structure** far from the “practice”...
And somehow this famous behavior of Simplex Algorithm let think that perhaps the question ($P? = NP$) is not so important “in practice” !

(Continuous) linear programming complexity

The Simplex algortihm is not the only one to solve Linear Programs : the complexity of Linear programming is the complexity of the best algorithm to solve any LP.

1970 Into the 1870's the question of solvinf LP was officialy asked and the Simplex Algorithm in 1947 have not answer the question with its exponential complexity shown by Klee et Minty in 1970.

1979 Leonid Khatchian inspired by the **ellipsoid metho** known in another context proposes a first polynomal algorithm for LP !
(Continuous) Linear Programming is then polynomial !
But the polynom degree of the complexity of ellipsoid methdeis rather high and so useless !

1984 Narendra Karmarkar proposes the **interior point method** which is polynomial and (now) efficient !

2000 Francisco Barahona and Anbil propose the **Volume algorithm**, polynomial and with good structural properties

2022 Sophie Huiberts and Daniel Dadush show that random modifications over the LP data will make transform many worst cases into simple ones without changing the solving solution... then the question is : is there is exponential cases left ?

(Continuous) linear programming complexity

The Simplex algortihm is not the only one to solve Linear Programs : the complexity of Linear programming is the complexity of the best algorithm to solve any LP.

1970 Into the 1870's the question of solvinf LP was officialy asked and the Simplex Algorithm in 1947 have not answer the question with its exponential complexity shown by Klee et Minty in 1970.

1979 Leonid Khatchian inspired by the **ellipsoid metho** known in another context proposes a first polynomal algorithm for LP !
(Continuous) Linear Programming is then polynomial !
But the polynom degree of the complexity of ellipsoid methdeis rather high and so useless !

1984 Narendra Karmarkar proposes the **interior point method** which is polynomial and (now) efficient !

2000 Francisco Barahona and Anbil propose the **Volume algorithm**, polynomial and with good structural properties

2022 Sophie Huiberts and Daniel Dadush show that random modifications over the LP data will make transform many worst cases into simple ones without changing the solving solution... then the question is : is there is exponential cases left ?

(Continuous) linear programming complexity

The Simplex algortihm is not the only one to solve Linear Programs : the complexity of Linear programming is the complexity of the best algorithm to solve any LP.

1970 Into the 1870's the question of solvinf LP was officialy asked and the Simplex Algorithm in 1947 have not answer the question with its exponential complexity shown by Klee et Minty in 1970.

1979 Leonid Khatchian inspired by the **ellipsoid metho** known in another context proposes a first polynomal algorithm for LP !
(Continuous) Linear Programming is then polynomial !
But the polynom degree of the complexity of ellipsoid methdeis rather high and so useless !

1984 Narendra Karmarkar proposes the **interior point method** which is polynomial and (now) efficient !

2000 Francisco Barahona and Anbil propose the **Volume algorithm**, polynomial and with good structural properties

2022 Sophie Huiberts and Daniel Dadush show that random modifications over the LP data will make transform many worst cases into simple ones without changing the solving solution... then the question is : is there is exponential cases left ?

(Continuous) linear programming complexity

The Simplex algortihm is not the only one to solve Linear Programs : the complexity of Linear programming is the complexity of the best algorithm to solve any LP.

1970 Into the 1870's the question of solvinf LP was officialy asked and the Simplex Algorithm in 1947 have not answer the question with its exponential complexity shown by Klee et Minty in 1970.

1979 Leonid Khatchian inspired by the **ellipsoid metho** known in another context proposes a first polynomal algorithm for LP !
(Continuous) Linear Programming is then polynomial !
But the polynom degree of the complexity of ellipsoid methdeis rather high and so useless !

1984 Narendra Karmarkar proposes the **interior point method** which is polynomial and (now) efficient !

2000 Francisco Barahona and Anbil propose the **Volume algorithm**, polynomial and with good structural properties

2022 Sophie Huiberts and Daniel Dadush show that random modifications over the LP data will make transform many worst cases into simple ones without changing the solving solution... then the question is : is there is exponential cases left ?

(Continuous) linear programming complexity

The Simplex algortihm is not the only one to solve Linear Programs : the complexity of Linear programming is the complexity of the best algorithm to solve any LP.

1970 Into the 1870's the question of solvinf LP was officialy asked and the Simplex Algorithm in 1947 have not answer the question with its exponential complexity shown by Klee et Minty in 1970.

1979 Leonid Khatchian inspired by the **ellipsoid metho** known in another context proposes a first polynomal algorithm for LP !
(Continuous) Linear Programming is then polynomial !
But the polynom degree of the complexity of ellipsoid methdeis rather high and so useless !

1984 Narendra Karmarkar proposes the **interior point method** which is polynomial and (now) efficient !

2000 Francisco Barahona and Anbil propose the **Volume algorithm**, polynomial and with good structural properties

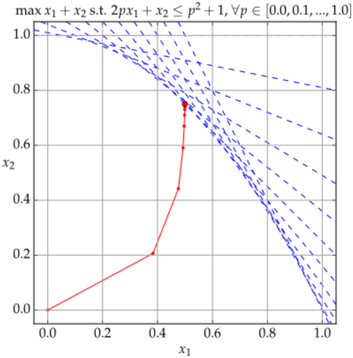
2022 Sophie Huiberts and Daniel Dadush show that random modifications over the LP data will make transform many worst cases into simple ones without changing the solving solution... then the question is : is there is exponential cases left ?

Interior point method

As its name indicates, the algorithm moves inside the polyhedron in the direction orthogonal to the objective vector.

The difficulty is not to go outside the polyhedron !

Even if it is theoretically polynomial unlike the simplex algorithm which is theoretically of exponential worst-case complexity,
the interior points method is **slower** on small instances
(but faster if more than 200,000 inequalities).



(Source : Wikipedia)

(Continous) linear solver

The
- polynomial interior point method
- (officially exponential but efficient) simplex algorithm.

have been implemented in numerous softwares called **linear solvers**.

- the historical commercial solver is Cplex (IBM) but a similar code exists in the powerful Gurobi
- there are other solvers like Xpress (and even Matlab or Excel !)
- solver from university LP (COIN-OR), Soplex (ZIB) inside the SCIP project, HIGHS from Scotland, Hexaly (ex-LocalSolver)...
- one totally free solver : GLPK(gnu)

The best of them can solve PLs up to 200 000 variables and 200 000 constraints in a few minutes.

Note that they have easy-to-access interface : text file, “simple or advanced modelers”, languages (C, C++, Python, Julia...).

1. Linear Programming
   - (Continuous) Linear Programming
   - Integer Linear Programming
   - Branch&Bound
2. List of MIP formulations
3. Compact formulation tricks

Integer linear program (MIP)

$$
\begin{aligned}
\text{Max } & c_1^T x_1 + c_2^T x_2 \\
& \text{s.t.} \\
& A_1 x_1 + A_2 x_2 \leq b \\
& x_1 \in \boldsymbol{R}^{n_1} \\
& x_2 \in \boldsymbol{Z}^{n_2}.
\end{aligned}
$$

with x_1 continuous
et x_2 integer

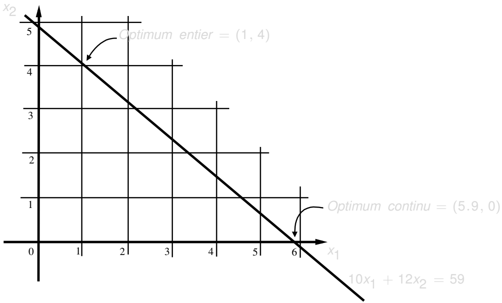
Constraints $x_2 \in \boldsymbol{Z}^{n_2}$ are called *integrity constraints*.

The first observation that can jump out at you is to imagine that MIP solving amounts to “rounding” the solution of its continuous relaxation. The following example demonstrates the inadequacy of this observation :

Let us consider this very simple MIP

$$\begin{aligned} &\text{Maximiser } 10x_1 + 11x_2 \\ &\quad 10x_1 + 12x_2 \leq 59 \\ &\quad x_1 \text{ et } x_2 \geq 0 \\ &\quad x_1, x_2 \text{ entiers.} \end{aligned}$$

and draw the corresponding polyhedron :

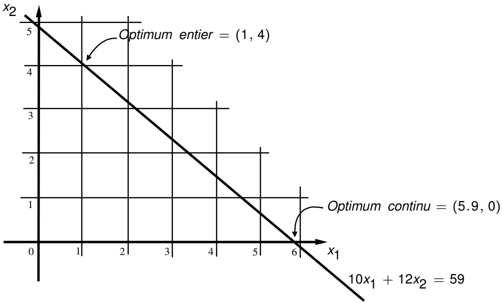


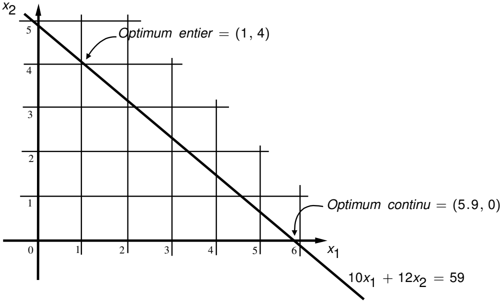
The first observation that can jump out at you is to imagine that MIP solving amounts to “rounding” the solution of its continuous relaxation. The following example demonstrates the inadequacy of this observation :

Let us consider this very simple MIP

$$\begin{aligned} \text{Maximiser } & 10x_1 + 11x_2 \\ & 10x_1 + 12x_2 \leq 59 \\ & x_1 \text{ et } x_2 \geq 0 \\ & x_1, x_2 \text{ entiers.} \end{aligned}$$

and draw the corresponding polyhedron :





We then notice that

- the optimum of the continuous relaxation has an objective value of 59 and that of the entire optimum is only 54.. and imagine that the unit is about bilion dollars !
- the important structural difference between these two points (which are here each a unique optimal solution of the LP and the MIP).

Complexity of Integer Linear Programming

It is easy to formulate the knapsack problem as a MIP

Knapsack MIP

$$
\begin{aligned}
\text{Max } & \sum_{i=1}^{n} w_i x_i \\
& \sum_{i=1}^{n} p_i x_i \leq P, \\
& 0 \leq x_i \leq 1, \text{ for each item } i = 1, \dots, n, \\
& x_i \in \boldsymbol{N}, \text{ for each item } i = 1, \dots, n.
\end{aligned}
$$

Complexity of Integer Linear Programming

It is easy to formulate the knapsack problem as a MIP

Knapsack MIP

$$
\begin{aligned}
\text{Max} \sum_{i=1}^{n} w_i x_i & \\
& \sum_{i=1}^{n} p_i x_i \leq P, \\
& 0 \leq x_i \leq 1, \ \text{for each item } i = 1, \ldots, n, \\
& x_i \in \mathbf{N}, \ \text{for each item } i = 1, \ldots, n.
\end{aligned}
$$

Hence solving a MIP is at least hard as solving a weakly NP-hard problem

Complexity of Integer Linear Programming

It is also direct to formulate the stable set problem as a MIP

Stable set MIP

$$
\begin{aligned}
\text{Max} \sum_{u \in V} c_u x_u & \\
x_u + x_v \leq 1, & \quad \text{for each edge } uv, \\
0 \leq x_u \leq 1 & \quad \text{for each node } u, \\
x_u \in \{0, 1\}, & \quad \text{for each node } u.
\end{aligned}
$$

Complexity of Integer Linear Programming

It is also direct to formulate the stable set problem as a MIP

Stable set MIP

$$
\begin{aligned}
\text{Max} \sum_{u \in V} c_u x_u & \\
x_u + x_v \leq 1, & \quad \text{for each edge } uv, \\
0 \leq x_u \leq 1 & \quad \text{for each node } u, \\
x_u \in \{0, 1\}, & \quad \text{for each node } u.
\end{aligned}
$$

Hence solving a MIP is at least hard as solving a strongly NP-hard problem

Programmation linéaire en nombres entiers

Then all the known methods to exactly solved a MIP are exponential.
They are based on the principle of Branch&Bound, using mathematical tools to better prune unsuccessful branches (polyhedral approaches, Branch&Cut algorithms, strong branching...).

These methods are gathed into **Integer Solvers** that have became more and more powerful during 30 years... but are sometimes limited to a few thousand variables for hard problems !

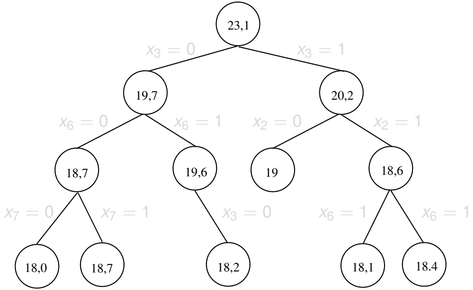
- The famous commercial solver Cplex have now lower performance than the Gurobi (and Xpress is really under these two). A new one : Hexali is performing well
- The university project like SCIP or HIGHS are much less efficicent

The strength of the project between these solvers is to be closed to research and new ideas !

1. Linear Programming
   - (Continuous) Linear Programming
   - Integer Linear Programming
   - Branch&Bound
2. List of MIP formulations
3. Compact formulation tricks

A **Branch&Bound algorithm** is defined by :
- an integer linear program
- a branching strategy (potentially on inequalities)

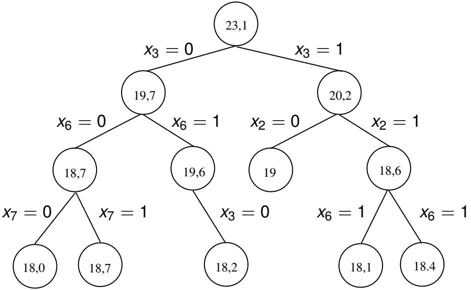
$\rightarrow$ B&B node = initial integer program + branching inequalities



v = linear relaxation of a B&B node
For a “max” problem, v is is an **upper bound** for the sub-tree
(each node only contains solutions of values at most v).

A **Branch&Bound algorithm** is defined by :
- an integer linear program
- a branching strategy (potentially on inequalities)

$\rightarrow$ B&B node = initial integer program + branching inequalities

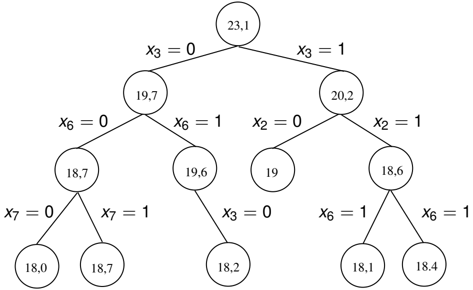


v = linear relaxation of a B&B node

For a "max" problem, v is is an **upper bound** for the sub-tree
(each node only contains solutions of values at most v).

A **Branch&Bound algorithm** is defined by :
- an integer linear program
- a branching strategy (potentially on inequalities)

$\rightarrow$ B&B node = initial integer program + branching inequalities



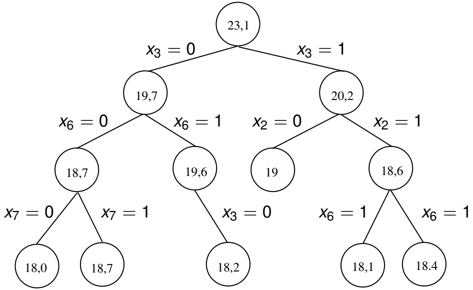
v = linear relaxation of a B&B node
For a "max" problem, v is is an **upper bound** for the sub-tree (each node only contains solutions of values at most v).

During the B&B algorithm, a **best known solution** can be obtained using heuristic, metaheuristic, rounding heurisitic,...

A sub-tree can be pruned if either

- the relaxation value of its root node

  is $\leq$ than the value of the best known solution.
- its root node is **integer** (its relaxation gives an integer solution)

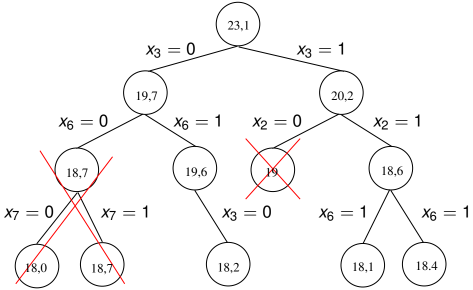


During the B&B algorithm, a **best known solution** can be obtained using heuristic, metaheuristic, rounding heurisitic,...

A sub-tree can be pruned if either

- the relaxation value of its root node

  is $\leq$ than the value of the best known solution.
- its root node is **integer** (its relaxation gives an integer solution)



Efficiency of a B&B algorithm

Several aspects drive to an efficient B&B algorithm :

- Initial preprocessing step
- Breaking symmetries within the B&B tree
- Having the lowest possible upper bound
- Having the highest possible lower bound
- Having a lot of integer nodes

Efficiency of a B&B algorithm

Several aspects drive to an efficient B&B algorithm :

- Initial preprocessing step
- Breaking symmetries within the B&B tree
- **Having the lowest possible upper bound**
  $\rightarrow$ Having the best possible relaxation value
- Having the highest possible lower bound
- Having a lot of integer nodes

Efficiency of a B&B algorithm

Several aspects drive to an efficient B&B algorithm :

- Initial preprocessing step
- Breaking symmetries within the B&B tree
- **Having the lowest possible upper bound**
  $\rightarrow$ Having the best possible relaxation value
- **Having the highest possible lower bound**
  $\rightarrow$ Having a good rounding heuristic
- Having a lot of integer nodes

Efficiency of a B&B algorithm

Several aspects drive to an efficient B&B algorithm :

- Initial preprocessing step
- Breaking symmetries within the B&B tree
- **Having the lowest possible upper bound**
  $\rightarrow$ Having the best possible relaxation value
- **Having the highest possible lower bound**
  $\rightarrow$ Having a good rounding heuristic
- **Having a lot of integer nodes**
  $\rightarrow$ If the relaxed linear programs have "often" integer solutions

Combinatorial Optimization Problem

To find a greatest (smallest) element within a valuated finite set.

Given :
- a finite subset of elements $E = \{e_1, \ldots, e_n\}$
- a **solution set** $\mathcal{F}$ of subsets of E
- a weight $c = (c(e_1), \ldots, c(e_n))$

a **Combinatorial Optimization Problem** is to find a solution $F \in \mathcal{F}$ whose weight $c(F) = \sum_{e \in F} c(e)$ is maximum (or min.),

$$i.e. \quad \max \{c(F) \mid F \in \mathcal{F}\}.$$

Combinatorial Optimization Problem

To find a greatest (smallest) element within a valuated finite set.

Given :
- a finite subset of elements $E = \{e_1, \ldots, e_n\}$
- a **solution set** $\mathcal{F}$ of subsets of E
- a weight $c = (c(e_1), \ldots, c(e_n))$

a **Combinatorial Optimization Problem** is to find a solution $F \in \mathcal{F}$ whose weight $c(F) = \sum_{e \in F} c(e)$ is maximum (or min.),

$$\textit{i.e.} \quad \max \{c(F) \mid F \in \mathcal{F}\}.$$

“Naive” algebraic formulation algébrique

Associate a binary variable to every solution :
$t_F = 1$ if the solution $F \in \mathcal{F}$ is choosen et 0 otherwise

$$\text{Max} \sum_{F \in \mathcal{F}} c(F) t_F$$
$$\sum_{F \in \mathcal{F}} t_F \leq 1$$
$$t_F \in \{0,1\} \quad \forall F \in \mathcal{F}.$$

This formulation proves that any Combinatorial Optimization problem can be written as a MIP !

But it’s not so obvious we can solve such a formulation with the number of variables equal to the number of solutions ! !

“Naive” algebraic formulation algébrique

Associate a binary variable to every solution :
$t_F = 1$ if the solution $F \in \mathcal{F}$ is choosen et 0 otherwise

$$\text{Max} \sum_{F \in \mathcal{F}} c(F) t_F$$
$$\sum_{F \in \mathcal{F}} t_F \leq 1$$
$$t_F \in \{0,1\} \quad \forall F \in \mathcal{F}.$$

This formulation proves that any Combinatorial Optimization problem can be written as a MIP !

But it's not so obvious we can solve such a formulation with the number of variables equal to the number of solutions ! !

“Naive” algebraic formulation algébrique

Associate a binary variable to every solution :
$t_F = 1$ if the solution $F \in \mathcal{F}$ is choosen et 0 otherwise

$$
\begin{aligned}
\text{Max} \ \sum_{F \in \mathcal{F}} c(F) t_F \\
\sum_{F \in \mathcal{F}} t_F \leq 1 \\
t_F \in \{0, 1\} \quad \forall F \in \mathcal{F}.
\end{aligned}
$$

This formulation proves that any Combinatorial Optimization problem can be written as a MIP !

But it’s not so obvious we can solve such a formulation with the number of variables equal to the number of solutions ! !

“Naive” algebraic formulation algébrique

Associate a binary variable to every solution :
$t_F = 1$ if the solution $F \in \mathcal{F}$ is choosen et 0 otherwise

$$\begin{aligned} \text{Max} \sum_{F \in \mathcal{F}} c(F) t_F \\ \sum_{F \in \mathcal{F}} t_F \leq 1 \\ t_F \in \{0,1\} \quad \forall F \in \mathcal{F}. \end{aligned}$$

This formulation proves that any Combinatorial Optimization problem can be written as a MIP !

But it’s not so obvious we can solve such a formulation with the number of variables equal to the number of solutions ! !

"Natural" MIP formulation

It's quite "natural" to set a variable for each element.

Binary variable $x_e = \begin{cases} 1 & \text{if } e \text{ is chosen} \\ 0 & \text{otherwise} \end{cases}$ for every $e \in E$.

$$\max \sum_{e \in E} c(e) x_e$$
$$Ax \leq b$$
$$x_e \in \{0,1\} \quad \forall e \in E.$$

BUT : To have such a formulation, $Ax \leq b$ must be known ! !

"Natural" MIP formulation

It's quite "natural" to set a variable for each element.

Binary variable $x_e = \begin{cases} 1 & \text{if } e \text{ is chosen} \\ 0 & \text{otherwise} \end{cases}$ for every $e \in E$.

$$\max \sum_{e \in E} c(e) x_e$$
$$Ax \leq b$$
$$x_e \in \{0, 1\} \quad \forall e \in E.$$

BUT : To have such a formulation, $Ax \leq b$ must be known ! !

"Natural" MIP formulation

It's quite "natural" to set a variable for each element.

Binary variable $x_e = \begin{cases} 1 & \text{if } e \text{ is chosen} \\ 0 & \text{otherwise} \end{cases}$ for every $e \in E$.

$$\max \sum_{e \in E} c(e) x_e$$
$$Ax \leq b$$
$$x_e \in \{0, 1\} \quad \forall e \in E.$$

BUT : To have such a formulation, $Ax \leq b$ must be known !!

Compact /non-compact MIP formulation

• An inequality set $Ax \leq b$ can be **compact**,
i.e. contain a polynomial number of inequalities and variables !

In this case, we can enumerate the inequalities and gives them to an integer solver.

• An inequality set $Ax \leq b$ can be **non-compact**,
i.e. having either an exponential number of variables or inequalities (or both !).

In this case, we need to study how to do solve such formulations !

Compact /non-compact MIP formulation

• An inequality set $Ax \leq b$ can be **compact**,
i.e. contain a polynomial number of inequalities and variables !

In this case, we can enumerate the inequalities and gives them to an integer solver.

• An inequality set $Ax \leq b$ can be **non-compact**,
i.e. having either an exponential number of variables or inequalities (or both !).

In this case, we need to study how to do solve such formulations !

0/1-Knapsack problem

Input : n objects
profit $g_i \; \forall i \in \{1, \ldots, n\}$
weight $p_i \; \forall i \in \{1, \ldots, n\}$
maximum total weight P.

Output : Subset $S \subseteq \{1, \ldots, n\}$
s.t. $\sum_{i \in S} p_i \leq P$

Objective : Max $\sum_{i \in S} g_i$

Knapsack problem

Very direct compact formulation !

$$x_i = \begin{cases} 1 & \text{if item } i \text{ is chosen} \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{aligned} \text{Max } & \sum_{i=1}^{n} g_i x_i \\ & \sum_{i=1}^{n} p_i x_i \leq P \\ & 0 \leq x_i \leq 1 \ \ \forall i \in \{1, \ldots, n\} \\ & x_i \in N \ \ \forall i \in \{1, \ldots, n\} \end{aligned}$$

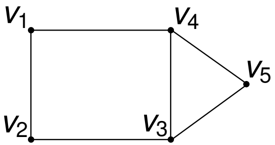
The single constraint is called the *knapsack constraint*.
This is the single constraint of a problem which is NP-hard, though !

Maximum weight stable set problem

Input : Undirected graph $G = (V, E)$
cost $w_u \ \forall u \in V$

Output : Subset $S \subseteq V$ of non-adjacent nodes

Objective : Max $\sum_{i \in S} w_i$



Compact formulation for the stable set problem

$$x_u = \begin{cases} 1 & \text{if node } u \text{ is chosen} \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{aligned} \text{Max} \sum_{u \in V} w_u x_u & \\ x_u + x_v \leq 1, & \quad \forall uv \in E, \\ x_u \in \{0, 1\}, & \quad \forall u \in V. \end{aligned}$$

Inequality $xu_+ x_v \leq 1$ is called edge inequality.

Compact formulation with $|V|$ variables and $|E|$ edges.

But this MIP formulation is hard to be solved by a (pure) Branch&Bound as its relaxation value is very low !

Indeed the fractional solution $\bar{x}_u = \frac{1}{2} \ \forall u \in V$ satisfies the constraints resulting a huge upper bound of at least $\frac{1}{2} \sum_{u \in V} x_u$.

Compact formulation for the stable set problem

$$x_u = \begin{cases} 1 & \text{if node } u \text{ is chosen} \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Max} \sum_{u \in V} w_u x_u$$
$$x_u + x_v \leq 1, \quad \forall uv \in E,$$
$$x_u \in \{0, 1\}, \quad \forall u \in V.$$

Inequality $xu_+ x_v \leq 1$ is called edge inequality.

Compact formulation with $|V|$ variables and $|E|$ edges.

But this MIP formulation is hard to be solved by a (pure) Branch&Bound as its relaxation value is very low !

Indeed the fractional solution $\bar{x}_u = \frac{1}{2} \ \forall u \in V$ satisfies the constraints resulting a huge upper bound of at least $\frac{1}{2} \sum_{u \in V} x_u$.

Compact formulation for the stable set problem

$$x_u = \begin{cases} 1 & \text{if node } u \text{ is chosen} \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{aligned} \text{Max} \sum_{u \in V} w_u x_u & \\ x_u + x_v \leq 1, & \quad \forall uv \in E, \\ x_u \in \{0, 1\}, & \quad \forall u \in V. \end{aligned}$$

Inequality $xu_+ x_v \leq 1$ is called edge inequality.

Compact formulation with $|V|$ variables and $|E|$ edges.

But this MIP formulation is hard to be solved by a (pure) Branch&Bound as its relaxation value is very low !

Indeed the fractional solution $\bar{x}_u = \frac{1}{2} \ \forall u \in V$ satisfies the constraints resulting a huge upper bound of at least $\dfrac{1}{2} \sum_{u \in V} x_u$.

A **clique** is a node subset inducing a complet subgraph.

Lemma

A stable set contains at most 1 node of a clique.

Then the inequality

$$\sum_{u \in K} x_u \leq 1 \quad \text{for each clique } K$$

is satisfied for every stable set !

The fractional point $\bar{x}_u = \frac{1}{2} \;\; \forall u \in V$ is **cut** by this inequality whenever G contains a triangle !
This inequality will lower down the relaxation value, which will be better for pruning into the B&B tree.
We say that this inequality **reinforce** the relaxation value of the formulation.

A **clique** is a node subset inducing a complet subgraph.

Lemma

A stable set contains at most 1 node of a clique.

Then the inequality

$$\sum_{u \in K} x_u \leq 1 \quad \text{for each clique } K$$

is satisfied for every stable set !

The fractional point $\bar{x}_u = \frac{1}{2} \ \forall u \in V$ is **cut** by this inequality whenever G contains a triangle !
This inequality will lower down the relaxation value, which will be better for pruning into the B&B tree.
We say that this inequality **reinforce** the relaxation value of the formulation.

Non-Compact formulation for the stable set problem

We then obtain an alternative formulation

$$\text{Max} \sum_{u \in V} w_u x_u$$

$$\begin{array}{ll} \sum_{u \in K} x_u \leq 1 & \text{for each clique } K \\ x_u \in \{0,1\} & \forall u \in V. \end{array}$$

There are an exponential number of cliques in a graph !
then this clique formulation is non-compact...

Non-Compact formulation for the stable set problem

We then obtain an alternative formulation

$$\text{Max} \sum_{u \in V} w_u x_u$$

$$\sum_{u \in K} x_u \leq 1 \quad \text{for each clique } K$$

$$x_u \in \{0,1\} \quad \forall u \in V.$$

There are an exponential number of cliques in a graph !
then this clique formulation is non-compact...

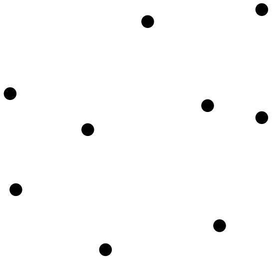
The Traveling Salesman Problem (TSP)

Input : **Directed** graph $G = (V, E)$
lenght $l_e \; \forall e \in E$

Output : An hamiltonian circuit C of G (i.e. C goes once through each noc

Objective : Min $\sum_{e \in C} l_e$

With a complete graph, it's a set of points to join with a circuit.



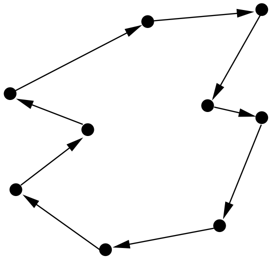
The Traveling Salesman Problem (TSP)

Input : **Directed** graph $G = (V, E)$
lenght $l_e \ \forall e \in E$

Output : An hamiltonian circuit C of G (i.e. C goes once through each noc

Objective : Min $\sum_{e \in C} l_e$

With a complete graph, it's a set of points to join with a circuit.



Natural variable formulation

$$x_{ij} = \begin{cases} 1 & \text{if arc } (i,j) \text{ is chosen} \\ 0 & \text{otherwise} \end{cases}$$

But this is not a TSP formulation !

$$\text{Min} \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} l_{ij} x_{ij}$$

$$\sum_{j \in V} x_{ij} = 1 \qquad \forall i \in V,$$

$$\sum_{i \in V} x_{ij} = 1 \qquad \forall j \in V,$$

$$x_{ij} \in N \qquad \forall i \in V, j \in V \setminus \{i\}).$$

Note we can add inequality $x_{ij} + x_{ji} \leq 1$.

Natural variable formulation

$$x_{ij} = \begin{cases} 1 & \text{if arc } (i,j) \text{ is chosen} \\ 0 & \text{otherwise} \end{cases}$$

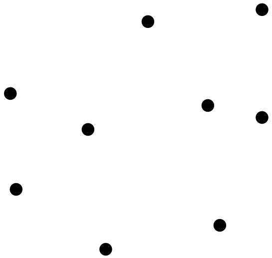
But this is not a TSP formulation !

$$\text{Min} \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} l_{ij} x_{ij}$$

$$\sum_{j \in V} x_{ij} = 1 \qquad \forall i \in V,$$

$$\sum_{i \in V} x_{ij} = 1 \qquad \forall j \in V,$$

$$x_{ij} \in N \qquad \forall i \in V, j \in V \setminus \{i\}).$$



Note we can add inequality $x_{ij} + x_{ji} \leq 1$.

Natural variable formulation

$$x_{ij} = \begin{cases} 1 & \text{if arc } (i,j) \text{ is chosen} \\ 0 & \text{otherwise} \end{cases}$$

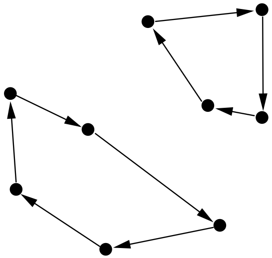
But this is not a TSP formulation !

$$\text{Min} \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} l_{ij} x_{ij}$$

$$\sum_{j \in V} x_{ij} = 1 \qquad \forall i \in V,$$

$$\sum_{i \in V} x_{ij} = 1 \qquad \forall j \in V,$$

$$x_{ij} \in N \qquad \forall i \in V, j \in V \setminus \{i\}).$$



Note we can add inequality $x_{ij} + x_{ji} \leq 1$.

Natural variable formulation

$$x_{ij} = \begin{cases} 1 & \text{if arc } (i,j) \text{ is chosen} \\ 0 & \text{otherwise} \end{cases}$$

But this is not a TSP formulation !

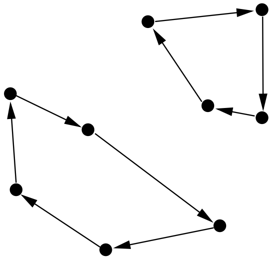
$$\text{Min} \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} l_{ij} x_{ij}$$

$$\sum_{j \in V} x_{ij} = 1 \qquad \forall i \in V,$$

$$\sum_{i \in V} x_{ij} = 1 \qquad \forall j \in V,$$

$$x_{ij} \in N \qquad \forall i \in V, j \in V \setminus \{i\}).$$

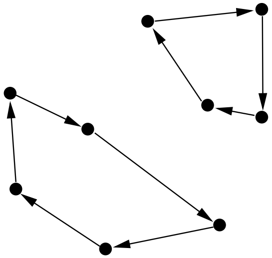
Note we can add inequality $x_{ij} + x_{ji} \leq 1$.



It's a formulation of the problem "cover a graph with circuits".

Formulation en variables naturelles

The relaxation of this MIP (which is not a TSP formulation) is “integer” as its extrem points are all integer (the matrix is totally unimodular) : this “covering a graph with circuit” problem is polynomial.

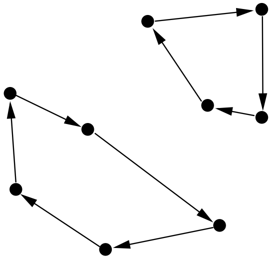


Unfortunately, since a solution of this MIP can be made up of several oriented cycles (called textcolorbluesubtours)

We then need to add inequalities to break sub-tours.

Formulation en variables naturelles

The relaxation of this MIP (which is not a TSP formulation) is “integer” as its extrem points are all integer (the matrix is totally unimodular) : this “covering a graph with circuit” problem is polynomial.



Unfortunately, since a solution of this MIP can be made up of several oriented cycles (called textcolorbluesubtours)

We then need to add inequalities to break sub-tours.

Breaking subtours by MTZ formulation

Miller-Tucker-Zemlin (MTZ) formulation
Adding real variables u_i, $i = 1, ..., n$ for each cities
Adding inequalities

$$\begin{aligned} u_1 &= 1, & \\ 2 \leq u_i &\leq n & \forall i \in V \setminus \{1\}, \\ u_i - u_j + 1 &\leq n(1 - x_{ij}) & \forall i \in V \setminus \{1\}, j \in V \setminus \{1, i\}. \end{aligned}$$

The latter inequalities are called **MTZ inequalities**.

Breaking subtours by MTZ formulation

Lemma

MTZ inequalities break subtours.

Proof. Let us consider an optimal (integer) solution of the MTZ formulation which then satisfies

$$u_i - u_j + 1 \leq n(1 - x_{ij}) \quad \forall (i,j) \text{ with } j \neq 1$$

- For an (i,j) with $x_{ij} = 0$, the MTZ inequalities are
$$u_i - u_j \leq n - 1$$
and are then always satisfied as since $1 \leq u_i \leq n$.
- For an arc (i,j) with $x_{ij} = 1$, the MTZ inequalities enforce
$$u_j \geq u_i + 1$$
- Hence, let us suppose there is a subtour not containing node 1, then the MTZ inequalities cannot be satisfied as variables u_i will then increase indefinitely !.
- Then there is only one subtour, which then is an hamiltonian tour □

Breaking subtours by MTZ formulation

The MTZ formulation is compact with additional n (continous) variables.

But, this is a **Very Very bad formulation !**

With a very low linear relaxation : the fractional solution $x_i = \frac{1}{n}$ is feasible and then the relaxation value is at most $\frac{\sum_{e \in E} l_e}{n}$ which is very high.

Breaking subtours by MTZ formulation

The MTZ formulation is compact with additional n (continous) variables.

But, this is a **Very Very bad formulation !**

With a very low linear relaxation : the fractional solution $x_i = \frac{1}{n}$ is feasible and then the relaxation value is at most $\frac{\sum_{e \in E} l_e}{n}$ which is very high.

Breaking subtours with flow formulation

Aggregating flow formulation
Add real flow variables z_{ij}, for each arc (i, j), $i \in V$, j$\in V \setminus \{1, i\}$.
Add the constraints :

$$\sum_{j \in V \setminus \{1\}} z_{1j} = |V| - 1$$

$$\sum_{j \in V \setminus \{1,i\}} z_{ij} + 1 = \sum_{j \in V \setminus \{i\}} z_{ji} \quad \forall i \in V \setminus \{1\},$$

$$z_{ij} \leq (|V| - 1) x_{ij} \quad \forall i \in V, j \in V \setminus \{1, i\}$$

$$z_{ji} \leq (|V| - 1) x_{ji} \quad \forall i \in V, j \in V \setminus \{1, i\}$$

$$z_{ij} \geq 0 \quad \forall i \in V, j \in V \setminus \{1, i\}.$$

Note that the flow variables $z_{i.}$ carry a flow of value $|V| - 1$ when it goes from node 1 and which is reduced by 1 unit at each node.

Breaking subtours with flow formulation

Lemma

Aggregating flow inequalities break subtours.

Proof. Let us consider an optimal (integer) MIP solution.
Let us suppose the graph corresponding to the x variables contains a subtour C which does not pass through node A. , which therefore contains at least 2 vertices. Note that this subtous contains at least two nodes and at most $|V| - 2$ vertices. On leaving 1, the flow $x_{i.}$ goes out from node 1 with value $|V| - 1$ and at each successive vertex of C, the flow decreases by one. Then the flow must decrease indefinitely, which is impossible. □

Breaking subtours with flow formulation

The aggregating flow formulation contains n^2 additional real variables, but it's still compact and can be solved directly by integer solver.

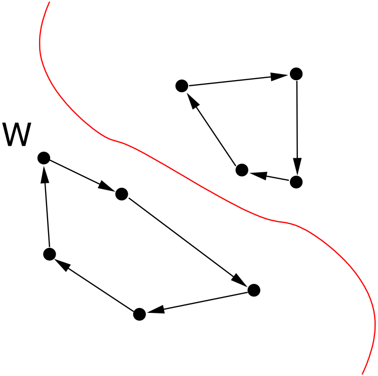
It's relaxation value is much better than the MTZ formulation but with the numerous additional variables and the “big M” constraints, it's still hard to solve large instance by B&B with it.

Subtour breaking through connectivity

Theorem (Menger)

A directed graph is strongly connected if and only if every graph cut contains at least one arc.

$$\sum_{e \in \delta^+(W)} x_e \geq 1, \forall W \subsetneq V \text{ et } W \neq \emptyset,$$



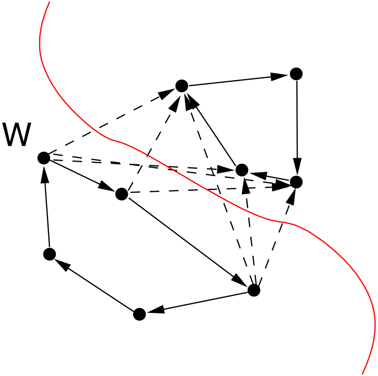
$\delta^+(W) = \{(i,j) \mid i \in W \text{ et } j \notin W\}$
is the cut going out from W.

Subtour breaking through connectivity

Theorem (Menger)

A directed graph is strongly connected if and only if every graph cut contains at least one arc.

$$\sum_{e \in \delta^+(W)} x_e \geq 1, \forall W \subsetneq V \text{ et } W \neq \emptyset,$$



$\delta^+(W) = \{(i,j) \mid i \in W \text{ et } j \notin W\}$
is the cut going out from W.

Subtour breaking through connectivity

Formulation by Menger cuts
Add the inequalities

$$\sum_{e \in \delta^+(W)} x_e \geq 1, \forall W \subsetneq V \text{ et } W \neq \emptyset,$$

As there is 2^n cuts in a graph, thee formulatuion is non-compact.
It contains **an exponential number of inequalities !**

However, this formulation have a real good relaxation value !
And it can be obtains through a Branch-and-Cut method than can be efficiently implemented.

Subtour breaking through connectivity

Formulation by Menger cuts
Add the inequalities

$$\sum_{e \in \delta^+(W)} x_e \geq 1, \forall W \subsetneq V \text{ et } W \neq \emptyset,$$

As there is 2^n cuts in a graph, thee formulatuion is non-compact.
It contains **an exponential number of inequalities !**

However, this formulation have a real good relaxation value !
And it can be obtains through a Branch-and-Cut method than can be efficiently implemented.

Menger cut formulation for the symmetric TSP

This is the "star" TSP formulation

$$Min \sum_{e \in E} l_e x_e$$

$$\sum_{e \in \delta(v)} x_e = 2 \text{ pour tout } u \in V,$$

$$\sum_{e \in \delta(W)} x_e \geq 2 \text{ pour tout } W \subsetneq V \text{ and } W \neq \emptyset,$$

$$x_e \in \{0, 1\} \text{ pour tout } e \in E.$$

With other reinforcement inequalities inside the Branch&Cut Concorde software, this formulation is the one which exactly solves TSP instances till 200 000 cities !

Some tricks to formulate some logical links between MIP variables.

• Some basic tricks

Let *a* ,*b* et *c* be some events corresponding to binary variables x_a, x_b et x_c.

- If *a* et *b* cannot happen at the same time : $x_a + x_b \leq 1$.
- If at least one event among *a* and *b* have to happen : $x_a + x_b \geq 1$.
- If *a* happens, then *b* must happen : $x_a \leq x_b$.
- Note that this inequaly also formulate the contraposed of the logical proposal, i.e. if *b* does not happel, then *a* must not happen.
- If *a* happen, then at least *b* and/or *c* must happen : $x_a \leq x_b + x_c$.

Some tricks to formulate some logical links between MIP variables.

• Some basic tricks

Let *a* ,*b* et *c* be some events corresponding to binary variables x_a, x_b et x_c.

- If *a* et *b* cannot happen at the same time : $x_a + x_b \leq 1$.
- If at least one event among *a* and *b* have to happen : $x_a + x_b \geq 1$.
- If *a* happens, then *b* must happen : $x_a \leq x_b$.
- Note that this inequaly also formulate the contraposed of the logical proposal, i.e. if *b* does not happel, then *a* must not happen.
- If *a* happen, then at least *b* and/or *c* must happen : $x_a \leq x_b + x_c$.

Some tricks to formulate some logical links between MIP variables.

• Some basic tricks

Let *a* ,*b* et *c* be some events corresponding to binary variables x_a, x_b et x_c.

- If *a* et *b* cannot happen at the same time : $x_a + x_b \leq 1$.
- If at least one event among *a* and *b* have to happen : $x_a + x_b \geq 1$.
- If *a* happens, then *b* must happen : $x_a \leq x_b$.
- Note that this inequaly also formulate the contraposed of the logical proposal, i.e. if *b* does not happel, then *a* must not happen.
- If *a* happen, then at least *b* and/or *c* must happen : $x_a \leq x_b + x_c$.

Some tricks to formulate some logical links between MIP variables.

• Some basic tricks

Let *a* ,*b* et *c* be some events corresponding to binary variables x_a, x_b et x_c.

- If *a* et *b* cannot happen at the same time : $x_a + x_b \leq 1$.
- If at least one event among *a* and *b* have to happen : $x_a + x_b \geq 1$.
- If *a* happens, then *b* must happen : $x_a \leq x_b$.
- Note that this inequaly also formulate the contraposed of the logical proposal, i.e. if *b* does not happel, then *a* must not happen.
- If *a* happen, then at least *b* and/or *c* must happen : $x_a \leq x_b + x_c$.

Some tricks to formulate some logical links between MIP variables.

• Some basic tricks

Let *a* ,*b* et *c* be some events corresponding to binary variables x_a, x_b et x_c.

- If *a* et *b* cannot happen at the same time : $x_a + x_b \leq 1$.
- If at least one event among *a* and *b* have to happen : $x_a + x_b \geq 1$.
- If *a* happens, then *b* must happen : $x_a \leq x_b$.
- Note that this inequaly also formulate the contraposed of the logical proposal, i.e. if *b* does not happel, then *a* must not happen.
- If *a* happen, then at least *b* and/or *c* must happen : $x_a \leq x_b + x_c$.

Some tricks to formulate some logical links between MIP variables.

• Some basic tricks

Let *a* ,*b* et *c* be some events corresponding to binary variables x_a, x_b et x_c.

- If *a* et *b* cannot happen at the same time : $x_a + x_b \leq 1$.
- If at least one event among *a* and *b* have to happen : $x_a + x_b \geq 1$.
- If *a* happens, then *b* must happen : $x_a \leq x_b$.
- Note that this inequaly also formulate the contraposed of the logical proposal, i.e. if *b* does not happel, then *a* must not happen.
- If *a* happen, then at least *b* and/or *c* must happen : $x_a \leq x_b + x_c$.

Some tricks to formulate some logical links between MIP variables.

• Some basic tricks

Let *a* ,*b* et *c* be some events corresponding to binary variables x_a, x_b et x_c.

- If *a* et *b* cannot happen at the same time : $x_a + x_b \leq 1$.
- If at least one event among *a* and *b* have to happen : $x_a + x_b \geq 1$.
- If *a* happens, then *b* must happen : $x_a \leq x_b$.
- Note that this inequaly also formulate the contraposed of the logical proposal, i.e. if *b* does not happel, then *a* must not happen.
- If *a* happen, then at least *b* and/or *c* must happen : $x_a \leq x_b + x_c$.

Some tricks to formulate some logical links between MIP variables.

• Some basic tricks

Let *a* ,*b* et *c* be some events corresponding to binary variables x_a, x_b et x_c.

- If *a* et *b* cannot happen at the same time : $x_a + x_b \leq 1$.
- If at least one event among *a* and *b* have to happen : $x_a + x_b \geq 1$.
- If *a* happens, then *b* must happen : $x_a \leq x_b$.
- Note that this inequaly also formulate the contraposed of the logical proposal, i.e. if *b* does not happel, then *a* must not happen.
- If *a* happen, then at least *b* and/or *c* must happen : $x_a \leq x_b + x_c$.

Some tricks to formulate some logical links between MIP variables.

• Some basic tricks

Let *a* ,*b* et *c* be some events corresponding to binary variables x_a, x_b et x_c.

- If *a* et *b* cannot happen at the same time : $x_a + x_b \leq 1$.
- If at least one event among *a* and *b* have to happen : $x_a + x_b \geq 1$.
- If *a* happens, then *b* must happen : $x_a \leq x_b$.
- Note that this inequaly also formulate the contraposed of the logical proposal, i.e. if *b* does not happel, then *a* must not happen.
- If *a* happen, then at least *b* and/or *c* must happen : $x_a \leq x_b + x_c$.

- Link between binary and continuous variables

Let *a* an event corresponding to a binary variable x_a and a continuous real quantity y.
If *a* does not happen , then a positive quantity y must be zero, otherwise y is free.
To do this : we must fix a quantity M such that y can never be greater than M when the optimum of the problem is reached. Such a constant M exists, since otherwise the problem would be unbounded.

$$y \leq Mx_a$$

Such an inequality is called a “big M” constraint.
Such constraints are often mandatory to formulate a problem. However they deeply impact the problem with a bad numerical behavior. Indeed, during the B&B exploeation tree exploration when $0 < x_a < 1$, then y can have an unrealistic value (generally really low one) which produces a really unapropriate relaxation value.

- Link between binary and continuous variables

Let a an event corresponding to a binary variable x_a and a continuous real quantity y.
If a does not happen , then a positive quantity y must be zero, otherwise y is free.
To do this : we must fix a quantity M such that y can never be greater than M when the optimum of the problem is reached. Such a constant M exists, since otherwise the problem would be unbounded.

$$y \leq Mx_a$$

Such an inequality is called a "big M" constraint.
Such constraints are often mandatory to formulate a problem. However they deeply impact the problem with a bad numerical behavior. Indeed, during the B&B exploeation tree exploration when $0 < x_a < 1$, then y can have an unrealistic value (generally really low one) which produces a really unapropriate relaxation value.

• Link between binary and continuous variables

Let a an event corresponding to a binary variable x_a and a continuous real quantity y.
If a does not happen , then a positive quantity y must be zero, otherwise y is free.
To do this : we must fix a quantity M such that y can never be greater than M when the optimum of the problem is reached. Such a constant M exists, since otherwise the problem would be unbounded.

$$y \leq Mx_a$$

Such an inequality is called a “big M” constraint.
Such constraints are often mandatory to formulate a problem. However they deeply impact the problem with a bad numerical behavior. Indeed, during the B&B exploeation tree exploration when $0 < x_a < 1$, then y can have an unrealistic value (generally really low one) which produces a really unapropriate relaxation value.

• The numerical “or”

We want to represent a variable x which must take values either 0 or greater than L, where L and x are bounded by a value M.
We need to add a binary variable $y \in \{0,1\}$ and use the constraints

$$x \geq Ly \qquad \text{et} \qquad x \leq My.$$

• The numerical “or”

We want to represent a variable x which must take values either 0 or greater than L, where L and x are bounded by a value M.
We need to add a binary variable $y \in \{0, 1\}$ and use the constraints

$$x \geq Ly \qquad \text{et} \qquad x \leq My.$$

• *Multi-objective Min-Max regret*

We want to maximize the minimal value of a set of linear values

Let a set of linear values a^1x, a^2x, a^3x..., a^mx
They can be several objective function correspondinf to different agents.

We need to add a continuous variable z and m inequalities

$$\begin{aligned} &\text{Max } z \\ &z \leq a^1x \\ &z \leq a^2x \\ &z \leq a^3x \\ &\quad ... \\ &z \leq a^mx \end{aligned}$$

• *Multi-objective Min-Max regret*

We want to maximize the minimal value of a set of linear values

Let a set of linear values a^1x, a^2x, a^3x..., a^mx
They can be several objective function correspondinf to different agents.

We need to add a continuous variable z and m inequalities

$$\begin{array}{c} \text{Max } z \\ z \leq a^1x \\ z \leq a^2x \\ z \leq a^3x \\ \ldots \\ z \leq a^mx \end{array}$$

• *Satisfy the most possible inequalities*

Let a set of n constraints $a^1x \leq b^1, a^2x \leq b^2, ..., a^nx \leq b^n$ be such that there no solution satisfy all the inequalities.
We want a solution that satisfies as many constraints as possible.
For each of the constraints $a^ix \leq b^i$, $i = 1, ..., n$, we determine a value M_i large enough for $a^ix \leq b^i + M_i$ to be satisfied whatever x.

We add the binary variables $y_1, ..., y_n$ so that $y_i = 0$ if the inequality i is satisfied.

$$\begin{array}{lcl} a^1x \leq b^1 & & a^1x \leq b^1 + M_1y_1 \\ a^2x \leq b^2 & & a^2x \leq b^2 + M_2y_2 \\ ... & \Rightarrow & ... \\ a^nx \leq b^n & & a^nx \leq b^n + M_ny_n \\ & & \\ & & Min \sum_{i=1}^{n} y_i \end{array}$$

• *Satisfy the most possible inequalities*

Let a set of n constraints $a^1x \leq b^1, a^2x \leq b^2, ..., a^nx \leq b^n$ be such that there no solution satisfy all the inequalities.
We want a solution that satisfies as many constraints as possible.
For each of the constraints $a^ix \leq b^i$, $i = 1, ..., n$, we determine a value M_i large enough for $a^ix \leq b^i + M_i$ to be satisfied whatever x.

We add the binary variables $y_1, ..., y_n$ so that $y_i = 0$ if the inequality i is satisfied.

$$\begin{array}{lcl} a^1x \leq b^1 & & a^1x \leq b^1 + M_1y_1 \\ a^2x \leq b^2 & & a^2x \leq b^2 + M_2y_2 \\ ... & \Rightarrow & ... \\ a^nx \leq b^n & & a^nx \leq b^n + M_ny_n \end{array}$$

$$Min \sum_{i=1}^{n} y_i$$

• *Satisfy the most possible inequalities*

Let a set of n constraints $a^1x \leq b^1, a^2x \leq b^2, ..., a^nx \leq b^n$ be such that there no solution satisfy all the inequalities.
We want a solution that satisfies as many constraints as possible.
For each of the constraints $a^ix \leq b^i$, $i = 1, ..., n$, we determine a value M_i large enough for $a^ix \leq b^i + M_i$ to be satisfied whatever x.

We add the binary variables $y_1, ..., y_n$ so that $y_i = 0$ if the inequality i is satisfied.

$$\begin{array}{lcl} a^1x \leq b^1 & & a^1x \leq b^1 + M_1y_1 \\ a^2x \leq b^2 & & a^2x \leq b^2 + M_2y_2 \\ ... & \Rightarrow & ... \\ a^nx \leq b^n & & a^nx \leq b^n + M_ny_n \\ & & \\ & & Min \sum_{i=1}^{n} y_i \end{array}$$

To conclude

A combinatorial optimization problem has many formulations :
- compact
- with an exponential number of constraints
- with an exponential number of variables
- with an exponential number of variables and constraints !

How do you choose a good formulation ?
It depends on the algorithmic framework of resolution : there are no generic "magic" tools for all these formulations and all the problems...

How to obtain a better relaxation value ?
Add reinforcement inequalities !
How to solve a non-compact formulation ?
Use Branch&Cut method !
What are the powerfulest inequalities
Make a polyhedral study !

To conclude

A combinatorial optimization problem has many formulations :
- compact
- with an exponential number of constraints
- with an exponential number of variables
- with an exponential number of variables and constraints !

How do you choose a good formulation ?
It depends on the algorithmic framework of resolution : there are no generic "magic" tools for all these formulations and all the problems...

How to obtain a better relaxation value ?
    Add reinforcement inequalities !
How to solve a non-compact formulation ?
    Use Branch&Cut method !
What are the powerfulest inequalities
    Make a polyhedral study !