Decision Aid Methodologies In Transportation
Lecture 4: MILP

Mixed Linear

Integer Programming

Shadi SHARIF AZADEH

Transport and Mobility Laboratory TRANSP-OR

École Polytechnique Fédérale de Lausanne EPFL

Linear Integer Programming

$$\begin{aligned} \min : \quad & \mathbf{c}'\mathbf{x} \\ s.t. \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \\ & x_i \in \mathbb{Z}, \forall i \in \mathcal{I} \end{aligned}$$

Suppose $x \in R^n$. If $|I| = n$, the integer programming problem is called pure integer programming problem; otherwise, if $|I| \leq n$, the problem is called mixed integer programming (MIP). MIP have extremely wide applications in practice.

The relaxation of integer programming problem

The linear relaxation problem of an integer programming problem is very crucial since it provides useful bound information for the integer one.

$$\begin{aligned} \min: \quad & \mathbf{c}'\mathbf{x} \\ s.t. \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \\ & x_i \in \mathbb{Z}, \forall i \in \mathcal{I} \end{aligned} \qquad \begin{aligned} \min: \quad & \mathbf{c}'\mathbf{x} \\ s.t. \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Question: if the sense of the objective function is minimization, the optimal value of the linear relaxation problem is a/an (lower/upper) bound of the integer programming problem?

ILP characteristics

Model characteristics:

- All constraints and objective are linear
- Some or all of variables are integer

ILP (binary variables)

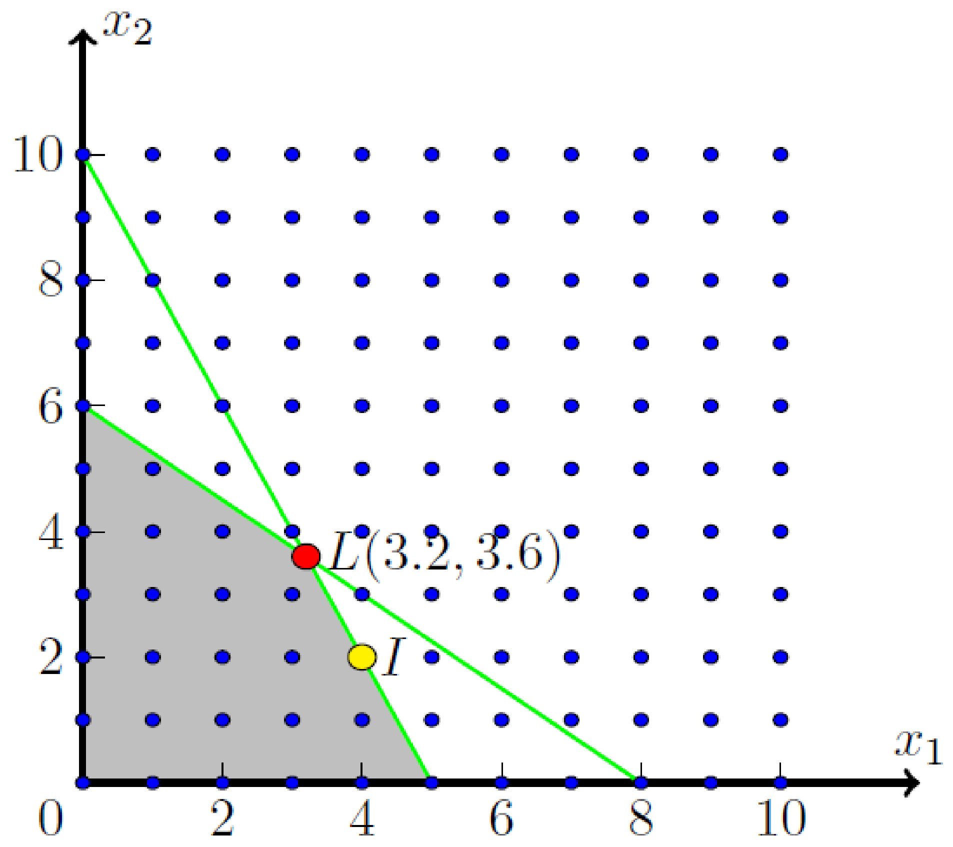
$$\begin{aligned} &\max \quad cx \\ &st. \\ &\quad Ax \le b \\ &\quad x \in \{0,1\} \end{aligned}$$

MILP

$$\begin{aligned} &\max \quad cx + hy \\ &st. \\ &\quad Ax + Gy \le b \\ &\quad x \ge 0 \ and \ \ Integer, y \ge 0 \end{aligned}$$

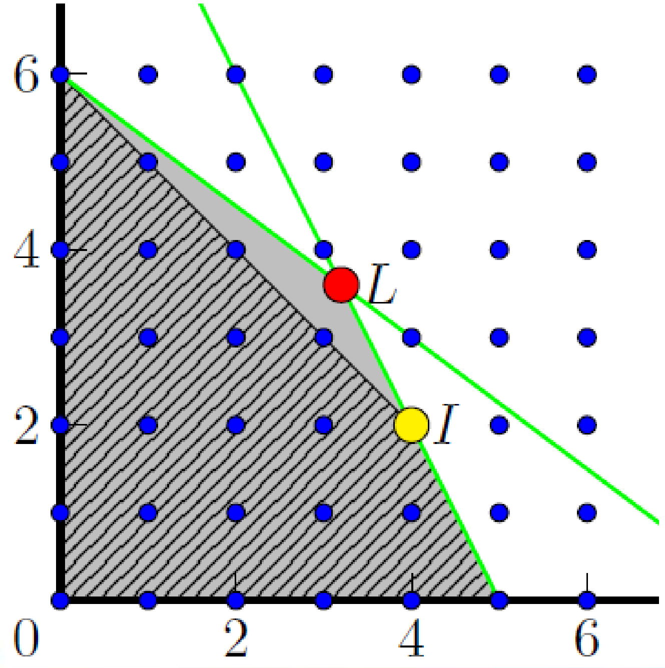
The feasible region of an ILP

$$\max\{7x_1+5x_2 \,|\, 2x_1+x_2 \leq 10, 3x_1+4x_2 \leq 24, x_1, x_2 \geq 0, x_1, x_2 \in \mathbb{Z}\}$$



The feasible region of an ILP

The hatched region is called the convex hull of the feasible region of the integer programming problem. The convex hull has an very important property:
Let $P = \{Ax = b, x \geq 0, x_i \in Z, \forall i \in I\}$ and denote the convex hull for P as CH(P); then the optimal solution of an integer programming problem is exactly the same as its linear relaxation problem under the feasible region CH(P).



Well Known Integer Programming Problems

Unfortunately, how to obtain CH(P) for an integer programming problem is an extremely hard problem (i.e., NP-hard (Non-deterministic Polynomial-time **hard)**).

Here are a list of some of the well known ILPs:

1. The assignment problem
2. The 0-1 knapsack problem
3. Set covering problem
4. Facility location problem
5. Traveling salesman problem

The assignment problem

There are n people available to carry out n jobs. Each person is assigned to carry out exactly one job. Some individuals are better suited to particular jobs than others, so there is an estimated cost C_{ij} if person i is assigned to job j.

The goal is to find a minimum cost assignment

Sets:

$i \in I$ Person

$j \in J$ Job

Parameters:

c_{ij} : The cost associated to assining person i to job j

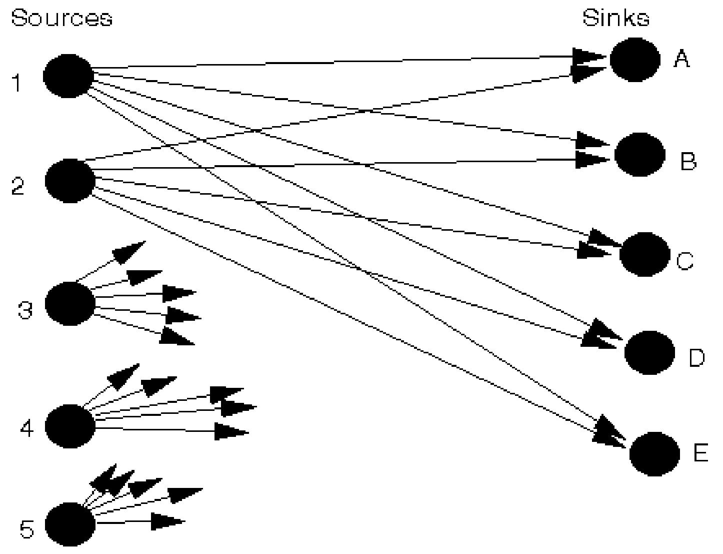
Variables:

$x_{ij} = 1$ if person i is assigned to job j, 0 otherwise

$$\text{Min} \sum_i \sum_j c_{ij} x_{ij}$$

$$\sum_i x_{ij} = 1 \quad \forall j \in J$$

$$\sum_j x_{ij} = 1 \quad \forall i \in I$$



The assignment problem

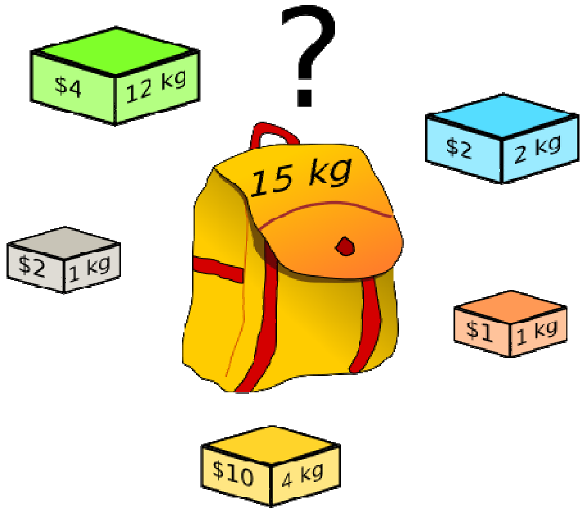
Assignement problem formulation allows for continuous values of decision variables. However, there is always an optimal solution with integer values.

Unimodularity of the matrix of coefficients:
An integer matrix is totally unimodular if the determinant of every square submatrix has value 0, +1, or −1. Unimodular matrix is a square integer matrix with determinant of +1 or -1.

The 0-1 knapsack problem

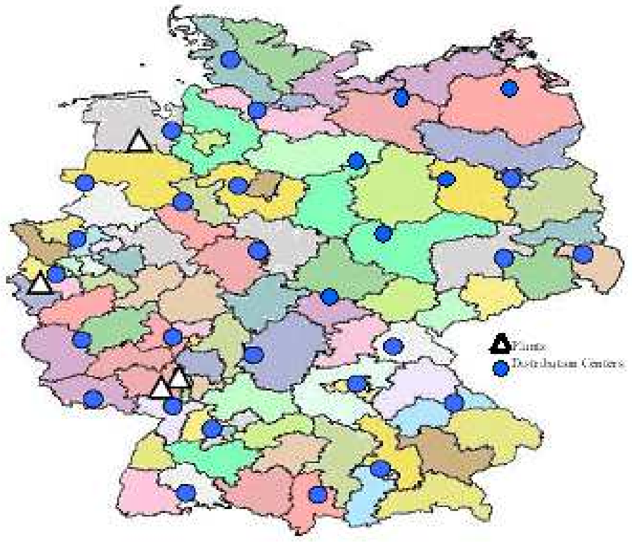
We are given n items. The j^{th} item has weight w_j (parameter) and its corresponding gain is presented by c_j (parameter.) Given a bound K on the weight that can be carried in a knapsack, we would like to select items to maximize the total benefit. We define a binary decision variable x_j which is 1 if item j is chosen, and 0 otherwise. The knapsack problem can then be formulated as:

$$
\begin{aligned}
\max : \quad & \sum_{j=1}^{n} c_j x_j \\
s.t. \quad & \sum_{j=1}^{n} w_j x_j \leq K \\
& x_j \in \{0,1\}, \forall j = 1, \ldots, n
\end{aligned}
$$



Set covering problem

Given a number of regions, we decide where to install for example a set of emergency service centers. The cost of installing a service center and to the region to which it serves are known. e.g., if the center is a fire station, a station can serve those regions for which rescue is guaranteed to arrive to the scene within 8 minutes. **The goal is to choose a minimum cost set of service centers so that each region is covered.**



Sets:

$i \in I$ Regions

$j \in J$ Service center

Parameters:

$$a_{ij} : \begin{cases} 1 & \text{if region i is covered by center j} \\ 0 & \text{otherwise} \end{cases}$$

c_j : The cost of installing a service center j

Variables:

$x_j = 1$ if center j is installed, 0 otherwise

$$\text{Min} \sum_j c_j x_j \qquad \forall j \in J$$

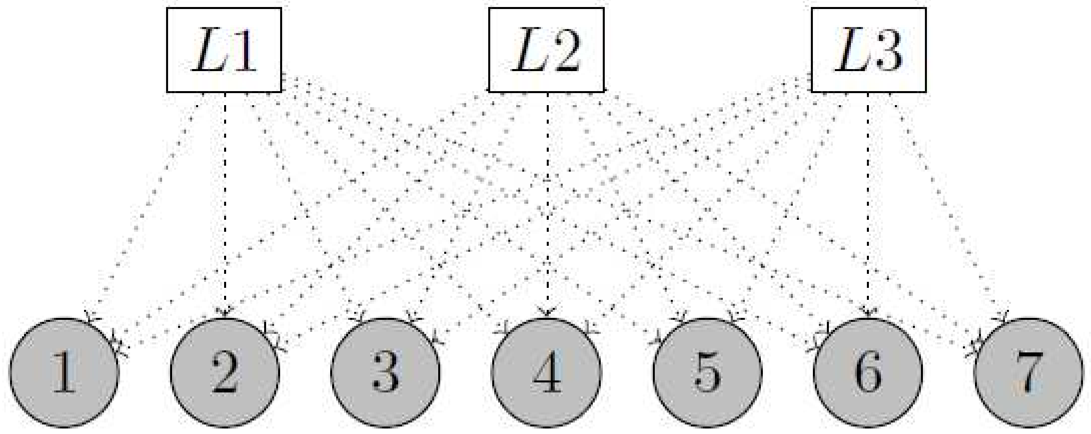
$$\sum_j a_{ij} x_j \geq 1 \quad \forall i \in I$$ i.e. Region i is covered at least by one of the centers.

Facility location problem

Suppose we are given n potential facility locations and a list of m clients who need to be served from these locations. There is a fixed cost c_j of opening a facility at location j, while there is a cost d_{ij} of serving client i from facility j. The goal is to select a set of facility locations and assign each client to one facility , while minimizing the total cost.

Facility

Clients



Facility location problem

We define a binary decision variable y_j which is 1 if facility j is opened, and 0 otherwise. In addition, we define another binary variable x_{ij} , which is 1 if client i is served by facility j, and 0 otherwise. The facility location problem can be formulated as follows:

$$\max : \quad \sum_{j=1}^{n} c_j y_j + \sum_{i=1}^{m} \sum_{j=1}^{n} d_{ij} x_{ij}$$

$$s.t. \quad \sum_{j=1}^{n} x_{ij} = 1, \forall i$$

Each client is served by exactly one facility center

$$x_{ij} \le y_j, \forall i, j$$

If client i is served by facility j, then j has to be open.
If facility j is closed then it can serve no one.

$$x_{ij}, y_j \in \{0, 1\}, \forall i, j$$

Facility location problem

The power of binary decision variables: To model the requirement of "at least one/exactly one/at most one".

At least one: $\sum_{j=1}^{n} x_j \geq 1$	Each client is served by at least one facility center
Exactly one: $\sum_{j=1}^{n} x_j = 1$	Each client is served by exactly one facility center
At most one: $\sum_{j=1}^{n} x_j \leq 1$	Each client is served by at most one facility center

In the facility location problem, we have used the "exactly one" type constraints.

Question: in the facility location problem, if we impose another requirement -- at most 2 locations can be chosen, then how to model?

Traveling salesman problem (TSP)

A salesman must visit each of n cities exactly once and then return to his starting point. The time taken to travel from city i to city j is c_{ij}.
Objective: Find the order in which he should make his tour so as to finish as quickly as possible.

Sets:

$i, j \in V$ City

Parameters:

c_{ij} : Traveling time between cities i and j

Variables:

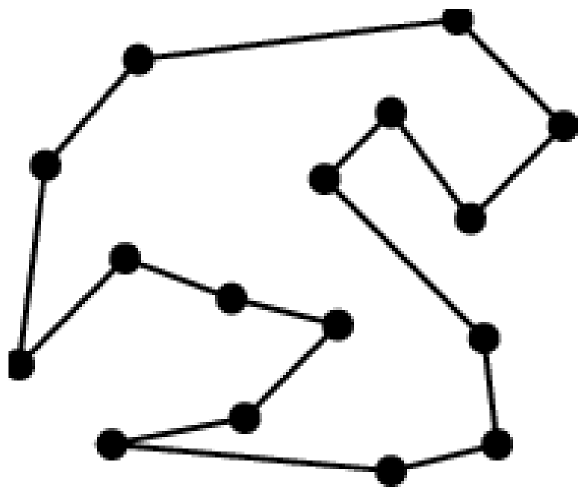
$x_{ij} = 1$ if person travels from i to j, 0 otherwise

$$\text{Min} \sum_i \sum_j c_{ij} x_{ij}$$

$$\sum_i x_{ij} = 1 \quad \forall j \in V$$

$$\sum_j x_{ij} = 1 \quad \forall i \in V$$

All nodes have exactly one entering and one exiting arcs to make sure all cities have been visited



Is this model correctly represent the problem?

Traveling salesman problem (TSP)

Sub-tour elimination constraints form:

$$\sum_{i,j \in S} x_{ij} \leq |S| - 1 \qquad 2^n - 2 \quad \text{Subtour Elimination Constraints}$$

Explanation:

Let $S \subset \{1, 2, 3, \ldots, n\}$ and $S \subset V, S \neq 0$

example: Suppose $S = \{1, 4, 5\}$

$x_{14} + x_{41} + x_{15} + x_{51} + x_{45} + x_{54}$

$0 + 1 + 1 + 0 + 0 + 1 \leq 3 - 1$ false

Some modeling tips: If-Then Decision and the big M

If-then decision: For example: If $f(x) \geq 0$ then $g(x) \geq 0$
To mathematically express the above “if-then" decision, we can introduce an auxiliary binary decision variable y. The meaning of y is

$$y = \begin{cases} 1, & \text{if } f(x) \geq 0 \\ 0, & \text{otherwise.} \end{cases}$$

To ensure such a relationship, let M be a sufficiently large (small) positive number.

$$\begin{aligned} f(x) &\leq M \cdot y \\ f(x) &\geq M(y-1) \end{aligned}$$

Then to enforce that if $f(x) \geq 0$ then $g(x) \geq 0$, we also need to add the following constraint:

$$g(x) \geq M(y-1)$$

The above method is usually called big-M method. Pro and Con of the big-M method:

Advantage: maintain the linearity of constraints
Disadvantage: adverse effect on the bound quality of the linear relaxation problem if tight M is not used

Some modeling tips: If-Then Decision and the big M

<u>Advantages of the big-M method:</u>

Consider the following question:
Let x be the amount of useful knowledge that you will receive from this decision-aid methodology course.

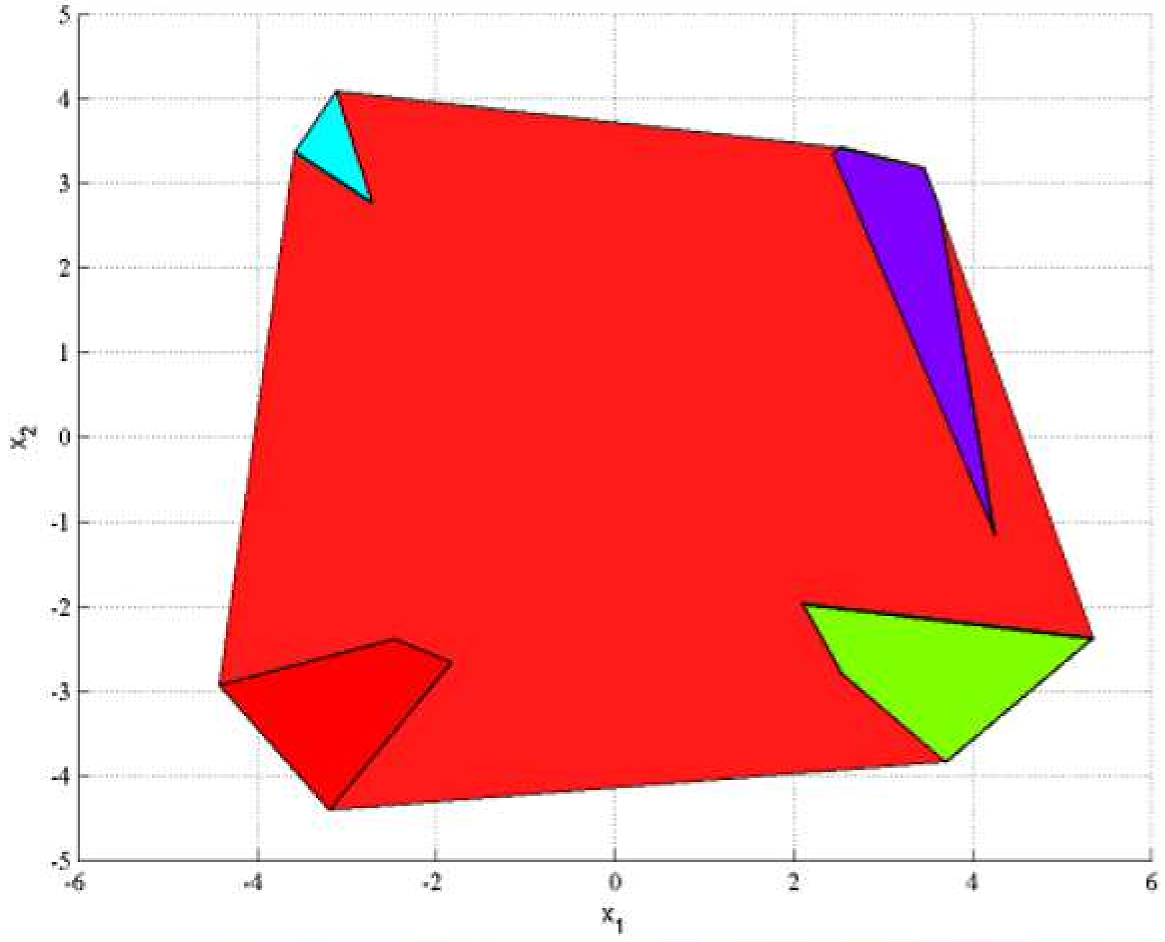
- Let $y = 1$ if the lecturer is good; 0, otherwise.
- Let $z = 1$ if you attend the course; 0, otherwise.
- Let $w = 1$ if you attend the laboratory; 0, otherwise.

Write the constraint stating that if the lecturer is good and you attend the course and the laboratory, then the amount of knowledge you will learn from this course should be greater than K.

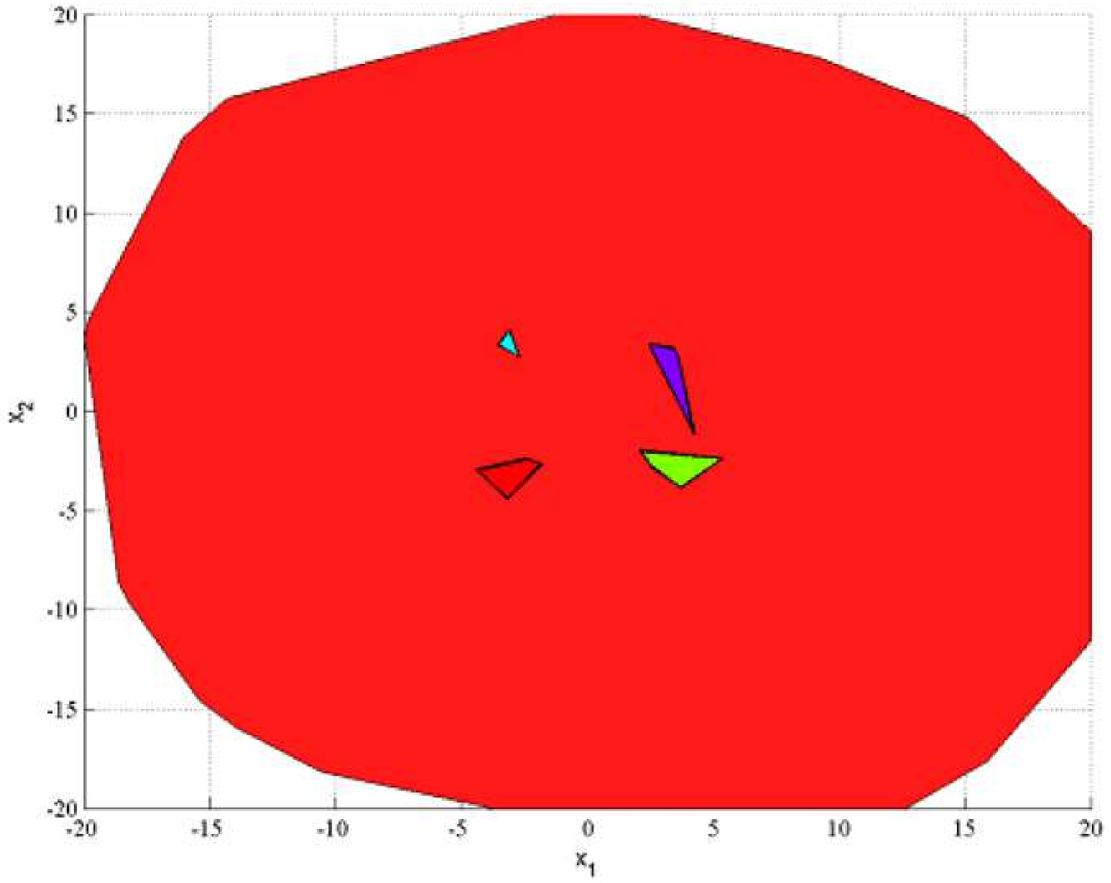
- Some people will choose the constraint: $x \geq K.y.z.w$
- By using big-M method, $x \geq K - M(3 - y - z - w)$

Some modeling tips: If-Then Decision and the big M

Impact of the choice of big-M:



Some modeling tips: If-Then Decision and the big M

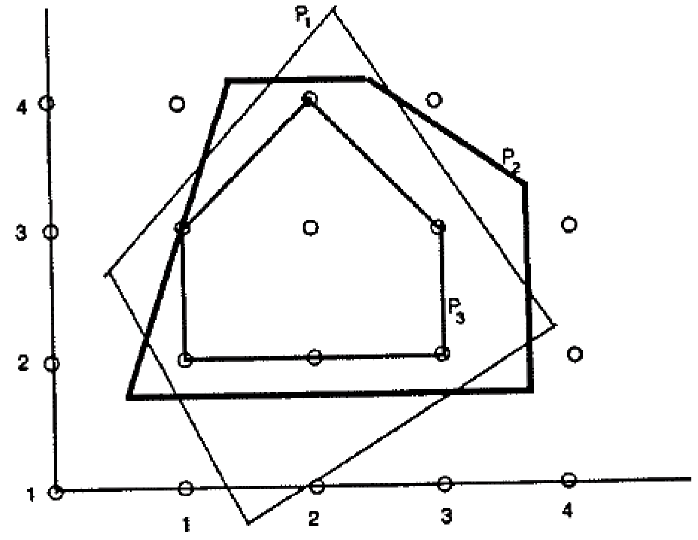


Some modeling tips: What is a good formulation

$$P_1 = \{ \quad 83x_1 + 61x_2 + 49x_3 + 20x_4 \leq 100\}$$

$$P_2 = \{ \quad 4x_1 + 3x_2 + 2x_3 + x_4 \leq 4\}$$

$$P_3 = \left\{ \begin{array}{c} 4x_1 + 3x_2 + 2x_3 + x_4 \leq 4 \\ 1x_1 + 1x_2 + 1x_3 \leq 1 \\ 1x_1 + 1x_4 \leq 1 \end{array} \right\}$$



$$X = \{(0,0,0,0\}, (1,0,0,0), (0,1,0,0), (0,0,1,0), (0,0,0,1), (0,1,0,1), (0,0,1,1)\}$$

P3 is ideal, because now if we solve a linear program over P3, the optimal solution is at an extreme point. In this case the corner points are integer and so the IP is solved

Some modeling tips: What is a good formulation

Given a set $X \subseteq R^n$, and two formulations P_1 and P_2 for X, P_1 is a better formulation than P_2 if $P_1 \subset P_2$. That is all the integer points inside P_2 belong to P_1.

Equivalent formulation for a knapsack set

Looking again at the set

$X = \{(0,0,0,0\}, (1,0,0,0), (0,1,0,0), (0,0,1,0), (0,0,0,1), (0,1,0,1), (0,0,1,1)\}$

At the three formulations,

$P_1 = \{x \in B^4\ 83x_1 + 61x_2 + 49x_3 + 20x_4 \leq 100\}$

$P_2 = \{x \in B^4 : 4x_1 + 3x_2 + 2x_3 + x_4 \leq 4\}$

$$P_3 = \left\{ \begin{matrix} x \in B^4 :\ 4x_1 + 3x_2 + 2x_3 + x_4 \leq 4 \\ 1x_1 + 1x_2 + 1x_3 \leq 1 \\ 1x_1 + 1x_4 \leq 1 \\ \end{matrix} \right\}$$

It is easily seen that $P_3 \subset P_2 \subset P_1$and it can be checked that $P_3 = conv(X)$, and $P_3$is thus an ideal formulation.

TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Relaxation, bounds and optimality

How is it possible to prove that a given point $X^*$is optimal?

Fast reply:

We could use an algorithm that will find a decreasing sequence of upper bound and increasing sequence of lower bound and stop when the difference between two sequences is negligible.

Primal bounds:

Every feasible solution x^*in X provides a lower bound. (maximization problem)

For some problem finding feasible solution may be very difficult to obtain.

Dual bounds:

Finding upper bound for a maximization problem (lower bound for minimization problem)

Approach:

Relaxation: replace a difficult max “min” IP by simpler optimization whose optimal value is at least as large (small) as Z.

Relaxation, bounds and optimality

Relaxation possibilities:

1. Enlarge the set of feasible solution so that one optimizes over a large set (relax variable type)
2. Replace the objective function by a function that has the same or a larger value everywhere (relax constraints) such as TSP without sub-tour elimination.

Relaxation, bounds and optimality

Example:TSP

Relaxing variable type: all variables are fractional.
Relaxing sub-tour elimination constraints (That is, TPS reduces to an assignment problem). Variable X is relaxed in this case.

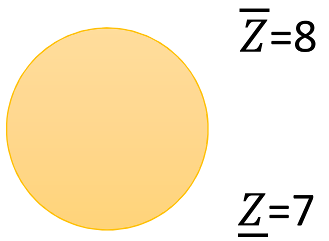
$$
\begin{array}{llcll}
\min & \sum_{(i,j)\in E} c_{ij}x_{ij} & & & \\
\text{s.t.} & \sum_{j\in V} x_{ij} & = & 1 & \forall i \in V \\
 & \sum_{i\in V} x_{ij} & = & 1 & \forall j \in V
\end{array}
$$

Relaxation, bounds and optimality

Why sensitivity analysis important in practice ?

$$\begin{array}{lrcrcr} \max & 4x_1 & - & x_2 & & \\ & 7x_1 & - & 2x_2 & \leq & 14 \\ & & & x_2 & \leq & 3 \\ & 2x_1 & - & 2x_2 & \leq & 3 \\ & x \in Z_+^2. & & & & \end{array}$$

To obtain a primal (lower) bound, observe that (2,1) is a feasible solution, so we have the lower bound $z \geq 7$. To obtain an upper bound, consider the linear programming relaxation. The optimal solution is $x^* = \left(\frac{20}{7}, 3\right)$ with value $z^{LP} = \frac{59}{7}$. Thus, we obtain an upper bound $z \leq \frac{59}{7}$. Observing that the optimal value must be integer, we can round down to the nearest integer and so obtain $z \leq 8$.



Optimal solution is between 7 and 8 , 7<Z<8
As the coefficients are integer → it is either
7 or 8
Z:{7,8}

Relaxation, bounds and optimality

The impact of better formulation:
The definition of better formulation is intimately related to that of linear programming relaxations. In particular better formulations gives tighter bound

Better model → better relaxation → finding optimal solution is faster

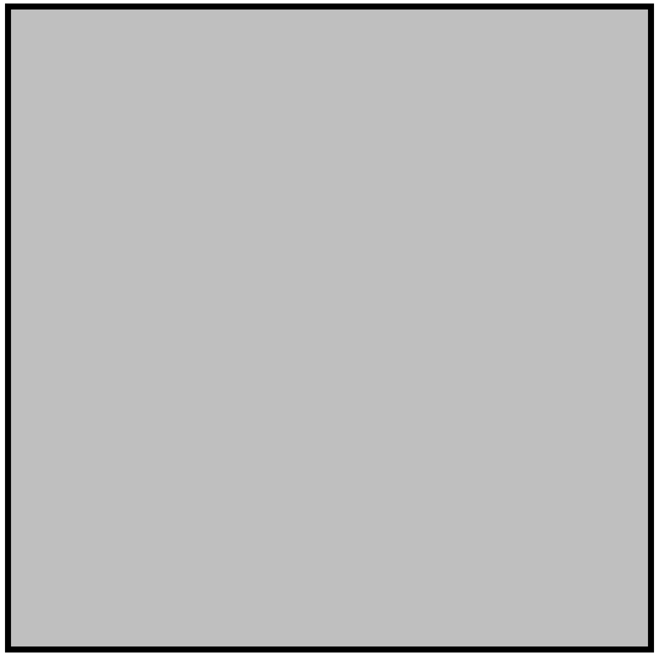
Proposition.
If a relaxation RP is infeasible, the original problem IP is infeasible.

Branch and Bound

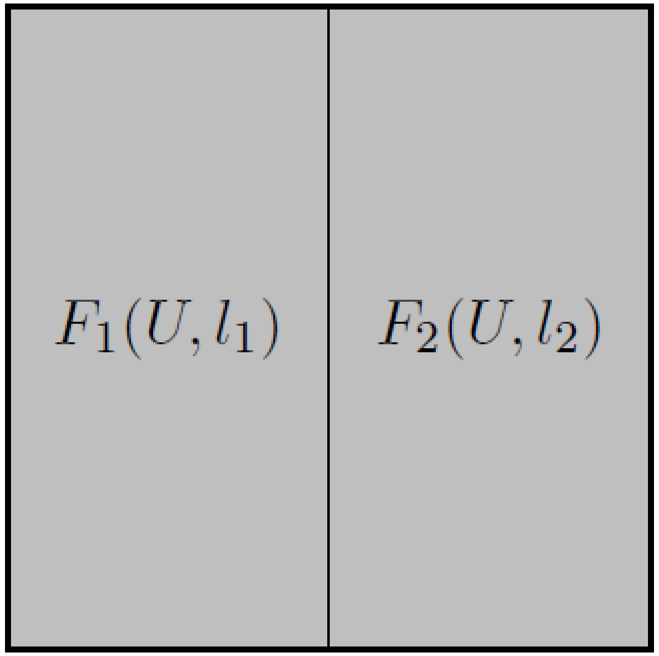
Idea: How we can break the problem into a series of smaller problems that are easier to solve the smaller problems, and then put the information together again to solve the original problem.

Method: enumeration tree

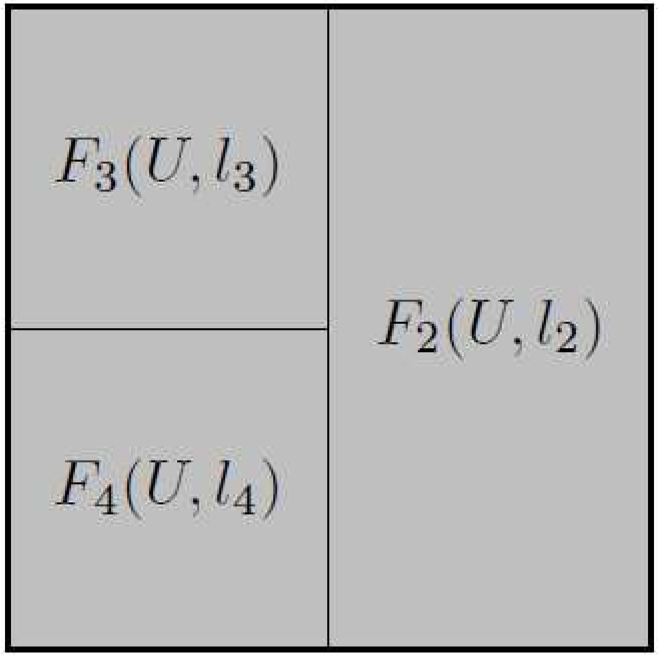
Branch and bound



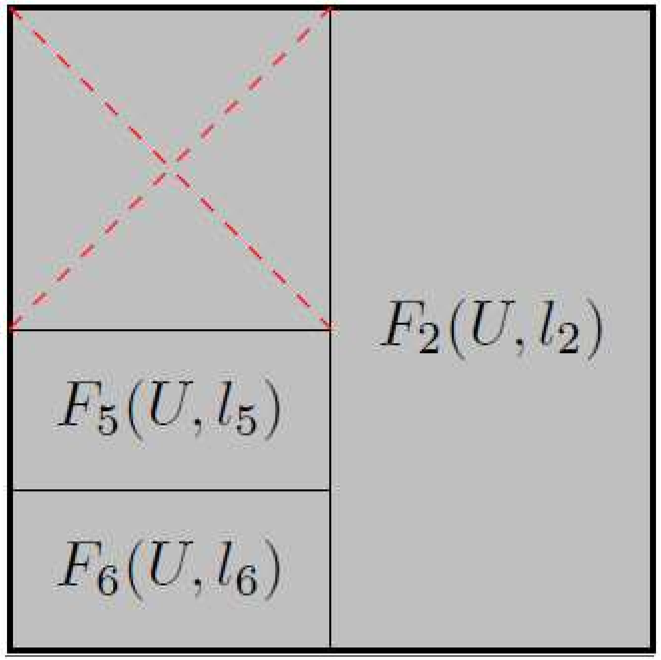
Branch and bound



Branch and bound



Branch and bound


$F_2(U, l_2)$
$F_5(U, l_5)$
$F_6(U, l_6)$