Mathematical programming: ILP/MILP modeling by continuous, binary and integer variables

ITEE PhD Course,
Lecture 3 – From observability to privacy and security in discrete event systems
September 15th, 2020

Claudio Sterle

Associate Professor in Operations Research,
Department of Electrical Engineering and Information Technology (http://www.dieti.unina.it)
OPSLab, Optimization and Problem Solving Laboratory (http://opslab.dieti.unina.it)
University ‘Federico II’ of Naples (http://www.unina.it)
claudio.sterle@unina.it

Outline of the lecture

Operations Research & Decision Problems & Mathematical Programming

1) Modeling with binary variables
   - Do-don’t do decisions
   - Logical conditions
   - Products of binary variables
   - Dichotomies

2) Binary variables do everything
   - General integer
   - Semi-continous
   - Partial integer
   - Semi-ordered set type

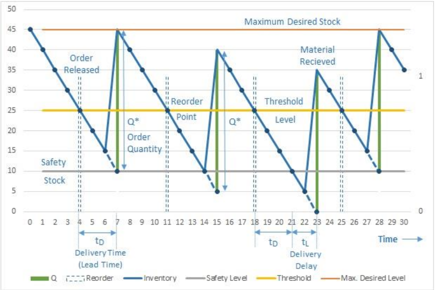
3) Connecting real variables to binary variables
   - Modeling fixed costs
   - Counting
   - Partial integers again
   - Price breaks and economies of scale
   - The product of a binary and a real variable

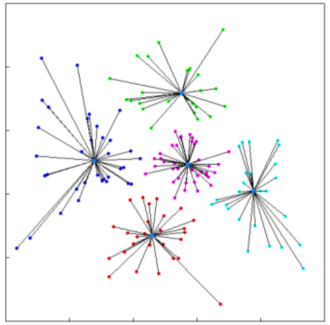
4) ILP and MILP samples

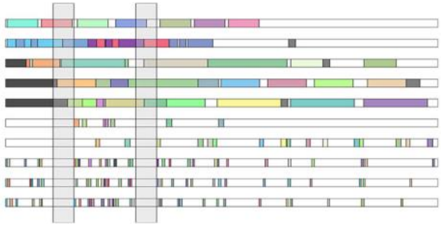
Operations research (operational research) (OR)

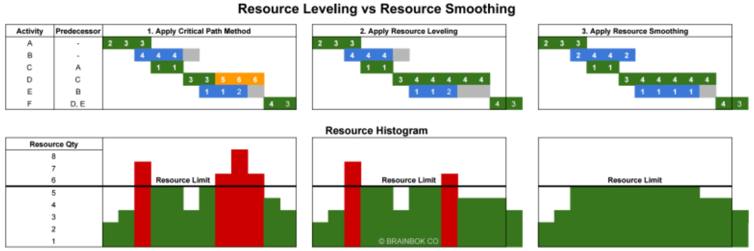
Discipline dealing with the application of advanced analytical methods to help make better decisions and solve complex decision problems. The terms **management science** and **decision sciences** are sometimes used as synonyms.

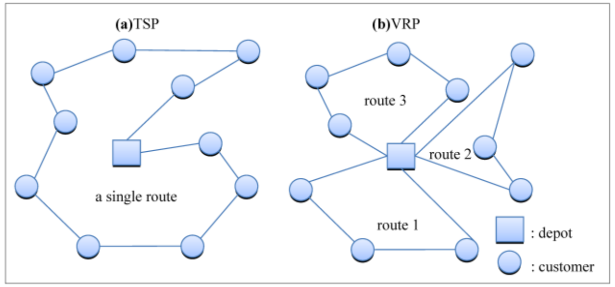
- Production Planning
- Inventory Management
- Facility Location and Sizing
- Network Design and Management
- Project Planning
- Scheduling
- Routing
- Portfolio Optimization
- Data Science and Machine Learning techniques
- …..

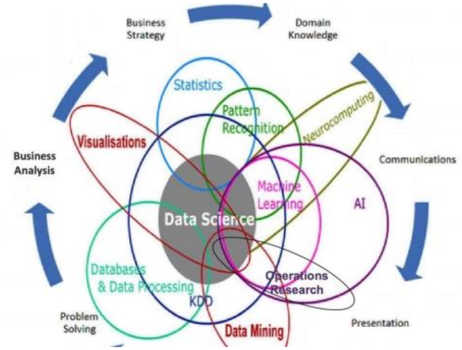







Resource Leveling vs Resource Smoothing
Activity
Predecessor
1. Apply Critical Path Method
2. Apply Resource Leveling
3. Apply Resource Smoothing
A
B
C
D
E
F
D, E
Resource Histogram
Resource Qty
Resource Limit
© BRAINBOK.CO



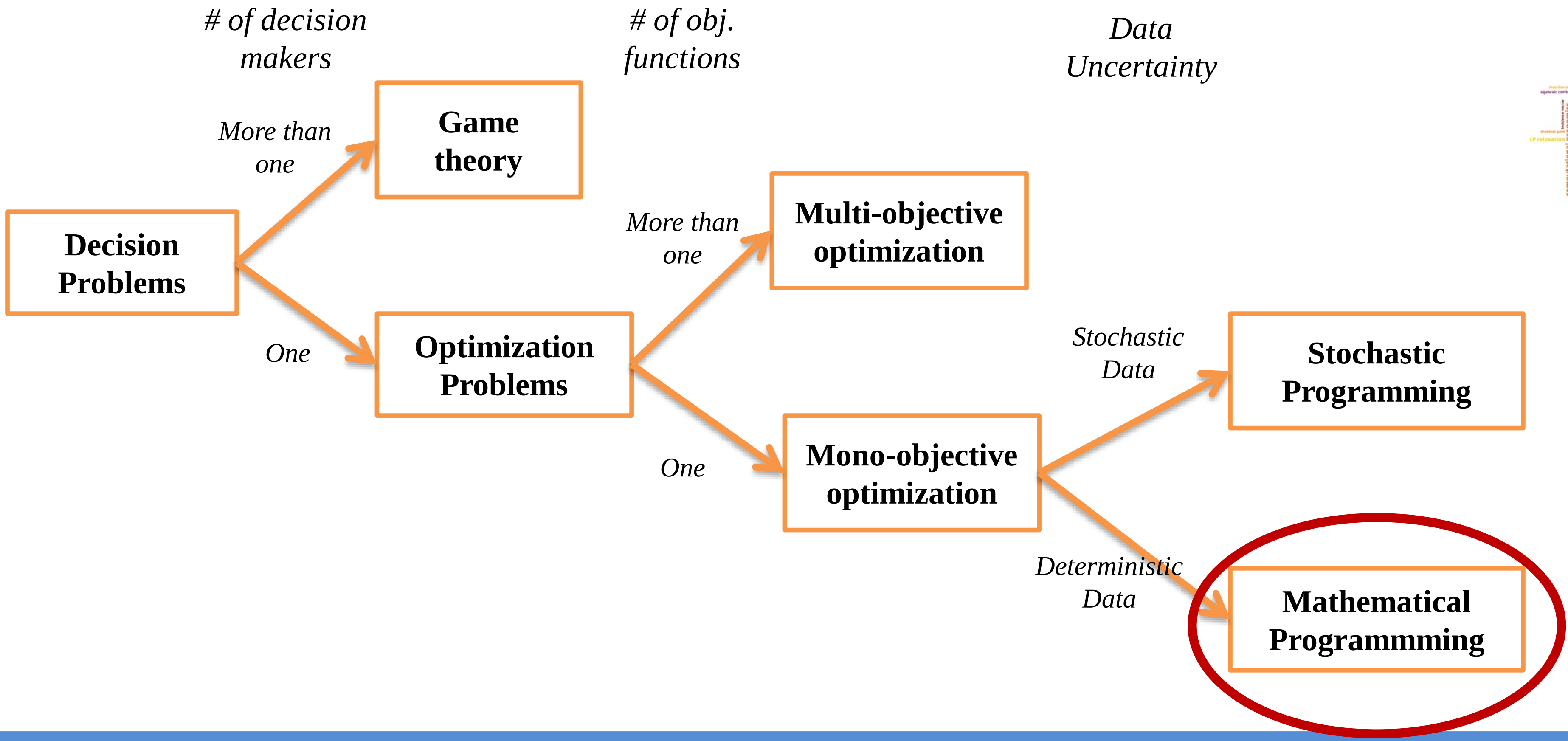


Operations research

It is a mathematical and computer science based approach to deal with the solution of decision problems.

Decision problem:

One or more decision maker have to make a choice among different alternative solutions to optimize one or more objective functions or performance criteria. In a decision problem, choices are not arbitrary. Constraints related to limited resources have to be taken into account.



# of decision makers

# of obj. functions

Data Uncertainty

Decision Problems

More than one

Game theory

One

Optimization Problems

More than one

Multi-objective optimization

One

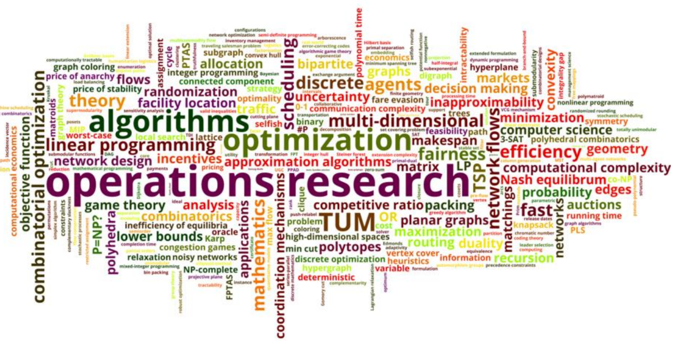
Mono-objective optimization

Stochastic Data

Stochastic Programming

Deterministic Data

Mathematical Programmming



Some examples….game theory

Prisoner's dilemma

Two members of a criminal gang are arrested and imprisoned. Each prisoner is in solitary confinement with no means of communicating with the other. The prosecutors lack sufficient evidence to convict the pair on the principal charge, but they have enough to convict both on a lesser charge. Simultaneously, the prosecutors offer each prisoner a bargain. Each prisoner is given the opportunity either to betray the other by testifying that the other committed the crime, or to cooperate with the other by remaining silent. The possible outcomes are:

- If A and B each betrays the other, each of them serves 3 years in prison
- If A betrays B but B remains silent, A will be set free and B will serve six years in prison (and vice versa)
- If A and B both remain silent, both of them will serve only one year in prison (on the lesser charge)

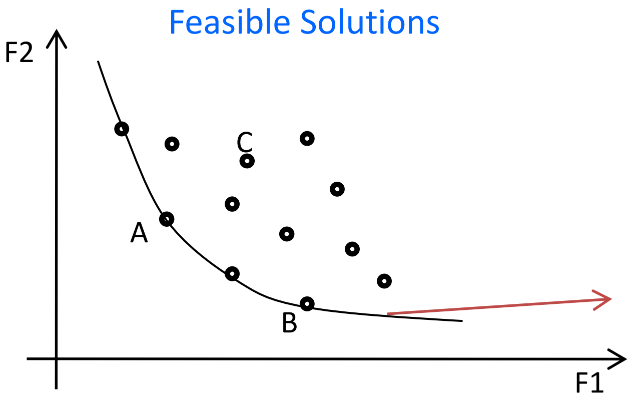
$$\begin{bmatrix} (-3,-3) & (0,-6) \\ (-6,0) & (-1,-1) \end{bmatrix}$$

Because defection always results in a better payoff than cooperation regardless of the other player's choice, it is a **dominant strategy**. Mutual defection is the only strong **Nash equilibrium** in the game (*i.e. the only outcome from which each player could only do worse by unilaterally changing strategy*). The dilemma, then, is that mutual cooperation yields a better outcome than mutual defection but is not the rational outcome because the choice to cooperate, from a self-interested perspective, is irrational.

Some examples….multi-objective optimization

In a production planning problem, the decision maker of a company has the following targets:

- F_1 -> minimizing the number of unsatisfied clients (maximizing quality of service)
- F_2 -> minimizing overall procution costs



A dominates C

B and C are not comparable

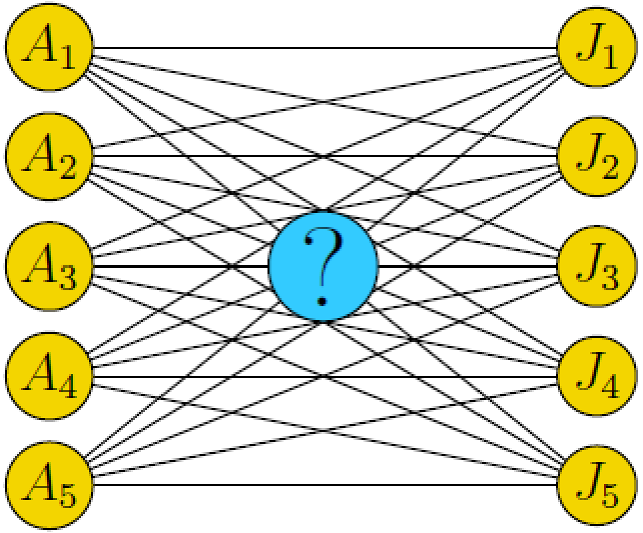
A and B Pareto optimal solutions

Pareto frontier

More Pareto (non-dominated) optimal solutions exist.

Some examples….mathematical programming

Assignment problem (job-worker)



Let us consider 70 jobs and 70 workers

- ✓ i=1,...,70 → workes and j=1,...,70 → jobs
- ✓ If the i-th worker makes the j-th job a cost c_{ij} is paid
- ✓ Each worker can make just one job (constraint)
- ✓ Each job can be made by just one worker (constraint)
- ✓ We have to define the worker-job assignment to minimize the overall cost (objective)

How to solve it?...*Brute Force:*

- ✓ build all the possible worker-job assignments and compute the related cost
- ✓ choose the best alternative

The number of all possible assignments is 70! (the number of permutations of 70 numbers)

$$70! \cong 100^{100}$$

Some examples….mathematical programming

Two possible permutations

Worker	1	2		70
Job	1	2		70

Worker	1	2		70
Job	2	1		70

Let us suppos to have a computer able to evaluate 10^{6} assignments per second

To “explore” 10^{100} assignments we need 10^{94} seconds = 10^{87} years!

Big Bang (beginning of the Universe) was around $15x10^{9}$ years ago!

Let us consider a computer which is 1000 times faster … we need 10^{84} years

Let us use 10^{9} computers in parallel, we need 10^{75} years.

Operations Research main target is to solve these problems
by algoritghms which are able to find optimal or ‘good’ sub-optimal solutions
with a reasonable computation time and burden

Mathematical programming

One decision maker, one performance criterion, no uncertainty about data

$$\min f(x)$$
$$x \in X$$

$f : \Re^n \to \Re$ ⟶ *Objective function*

$X \subseteq \Re^n$ ⟶ *Set of feasible solutions*

$x = \left(x_1, x_2, \ldots, x_n\right)$ ⟶ *Decision variables*

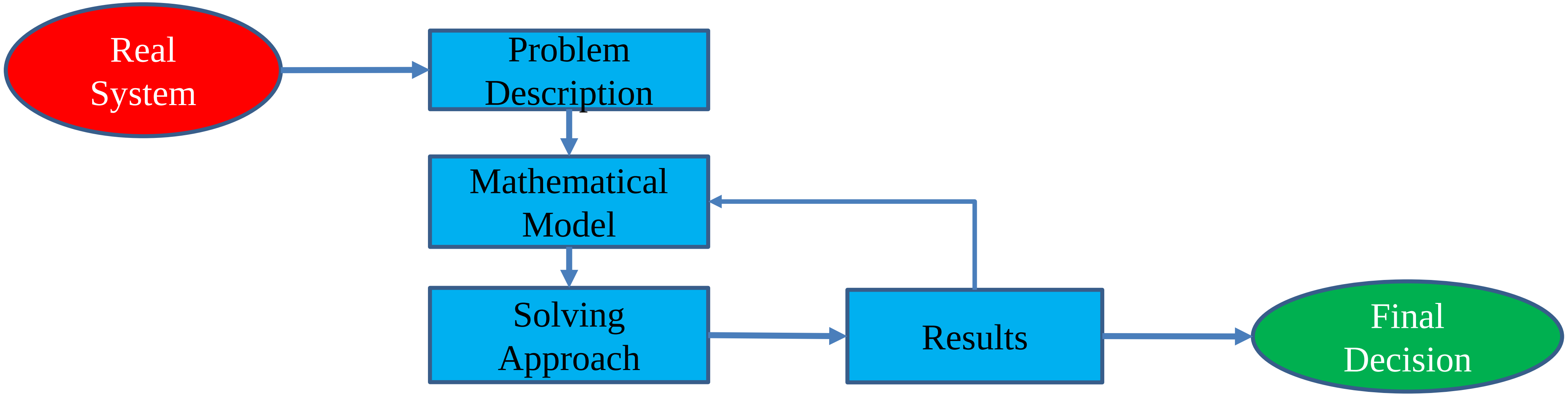
$$\min f(x)$$
$$g_1(x) \leq b_1$$
$$\ldots\ldots\ldots\ldots\ldots\ldots$$
$$g_m(x) \leq b_m$$

$f : \Re^n \to \Re$ ⟶ *Objective function*

$g_i : \Re^n \to \Re$ i=1...m ⟶ *Constraints*

Linear programming (PL)
Non-Linear Programming
Integer Non-Linear and Linear Programming

Modelling approach



Problem Description consists in analyzing the structure of the problem to identify logic and functional relationships and targets

Model building or problem formulation, consists in describing the problem by mathematical relationships

Solution of the problem by ad-hoc exact or heuristic approaches

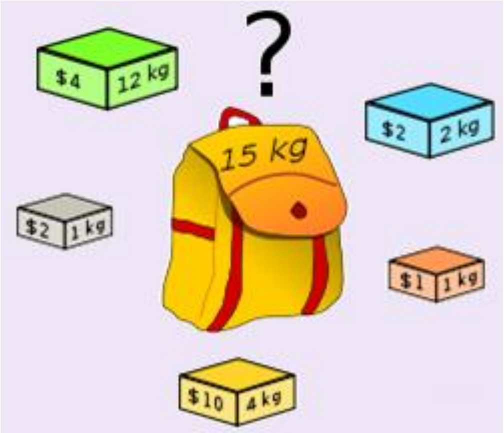
Model validation by experimentation and simulation

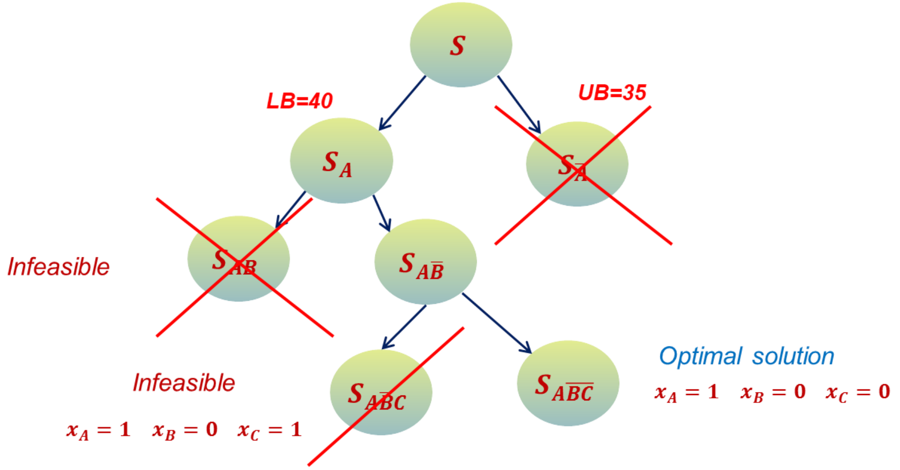
A first basic example….knapsack problem

Let us consider:

- ✓ A ***knapsack*** characterized by a ***maximum capacity b***
- ✓ A set of items ***i є {1,…,n}*** each of the characterized by a weight p_i and a value v_i

We want to determine the subset of items to be put in the knapsack with the aim of maximizing the overall value and satisfying capacity constraints.





Formulation

$$\max \sum_{i=1}^{n} v_i x_i$$

$$\sum_{i=1}^{n} p_i x_i \leq b$$

$$x_i \in \{0,1\} \quad i = 1..n$$

Modeling with binary variables

Modelling with binary variables: logical conditions

Do / Dont’do decisions

Binary variables: variables assuming only values *0* and *1*, expressing ‘*do/don’t do decisions*’, ‘*yes/no*’ choices

Choice among several possibilities

Let us consider a set of ‘*do/dont’ do’* decisions/options/projects to be selected: *A, B, C, …, H*

Let us define the following set binary variable sassociated with each decision: *a, b, c, …., h*

$a + b + c + d + e + f + g + h \leq 1$ **No more than one** option

$a + b + c + d + e + f + g + h \geq 1$ **At least one** option → OR (∨ condition)

$a + b + c + d + e + f + g + h = 1$ **A single option** → XOR (⊕ condition)

a	b	a ∨ b	a ⊕ b
F	F	F	F
F	T	T	T
T	F	T	T
T	T	T	F

These conditions can be generalized as follows to a set of binary variables x_i, $i \in B$:

$$\sum_{i \in B} x_i \begin{cases} \leq k \\ \geq k \\ = k \end{cases}$$

Modelling with binary variables: logical conditions

Simple Implications with two variables

- If we **do** A then we must **do** B, in other words: A→ B (implication)

$b \geq a$

a	b	a → b
F	F	T
F	T	T
T	F	F
T	T	T

Four possible conditions:
{do not do A, do not do B},
{do not do A, do B},
{do A, do not do B}, → RULED OUT
{do A, do B}

- If we **do** A then we must **not do** B, 'if A then not B':

The property of *notdoing* B can be modeled very easily when we already have a binary variable *b* (representing doing B).

$$\bar{b} = 1 - b \quad \Rightarrow \quad 1 - b \geq a \quad \Rightarrow \quad a + b \leq 1$$

- If we **do not do** A then we must **do** B, 'if not A then B'.

$$\bar{a} = 1 - a \quad \Rightarrow \quad b \geq 1 - a \quad \Rightarrow \quad a + b \geq 1$$

- If we **do** A we must **do** B, and if we **do** B we **do** project A: A←→ B (double implication)

$$a \geq b \qquad b \geq a \quad \Rightarrow \quad a = b$$

a	b	a ←→ b
F	F	T
F	T	F
T	F	F
T	T	T

Modelling with binary variables: logical conditions

Simple Implications with three variables

- If we **do** A then we must **do** B **and** C: $b \geq a$ $c \geq a$

- If we **do** A then we must **do** B **or** C:

$$b + c \geq a$$

- If we **do** B **or** C then we must **do** A:

$$a \geq b \qquad a \geq c$$

- If we **do** both B and C then we must **do** A. One way to think about it is to express it in the following way: ‘if we **do** both B and C then we must **not do not A**

$$b + c + (1 - a) \leq 2$$

$$b + c - a \leq 1$$

$$a \geq b + c - 1$$

There is no effect on *a* when *b* and *c* are 0,
or when just one of *b* and *c* is 1, but *a* does have to be 1 when both *b* and *c* are 1.

Modelling with binary variables: logical conditions

Generalized Implications with more variables

- If we **do two or more** of B, C, D **or** E then we must **do** A:

$$\xcancel{a \geq b \qquad\qquad -1}$$

$$a \geq \frac{1}{3}\,(b + c + d + e \; - 1)$$

- If we **do** *M* **or more of** *N* projects (B, C, D, ...) then we must do **A**’ by the constraint:

$$a \geq \frac{b + c + d + \;\ldots - M + 1}{N \; - M + 1} \quad \Rightarrow \quad \left.\begin{array}{l} a \geq \dfrac{b + c \; - 1 + 1}{2 \; - 1 + 1} \\[2ex] a \geq \dfrac{1}{2}\,(b + c) \end{array}\right\} \text{If we do B or C then we must do A}$$

Modelling with binary variables: products

Products of binary variables

$$c = a * b \Rightarrow \begin{cases} c \leq a \\ c \leq b \\ c \geq a + b - 1 \end{cases}$$

a	b	c	$c = a * b$	$c \leq a$	$c \leq b$	$c \geq a + b - 1$
0	0	0	Yes	Yes	Yes	Yes
0	0	1	No	No	No	Yes
0	1	0	Yes	Yes	Yes	Yes
0	1	1	No	No	Yes	Yes
1	0	0	Yes	Yes	Yes	Yes
1	0	1	No	Yes	No	Yes
1	1	0	No	Yes	Yes	No
1	1	1	Yes	Yes	Yes	Yes

$$d = a * b * c \Rightarrow \begin{cases} d \leq a \\ d \leq b \\ d \leq c \\ d \geq a + b + c - 2 \end{cases}$$

Modelling with binary variables: dichotomies

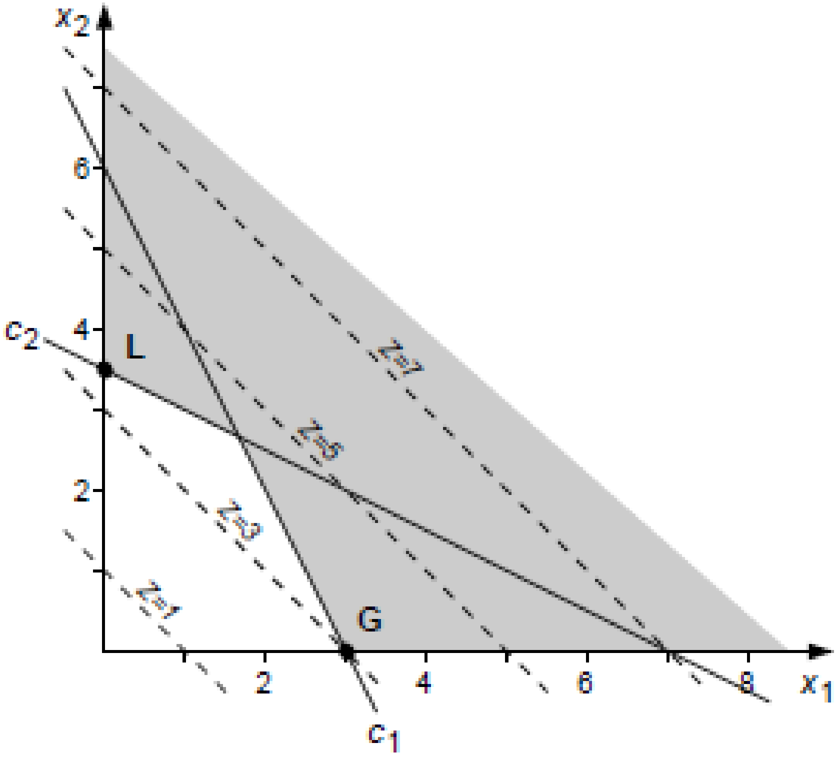
Dichotomies: either/or constraints

$$Minimize\ Z = x_1 + x_2$$

$$x_1 \geq 0, x_2 \geq 0\ and$$

$$Either\ 2 * x_1 + x_2 \geq 6\ (constr\ 1)\ or\quad x_1 + 2 * x_2 \geq 7\ (constr\ 2)$$

$$\begin{gathered} 2 * x_1 + x_2 \geq 6 * b \\ x_1 + 2 * x_2 \geq 7 * (1 - b) \end{gathered}$$



Binary variables do everything

Binary variables do everything

General integer variables

Consider an integer variable v which must take a value between 0 and 10: We could replace this integer variable with four binary variables b_1, b_2, b_3, and b_4 everywhere in the model using the expression

$$v = b_1 + 2 \cdot b_2 + 4 \cdot b_3 + 8 \cdot b_4$$

remembering that we also have to have the constraint

$$b_1 + 2 \cdot b_2 + 4 \cdot b_3 + 8 \cdot b_4 = 10$$

Semi-continuous variables

Suppose we have a semi-continuous variable s which can take on either the value 0 or any value between some lower limit L and some upper limit U. If we introduce a binary variable b then we can represent the semi-continuity of s by the following pair of constraints

$$L \cdot b \leq s$$
$$s \leq U \cdot b$$

- Either $b = 0$, in which case s is constrained to be 0,
- or $b = 1$, in which case s is constrained to lie between L and U.

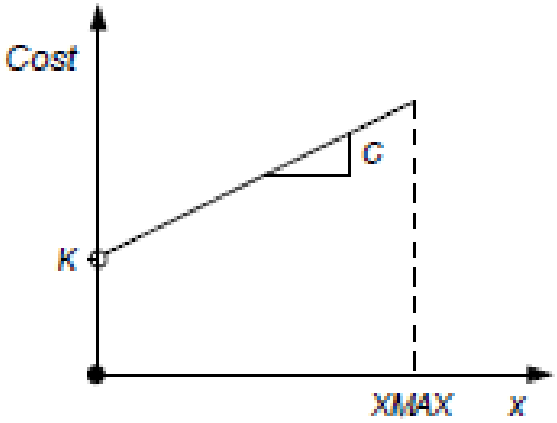
Partial integer and Special ordered sets variables…next slides

Connecting real variables to binary variables

Connecting real and binary variables: fixed cost

We want to model fixed costs, that is, where we incur a fixed cost ***K*** if a particular real variable ***x*** is strictly greater than **0** (e.g. cost of setting up a piece of machinery.

We do not incur the fixed cost if we do not set up the piece of machinery but if the output level ***x*** of the machine is anything different from zero then we have to incur the fixed cost.



Cost:

- If the throughput is **zero** then the cost is **zero**
- else-if throughput takes a value which is **different** from **0** then the cost is $\boldsymbol{Cost = K + C \cdot x}$, where ***C*** is the per unit cost of output ***x***.

Let us introduce a binary variable b which is equal to zero if x is equal to zero and equal to 1 if x is strictly greater than zero.
Then the cost equals $b \cdot K + C \cdot x$, but we have to ensure that if b is zero then x is zero.
We can do this by introducing the constraint

$$x \leq X_{max} \cdot b$$

where X_{max} is the largest value that x can take.

Connecting real and binary variables: counting

Often we want to have a constraint that only a certain number of real variables are strictly greater than 0.

An example might be where we have a whole variety of different ingredients that could go into a blend and we have a constraint that perhaps at most 10 of these ingredients can actually go into the blend.

If the quantity of item *i* going into the blend is $\boldsymbol{x_i}$ and we require no more than 10 of these to be non zero, for each item *i* we introduce a binary variable b_i and have constraints of the form

$$x_i \leq VOL \cdot b_i$$

where *VOL* is the volume of the blend to be made.
These constraints ensure that if a particular x_i is greater than zero then the corresponding b_i must be equal to 1.

Then the constraint that says that no more than 10 of the ingredients must be non zero could be expressed as:

$$\sum_{i=1}^{10} b_i \leq 10$$

Connecting real and binary variables: partial integer

A partial integer is a variable which has to be integral in any acceptable solution if the value is less than some bound, and otherwise can be real.

The natural way to express a partial integer p is:

$$p = v + x$$

where variable v is **an integer that must be less than or equal to *U*** (*i.e.* $0 \ \ v \leq U$) and x is **a real variable**.

We have already seen that the integer variable v can be expressed in terms of several binary variables using the binary expansion.

Now we have to express the fact that if v **is strictly less than** U then x **must be zero**. Suppose that we know some value X_{max} which x is guaranteed not to exceed, *i.e.* an upper bound for x. Then we can introduce a new binary variable b and the following constraints

$$x \leq X_{max} \cdot b$$

$$b \leq {}^{v}/_{U} \text{ or equivalently } v \geq U \cdot b$$

To show that this formulation is valid, consider the two possible values of b.

First, if $b = 0$ then $x = 0$, *i.e.* x **is equal to** 0 and $v \geq 0$.

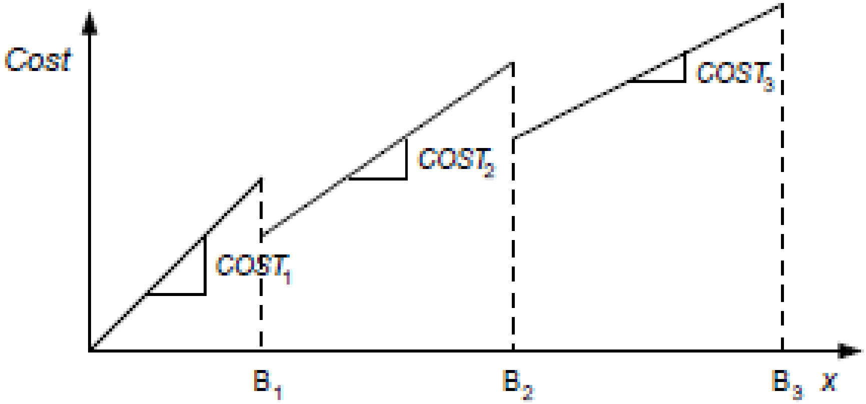
If $b = 1$ then $x \leq X_{max}$ and $v \geq U$, which combined with the fact that $v \leq U$ means $v = U$. So if $b = 1$, v **is at its maximum**, and we can then start **having non-zero amounts of** x.

Connecting real and binary variables: price breaks

Basic Idea – All item discount pricing:

If we buy a number of items between 0 and B_1 then for each item we pay an amount $COST_1$, whereas if the quantity we buy lies between B_1 and B_2 we pay an amount $COST_2$ for each item. Finally if we buy an amount between B_2 and B_3 then we pay an amount $COST_3$ for each item.

For the sake of concreteness we assume that if we order exactly B_1 items then we get the unit price $COST_2$, whereas if we order just a little less than B_1 then we would get the unit price $COST_1$, and similarly for all the other price breaks.



To model these item price breaks we use:

- binary variables b_1, b_2 and b_3, where b_i is 1 if we have bought any items at a unit cost of $COST_i$
- real decision variables x_1, x_2 and x_3 which represent the number of items bought at price $COST_1$, $COST_2$ and $COST_3$

Note that we cannot buy any items at price $COST_2$ until we have bought the maximum number of items at the higher price $COST_1$, otherwise the solution would be easy as we buy all items at the least expensive unit price!

The total amount x that we buy is given by

$$x = x_1 + x_2 + x_3$$

Connecting real and binary variables: price breaks

$$\begin{cases} B_1 \cdot \ b_2 \leq \ x_1 \leq B_1 \cdot \ b_1 \\ (B_2 \ - \ B_1) \cdot b_3 \leq \ x_2 \leq (B_2 \ - \ B_1) \cdot b_2 \\ x_3 \leq (B_3 \ - \ B_2) \cdot b_3 \\ b_1 \geq b_2 \geq \ b_3 \end{cases}$$

First relation imposes that if we have bought any in the second price range ($b_1 = b_2 = 1$) then x_1 is fixed to B_1.

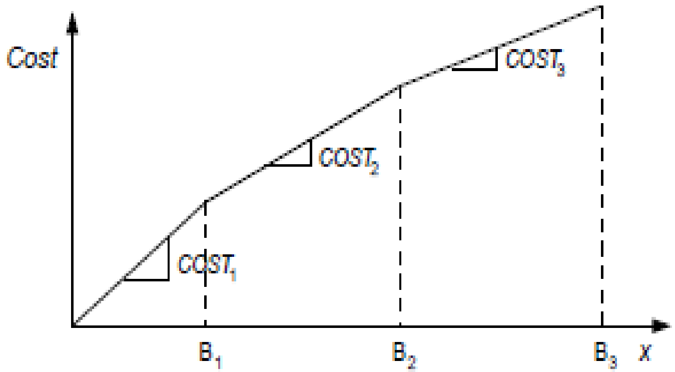
Second relation ensures that if $b_2 = 0$, then $x_2 = 0$, whereas if $b_2 = 1$ then x_2 is only constrained by the maximum amount that can be bought at the price $COST_2$.

Third relation ensures that $x_3 = 0$ if $b_3 = 0$, and that if $b_3 = 1$ then we cannot buy more than the maximum number at price $COST_3$. Equations

Forth relation ensures that we can only buy at a lower price if we have bought at all the higher prices.

Connecting real and binary variables: economies of scale

We are buying a certain number of items and we get discounts incrementally.
The unit cost for items between 0 and B_1 is C_1, whereas items between B_1 and B_2 cost C_2 each, and items between B_2 and B_3 cost C_3 each.
At the points 0, B_1, B_2 and B_3 we introduce real valued decision variables w_i (i = 0, 1, 2, 3) and 3 binary variables b_i. We also define cost break points CBP_i that correspond to the total cost of buying quantities 0, B_1, B_2 and B_3.



$$\begin{cases} CBP_0 = 0 \\ CBP_1 = C_1 \cdot B_1 \\ CBP_2 = CBP_1 + C_2 \cdot (B_2 - B_1) \\ CBP_3 = CBP_2 + C_3\ (B_3 - B_{2)} \end{cases}$$

$$w_0 + w_1 + w_2 + w_3 = 1$$

$$Cost = 0 \cdot w_0 + CBP_1 \cdot w_1 + CBP_2 \cdot w_2 + CBP_3 \cdot w_3$$

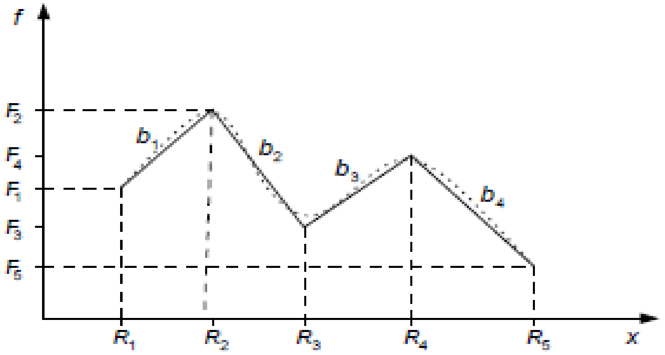
$$x = 0 \cdot w_0 + B_1 \cdot w_1 + B_2 \cdot w_2 + B_3 \cdot w_3$$

$$w_0 \le b_1 \qquad w_1 \le b_1 + b_2 \qquad w_2 \le b_2 + b_3 \qquad w_3 \le b_3$$

$$b_1 + b_2 + b_3 = 1$$

For a solution to be valid, at most two of the w_i can be non-zero, and if there are two non-zero they must be contiguous, thus defining one of the line segments

An ordered set of variables, of which at most two can be non-zero, and if two are non-zero these must be consecutive in their ordering



Connecting real and binary variables: product of binary and real variable

Let us consider:

$$y = b \cdot x$$

where x and y are real variables, and b is binary.

Suppose we have some upper bound U on the value of x. Then consider the following constraints:

$$\begin{cases} y \leq x \\ y \geq x - U \cdot (1 - b) \\ y \leq U \cdot b \end{cases}$$

If $b = 0$, then third inequality means that $y = 0$.
If $b = 1$, then third inequality is harmless, and we have $y \leq x$ from first inequality and $y \geq x$ from second inequality, *i.e.* $y = x$, which is what is desired.

More generally:

$$y = f(x) \cdot b$$

Then, being U and L an upper and a lower bound respectively of f(x):

$$\begin{cases} y \leq f(x) - L \cdot (1 - b) \\ y \geq x - U \cdot (1 - b) \\ y \leq U \cdot b \\ y \geq L \cdot b \end{cases}$$

ILP and MILP samples

Max-min and min-max problem

- Let us consider a generic mathematical programming model with linear constraints and the following objective function:

$$max \min \left\{e_1(\boldsymbol{x}), e_2(\boldsymbol{x}), \dots, e_q(\boldsymbol{x})\right\}$$

where $e_1(\boldsymbol{x}), e_2(\boldsymbol{x}), \dots, e_q(\boldsymbol{x})$ are linear functions.

It is always possible to transform a ‘max-min’ (or ‘min-max’) model in an equivalent linear model in the following way:

- ✓ Introducing a new variable $y = \min \left\{e_1(\boldsymbol{x}), e_2(\boldsymbol{x}), \dots, e_q(\boldsymbol{x})\right\}$ and the objective function is substituted by: $max\ y$

- ✓ Adding the following constraints: $y \leq e_i(x), \forall i \in 1, \dots q$

Max-min and min-max problem: sample problem

A product is realized by composing 3 parts which have to be processed by four different production lines. The table reports the capacity of each line and the production rate (#pieces/h) of each part on each line.

Line	Capacity	Production Rate (# pieces/ h)		
		Part 1	Part 2	Part 3
1	100	10	15	5
2	150	15	10	5
3	80	20	5	10
4	200	10	15	20

We want to determine the number of hours that each part has to be processed on each line with the aim of maximizing the number of realized complete products.

Max-min and min-max problem: sample problem

- ❑ **Decision Variables**:
  - x_{ij} *number of hours each part j is processed on line i*

- ❑ **Formulation:**

$$max\left\{min\overbrace{\begin{bmatrix}(10x_{11}+15x_{21}+20x_{31}+10x_{41});\\(15x_{12}+10x_{22}+5x_{32}+15x_{42});\\(5x_{13}+5x_{23}+10x_{33}+20x_{43})\end{bmatrix}}\right\}$$

$$s.t.$$

$$\left.\begin{aligned}x_{11}+x_{12}+x_{13}&\le 100\\x_{21}+x_{22}+x_{23}&\le 150\\x_{31}+x_{32}+x_{33}&\le 80\\x_{41}+x_{42}+x_{43}&\le 200\end{aligned}\right\}\ \textit{Capacity Constraints}$$

$$x_{ij}\ge 0 \qquad intero\ \forall i,\forall j$$

Max-min and min-max problem: sample problem

- ❑ **Linear formulation:**

$$\max \ \boxed{y}$$

$$s.t.$$

$$\left\{\begin{array}{l} y \leq 10x_{11} + 15x_{21} + 20x_{31} + 10x_{41} \\ y \leq 15x_{12} + 10x_{22} + 5x_{32} + 15x_{42} \\ y \leq 5x_{13} + 5x_{23} + 10x_{33} + 20x_{43} \end{array}\right\} \textit{Additional Constraints}$$

$$\left.\begin{array}{l} x_{11} + x_{12} + x_{13} \leq 100 \\ x_{21} + x_{22} + x_{23} \leq 150 \\ x_{31} + x_{32} + x_{33} \leq 80 \\ x_{41} + x_{42} + x_{43} \leq 200 \end{array}\right\} \textit{Capacity Constraints}$$

$$x_{ij} \geq 0 \qquad \textit{intero}\ \forall i, \forall j$$

min-abs problem

- Let us consider a generic mathematical model with *linear constaints and the following objective function*:

$$min|e(\boldsymbol{x})|$$

where $e(x)$ is linear

Remembering that $|e(\boldsymbol{x})| = max\{e(\boldsymbol{x}); -e(\boldsymbol{x})\}$ the model min-abs can be transformed in equivalent linear one in the following way:

- ✓ Introducing a new variable $y = \max\{e(\boldsymbol{x}),\ -e(\boldsymbol{x})\}$ and the objective function becomes: $min\, y$
- ✓ Adding the following inequalities: $y \geq e(x), y \geq -e(x)$

min-abs problem: just-in-time scheduling sample problem

A supervisor has to plan the processing of 5 lots on a single machine. Each of them requires the following processing times 5 minutes, 7 minutes, 4 minutes, 7 minutes and 10 minutes.

The processing sequence is fixed a-priori: 1-2-3-4-5. No overlapping can be made by the lots.

The first lot has a desired delivery at 10.32, the second at 10.38, the third at 10.42, the fourth at 10.52 and the fifth at 10.57.

Let us define the 'lot error' has the absolute value of the difference between the processing time and the delivery time. We want to determine the processing initial time in order to minimize the sum of the lot errors, assuming that the process starts at 8.30.

min-abs problem: just-in-time scheduling sample problem

❑ **Decision variable**:

- i_j *tempo di inizio lavorazione del lotto* j

❑ **Formulation:**

$$\min \quad |i_1 + 5 - 122| + |i_2 + 7 - 128| + |i_3 + 4 - 132| + |i_4 + 7 - 142| + |i_5 + 10 - 147|$$

$s.t.$

$$\left.\begin{aligned} i_2 &\geq i_1 + 5 \\ i_3 &\geq i_2 + 7 \\ i_4 &\geq i_3 + 4 \\ i_5 &\geq i_4 + 7 \end{aligned}\right\} \quad \textit{Precedence constaints}$$

$$i_j \geq 0 \qquad \forall j$$

min-abs problem: just-in-time scheduling sample problem

- ❑ **Linear formulation:**

$$\min \quad y_1 + y_2 + y_3 + y_4 + y_5$$

$$s.t.$$

$$y_1 \geq i_1 + 5 - 122$$
$$y_1 \geq 122 - 5 - i_1$$
$$y_2 \geq i_2 + 7 - 128$$
$$y_2 \geq 128 - 7 - i_2$$
$$y_3 \geq i_3 + 4 - 132$$
$$y_3 \geq 132 - 4 - i_3$$

$$y_4 \geq i_4 + 7 - 142$$
$$y_4 \geq 142 - 7 - i_4$$
$$y_5 \geq i_5 + 10 - 147$$
$$y_5 \geq 147 - 10 - i_5$$

$$i_2 \geq i_1 + 5$$
$$i_3 \geq i_2 + 7$$
$$i_4 \geq i_3 + 4$$
$$i_5 \geq i_4 + 7$$

$$i_j \geq 0 \qquad \forall j$$

What about max-abs problem

- Let us consider a generic mathematical model with *linear constaints and the following objective function*:

$$\mathrm{max}|e(\boldsymbol{x})|$$

where e(x) is linear

Remembering that $|e(\boldsymbol{x})| = max\{e(\boldsymbol{x}); -e(\boldsymbol{x})\}$ the model max-abs can be transformed in the following way:

- ✓ Introducing a new variable $\mathrm{k} = \min\ \{e(\boldsymbol{x}),\ -e(\boldsymbol{x})\}$ and the objective function becomes: $max\ k$
- ✓ Adding the following inequalities: $k \leq e(x) \qquad if\ e(x) \geq 0$

  $k \leq -e(x) \qquad if\ e(x) \leq 0$

⇒

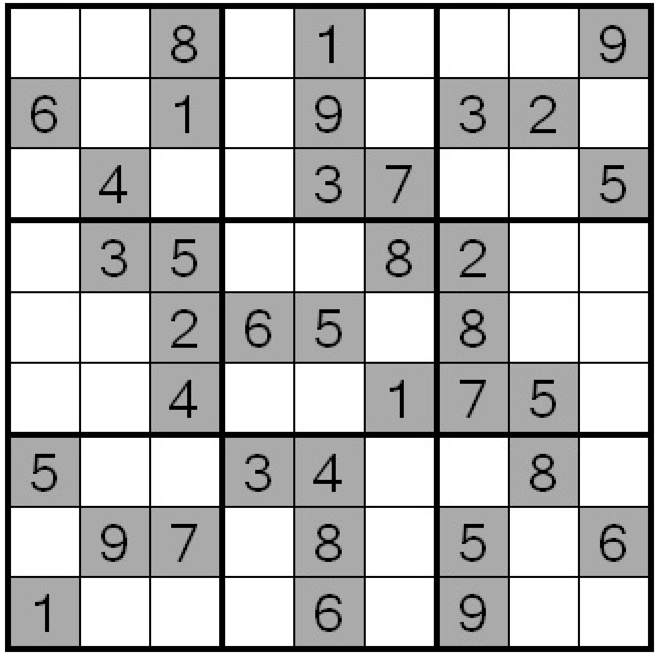
$$k \leq e(x) + M{\cdot}y$$

$$y \leq -e(x) + M{\cdot}(1\text{-}y)$$

$$M{\cdot}(1\text{-}y) \geq e(x)$$

$$M{\cdot}y \geq \text{-}\ e(x)$$

Sudoku and linear programming



		8		1				9
6		1		9		3	2	
	4			3	7			5
	3	5			8	2		
		2	6	5		8		
		4			1	7	5	
5			3	4			8	
	9	7		8		5		6
1				6		9		

Sudoku and linear programming

✓ Given an *inital matrix* where several elements (i, j) are defined, and let RQ_h, $h = 1, \ldots, 9$, be the *h-th square* in tables.

❑ **Decision Variable**:

– $x_{i,j,k}$ $i, j, k \in 1, \ldots, 9$ $x_{i,j,k} = 1$ *if the element* (i, j) *takes value* k, *0 otherwise*.

❑ **Objective function**:

No real objective fucntion, since we are only looking for a feasible solution. When we use an optimization software, we can define a dummy objective function (eg. $\min z \, , z \geq 0$)

Sudoku and linear programming

- ❑ **Constraints**:

$$\sum_{k=1}^{9} x_{i,j,k} = 1 \qquad \forall i,j$$

Each element has to assume a value between 1 and 9

$$\sum_{j=1}^{9} x_{i,j,k} = 1 \qquad \forall i,k$$

In each row all the numbers between 1 and 9 have to be present

$$\sum_{i=1}^{9} x_{i,j,k} = 1 \qquad \forall j,k$$

In each column all the numbers between 1 and 9 have to be present

$$\sum_{(i,j)\in RQ_h}^{9} x_{i,j,k} = 1 \qquad \forall h,k$$

In each square, all the numbers between 1 and 9 have to be presente

$$x_{i,j,k} = 1 \qquad \forall\ \textit{element}\ (i,j)\ \textit{in the starting matrix containing number}\ k$$

Travelling salesman problem (TSP) → Hamiltonian Cycle

❑ A *travelling salesman* has to visit a given number of ***cities***, leaving from its city and coming back to the same city at the end of the tour.

❑ The travelling salesman wants to minimize the run distance.

The ***TSP*** is a combinatorial optimization problem and is the most known and studied problem in literature. Its ‘success’ is due to the fact that: :

➢ It is very easy to define

➢ It is very difficult to solve, even with few cities

Travelling salesman problem (TSP) → Hamiltonian Cycle

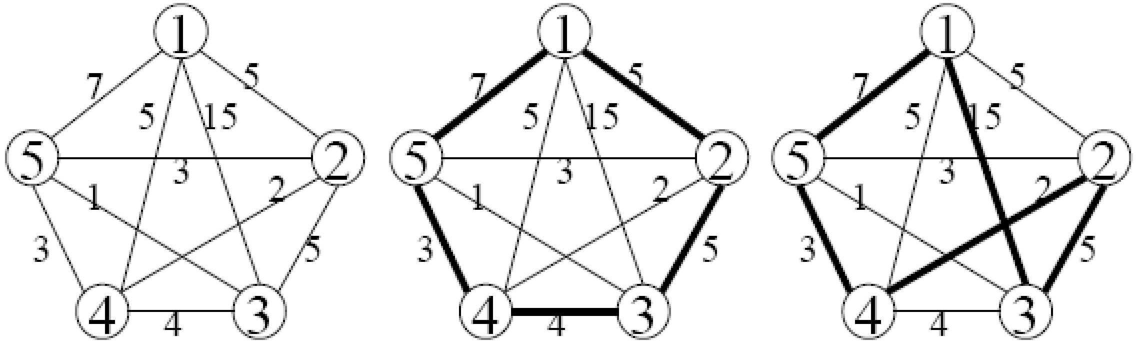
Hamiltonian Cycle:

Given a graph *G(V,E)*, the Hamiltonian cycle is the cycle traversing all the nodes only once.

Cost of Hamiltonian Cycle

If we associate a positive weight d_{uv} to each arc of the graph *G(V,E)* each hamiltonian cycle is associated with a cost given by the sums of all the arcs composing the cycle

The TSP consists in determining the hamiltonian cycle with the minimum cost



Travelling salesman problem (TSP) → Hamiltonian Cycle

Formulation:

❑ **Decision variables**:

- $x_{ij} = 1$ *if arc* (i,j) ϵ A *belongs to* the hamiltonian cycle,
- $x_{ij} = 0$ otherwise

❑ **Objective function**:

$$\min \sum_{(i,j)\in A} d_{ij} x_{ij}$$

❑ **Constraints**:

$$\sum_{i:(i,j)\in A} x_{ij} = 1 \qquad j \in V$$

→ *In each node j an arc has to enter*

$$\sum_{i:(j,i)\in A} x_{ji} = 1 \qquad j \in V$$

→ *From each node j an arc has to leave*

$$x_{ij} \in \{0,1\} \qquad ij \in A$$

$$\sum_{\substack{i\in S,\, j\in S:\\ (i,j)\in A}} x_{ij} \le |S|\text{-}1 \qquad \forall \mathrm{S} \subset \mathrm{V}: 2 \le |\mathrm{S}| \le |V| - 1$$

Subtour elimination constraints