Optimization

EMINES - 2023

abs.um6p.ma

Day 5 - Detailled schedule

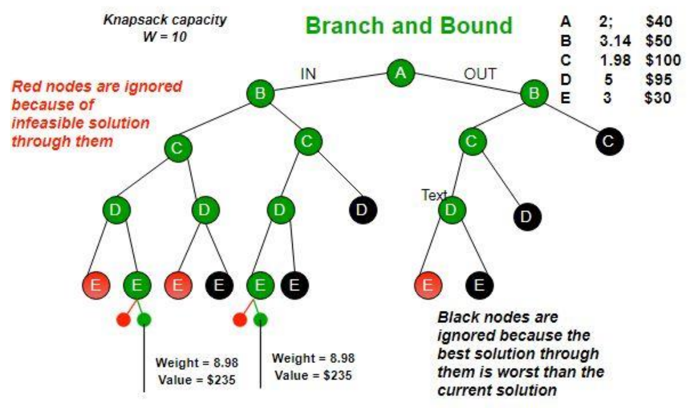
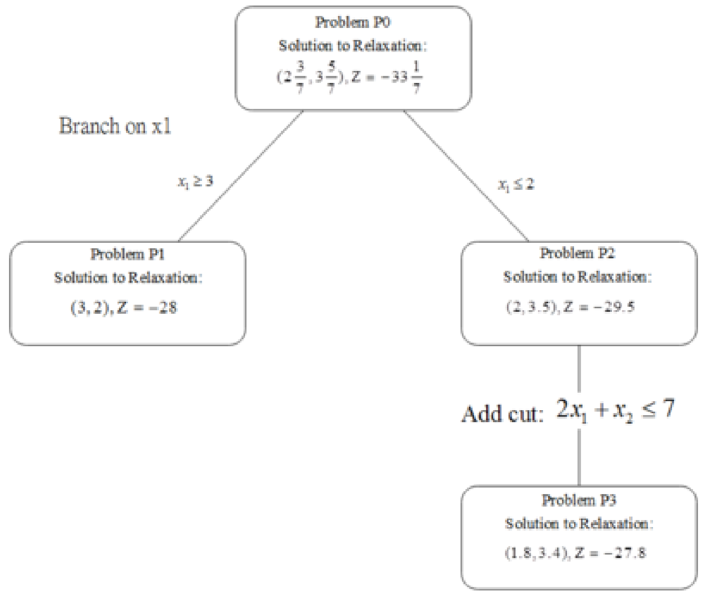
Morning	
9h – 10h	Refresher on Combinatorial optimization. Scheduling pb + The index Fund
10h - 10h30	Multi-criteria optimization
10h30-10h45	*Break*
10h45- 11h30	Considering uncertainty
11h30 – 12h15	The efficient portfolio The newsvendor

Afternoon	
13h – 13h45	The savvy farmer
13h45 – 14h30	The savvy farmer
14h30-14h45	*Break*
14h45 – 16h	Play with simplex The red tomato Wrap-up

Africa Business School

Refresher

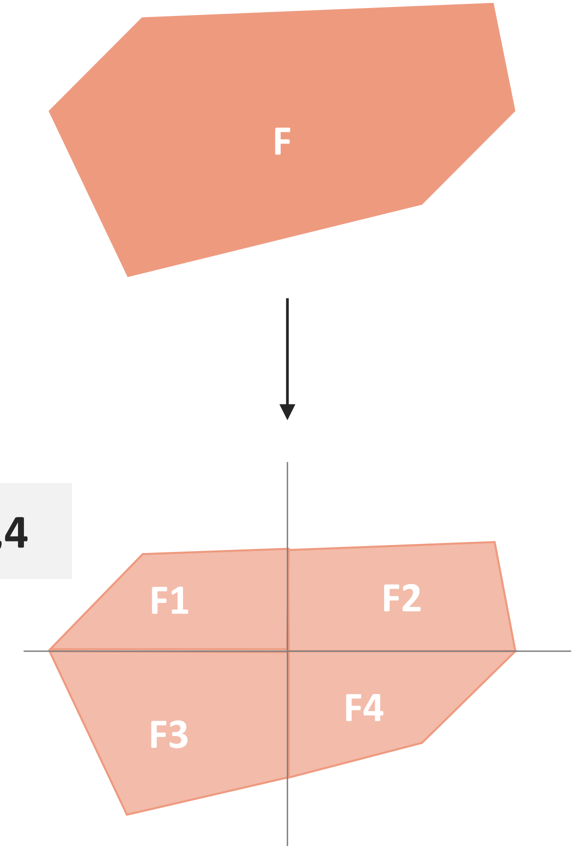
Several algorithms exist for solving MILPs

	Branch & Bound	Branch & Cut
Principle	Use the Linear relaxation to eliminate (= prune) some « infertile » branchs of the tree	Add valid constraints, in order to tighten the linear relaxation (and get closer to the convex hull)
Illustration		

The Branch & Bound algorithms combines two basic tenets

Divide & conquer (= branching)

$p^* = \min_{x \text{ in } F} f(x)$



$p_{i^*} = \min_{x \text{ in } F_i} f(x), i=1,\ldots,4$

$\downarrow$

$p^* = \min_{i=1,\ldots,4} \{p_i^*\}$

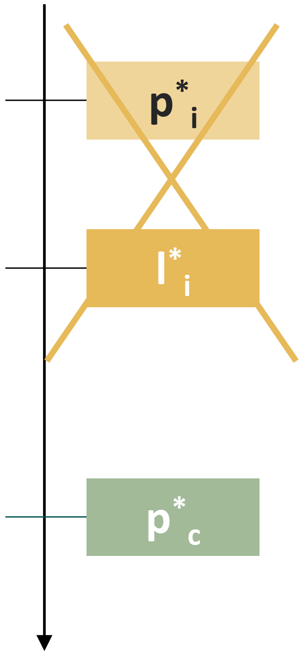
Detect and eliminate the none promising parts

Assume that F has been split into $F_1, \ldots, F_N$

Let i in {1, …, N} and assume that p_c^* is the optimal value on UF_j, for all j < i.

Suppose that we are able to compute (easily) l^*_i a lower bound of p_i^* : $l^*_i \leq p^*_i$

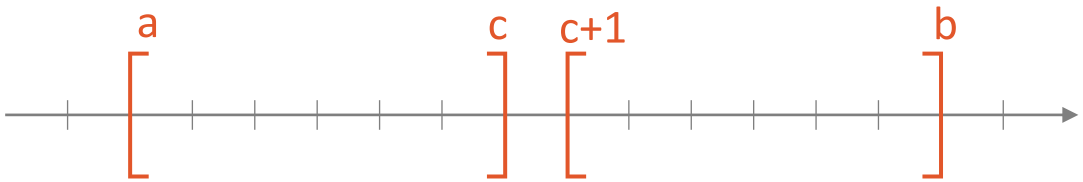
If l^*_i is such that $p_c^* < l^*_i$ then there is no hope that Fi contains the optimal solution. So no need to explore it further. Move on to i+1



In the case of an integer optimization problem, there is a smart way to divide the feasible set into subsets

- Consider an integer variable x_1
- Without loss of generality, we can consider that x_1 lies in the range [a, b]

- Let c an integer in [a, b].
- Then we can split the feasible set in the following way :



At the same time we divide the feasible set in two and we eliminate non-integer solutions

$Z^* = \min c(x)$ s.t. x in F

Branch & bound : what do we mean by branching ?

Suppose the variable x_1 has to be integer.

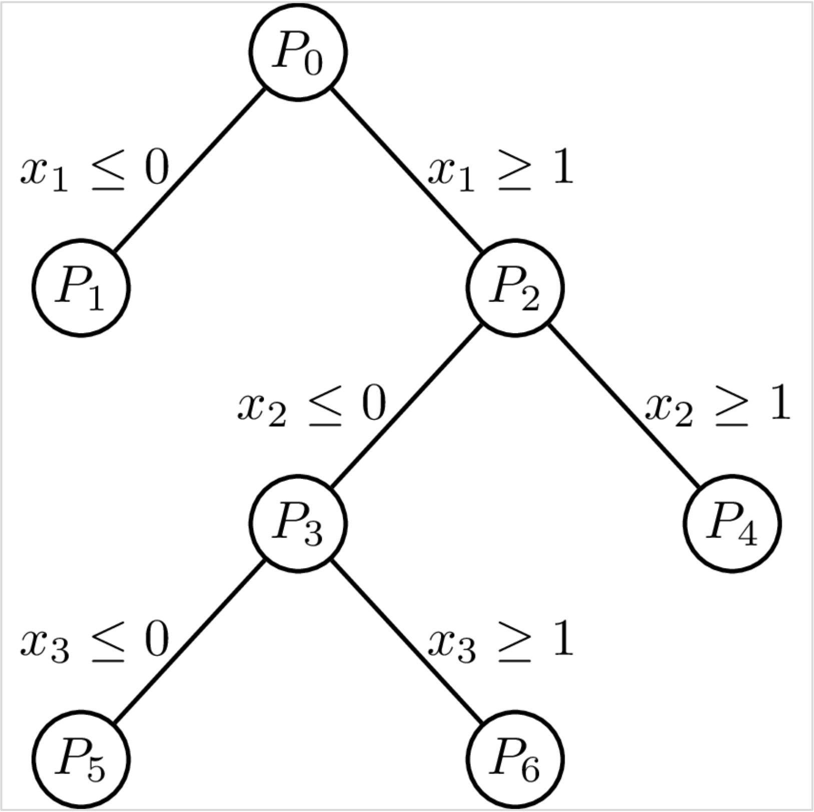
Solve the linear relaxation and get $x_1 = 0.57$

Then define 2 subproblems (branches)

P_1 : by adding $x_1 \leq 0$

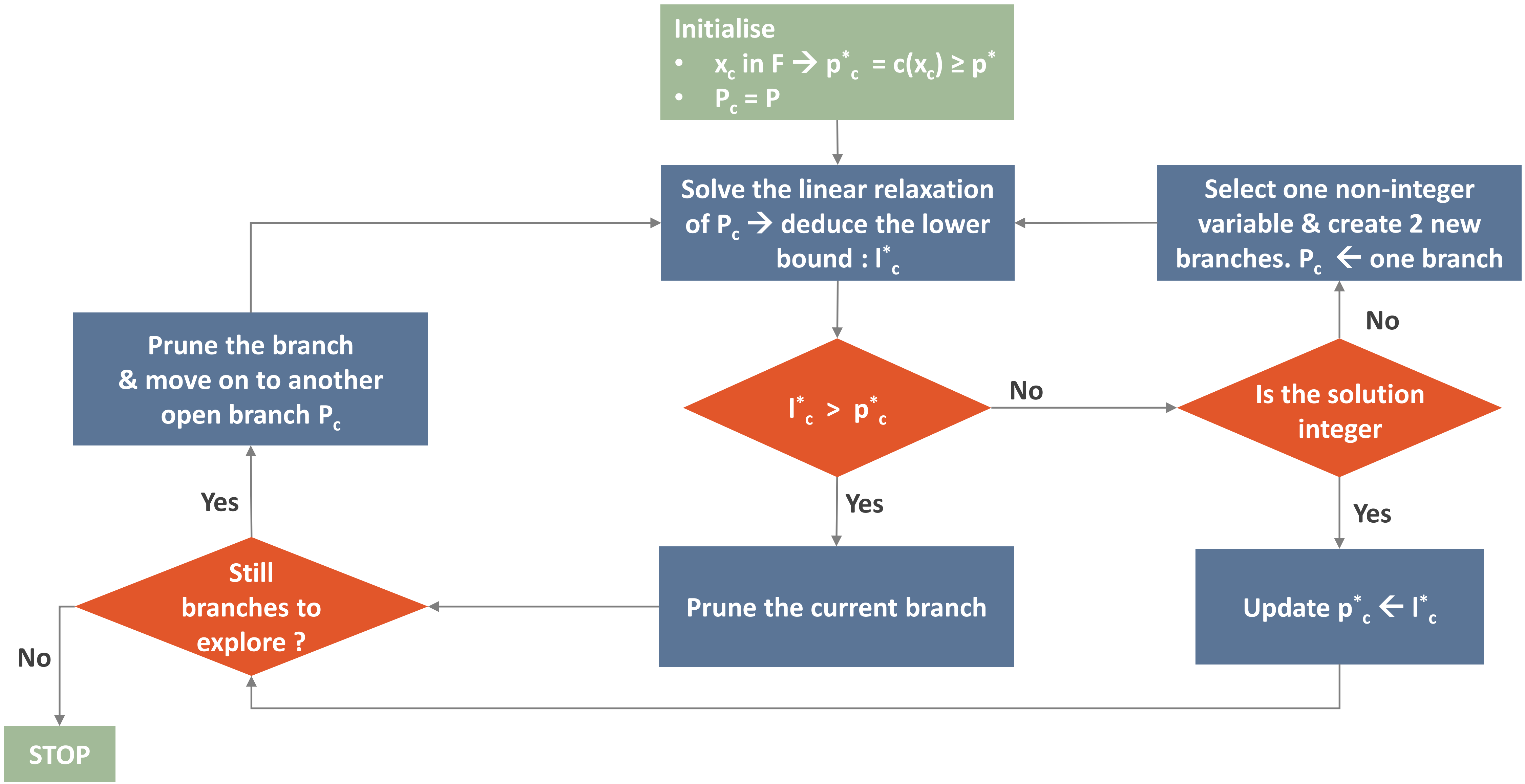
P_2 : by adding $x_1 \geq 1$

$Z^* = \min \{Z^*_1, Z^*_2\}$



Branch & bound

$Z^* = \min c(x)$ s.t. x in F



How to prune ?

$Z^* = \min c(x)$ s.t. x in F

Let p^*_1 the optimal value of of P_1

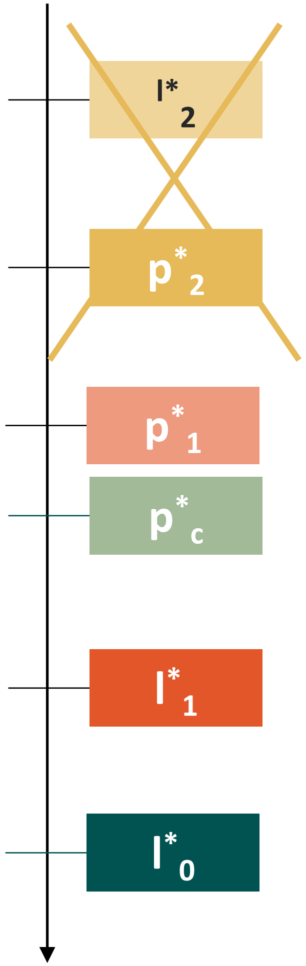
Let l^*_1 the optimal value of the linear relaxation of P_1

Let p*2 the optimal value of P2

Let l^*_2 the optimal value of the linear relaxation of P_2

p^*_c the best current solution

l^*_0 the linear relaxation of the ascendant branch of P_1 & P_2

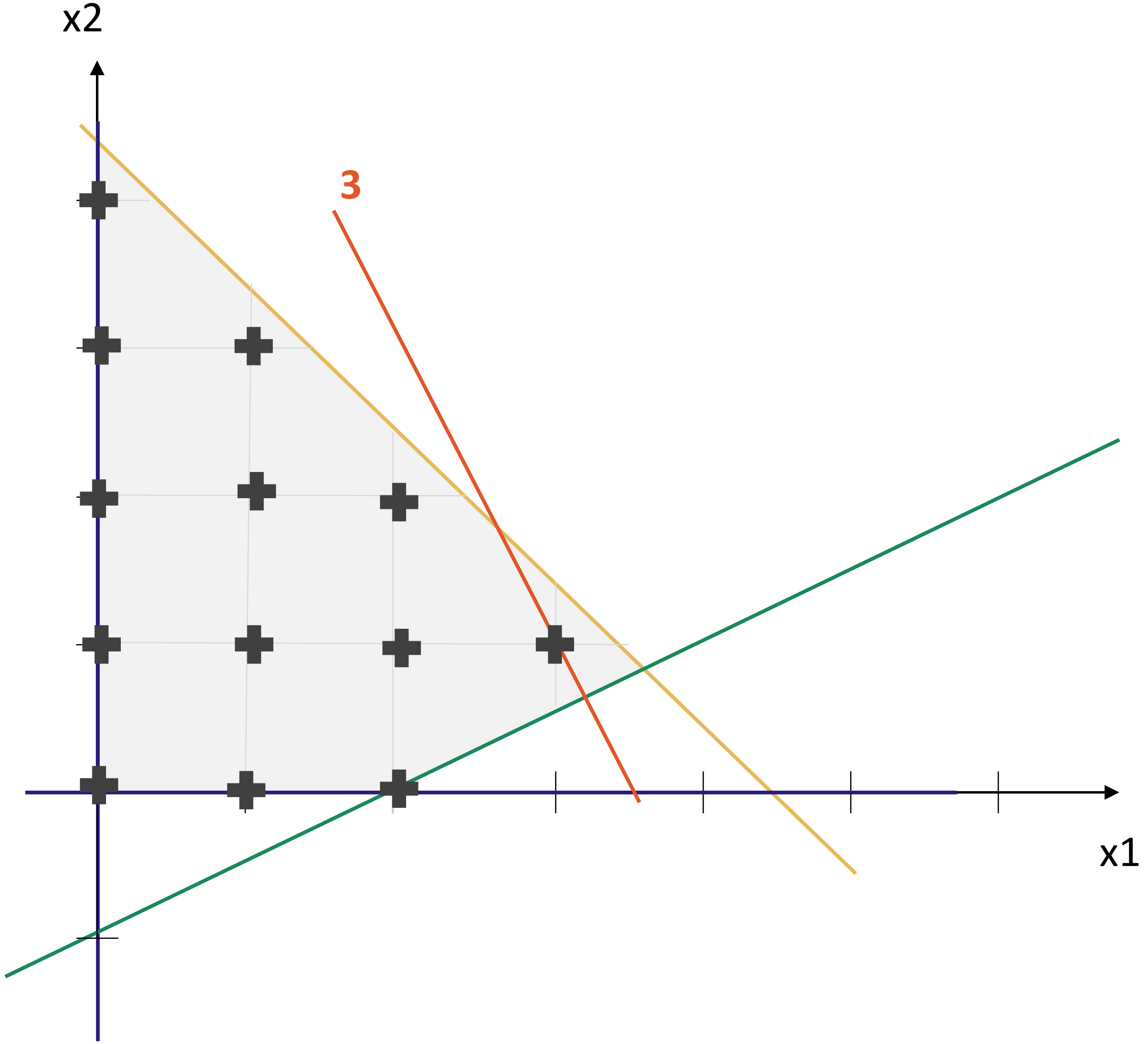


All descendants of a branch have an optimal value that is higher (= less good) than that of their ancestor, since the more you get in the tree, the more you add constraints

So if the ancestor is already worse than the current solution, we can delete the branch
→ ***no hope of improving the current solution on this branch***

→ So we can prune P2

Geometry of Integer Linear Programming



The feasible set is a countable set with 13 solutions (✚)

Max $2x1 + x2$

$x1 + x2 \leq 4.5$

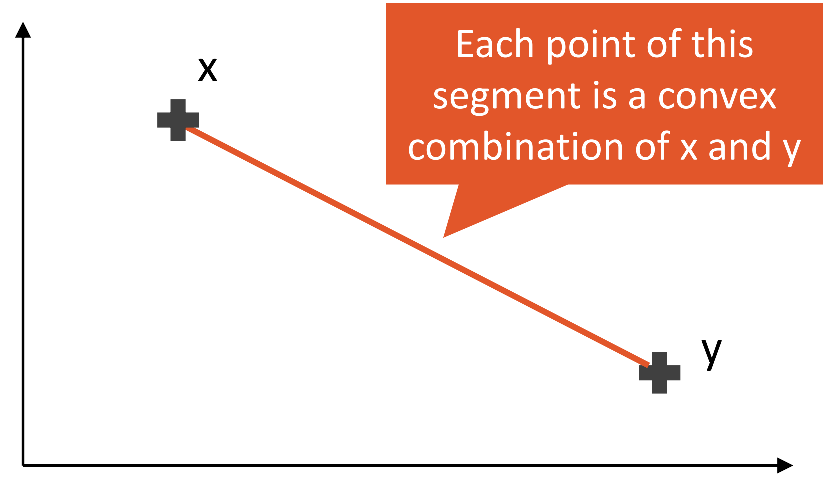
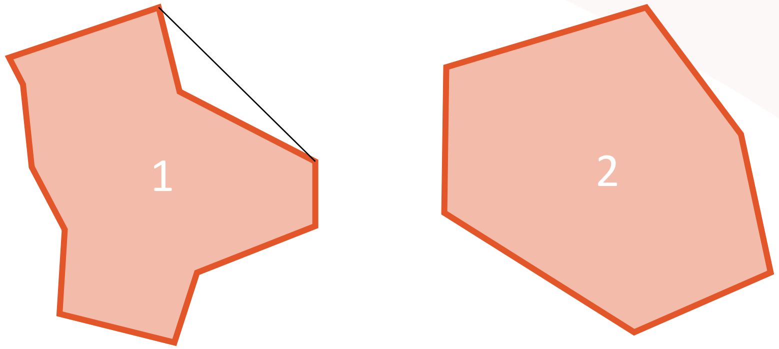
$- x1 + 2\ x2 \geq -2$

$x1 \geq 0,\ x2 \geq 0$

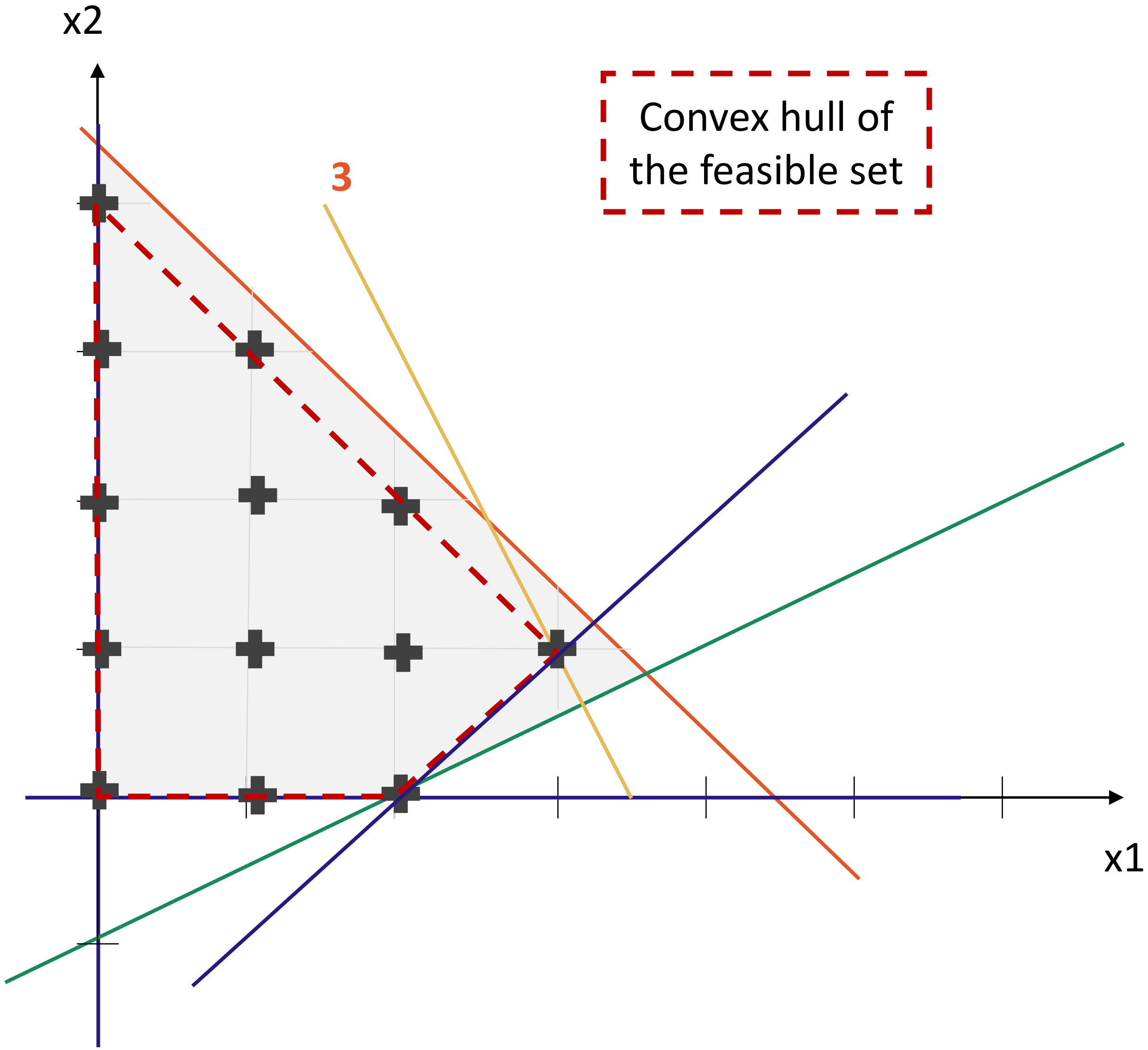
x1, x2 integer

WARNING : different formulation may work for a same problem.
Which one is the best ?

Refresher : convexity

	Convex combination	Convex set
Definition	z is a **convex combination** of x and y if there exists λ in [0, 1] such that z = λ x + (1- λ) y	**S is convex** if and only if : **x,y in S ==> z in S**, *for any z convex combination of x and y*
Illustration	Ex in $\mathbb{R}^2$ 	Ex in $\mathbb{R}^2$  **What about the feasible set of a MILP ?**

Notion of convex hull



Definition : a convex hull is the smallest LP feasible region that contains all of the integer solutions

If you are able to describe **the convex hull** through a set of **linear constraints,** then **the integer solution is obtained by solving this new linear program**

Pb : determining the convex hull may be a very difficult problem

Definition : a constraint if valid if adding it does not reduce the feasible set

Branch & cut, or cutting planes method

Overview

To iteratively refine the feasible set of the relaxed problem (P) using linear inequalities, termed **cuts**

Basic idea

Let consider a linear constraint C added to the problem.

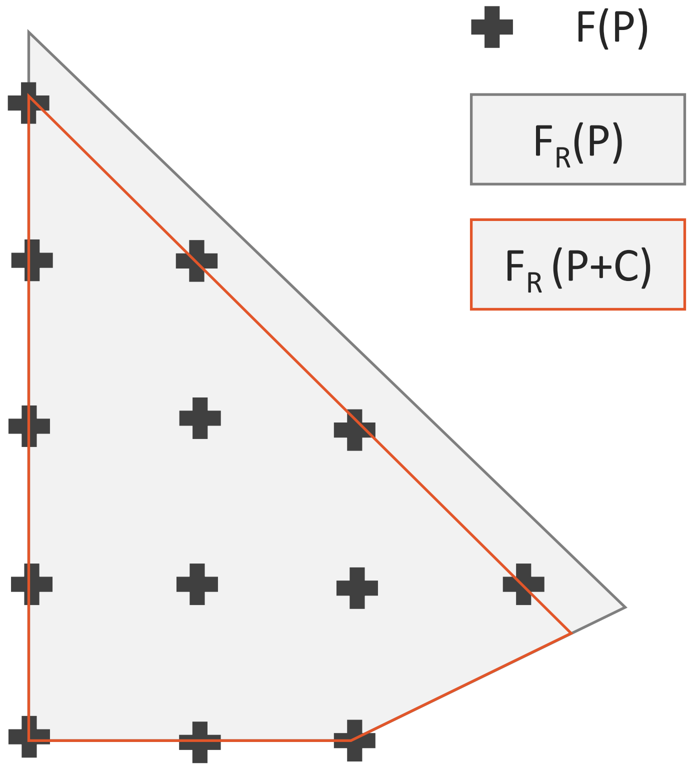
- Let F be the feasible set of a problem, and F_R the feasible set of its linear relaxation
- Let (P) be the initial problem and (P+C) the problem augmented with the constraint C.

Then, **the constraint C is a cut if** :

- $F(P+C) = F(P)$: It does not eliminate feasible solutions of the original IP problem.
- $F_R(P+C) \subset F_R(P)$: The cut removes the optimal solution of the relaxed LP problem.

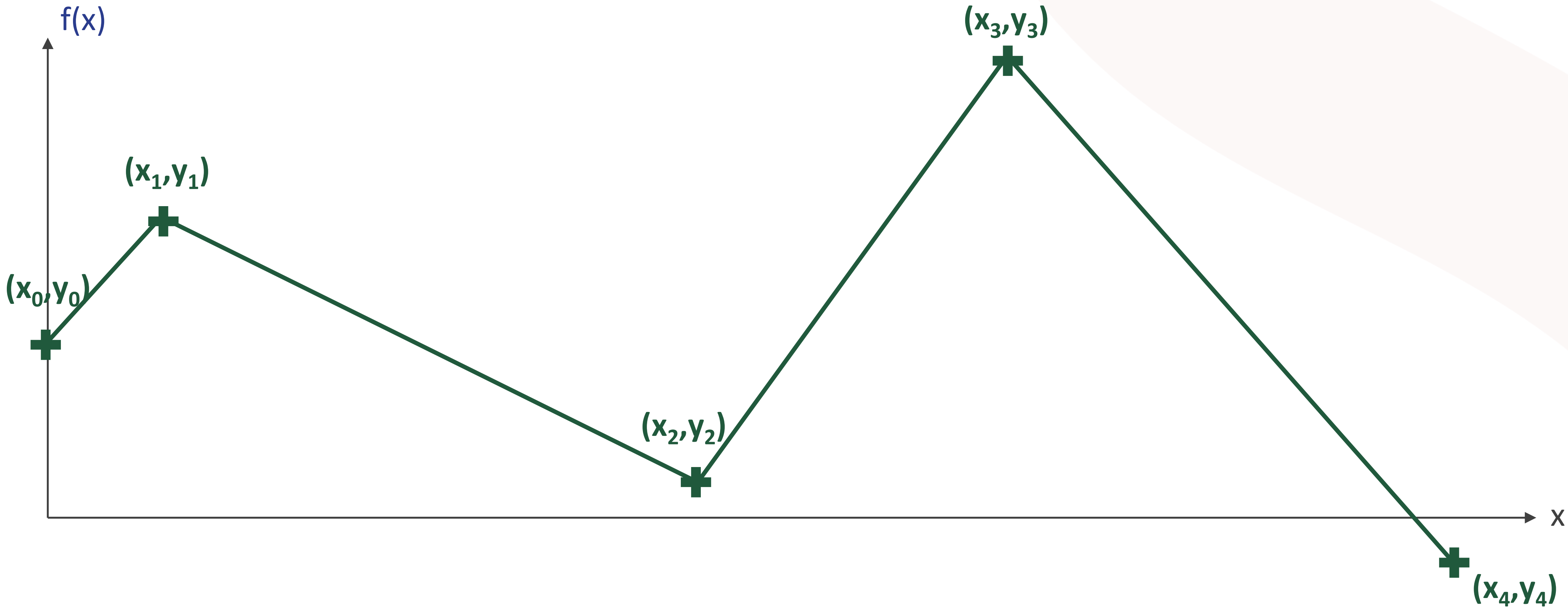
Ultimate goal

The optimal integer solution becomes an extreme point and therefore can be found by solving the LP problem.



Tips 5 : Use binary variables to formulate arbitrary piecewise linear functions

The function f is specified by ordered pairs (x_i, y_i)



Africa Business School

Let’s practice. How to compute f(x) using binary variables ?

Data : n pieces
$(x_i, y_i) \quad \forall i = 0, \dots, n$

Constraints
• $f(x) =$

Solution. How to model this using binary variables ?

Data : n pieces

$(x_i, y_i) \quad \forall i = 0, \dots, n$

Variables

$z_i \in \{0, 1\} \quad \forall i = 1, \dots, n$

→ *indicate whether* $x_{i-1} \leq x \leq x_i$

$\lambda_i \in [0, 1] \quad \forall i = 0, \dots, n$

→ *indicate the position over* $[x_{i-1}, x_i]$ *or* $[x_i\ x_{i+1}]$

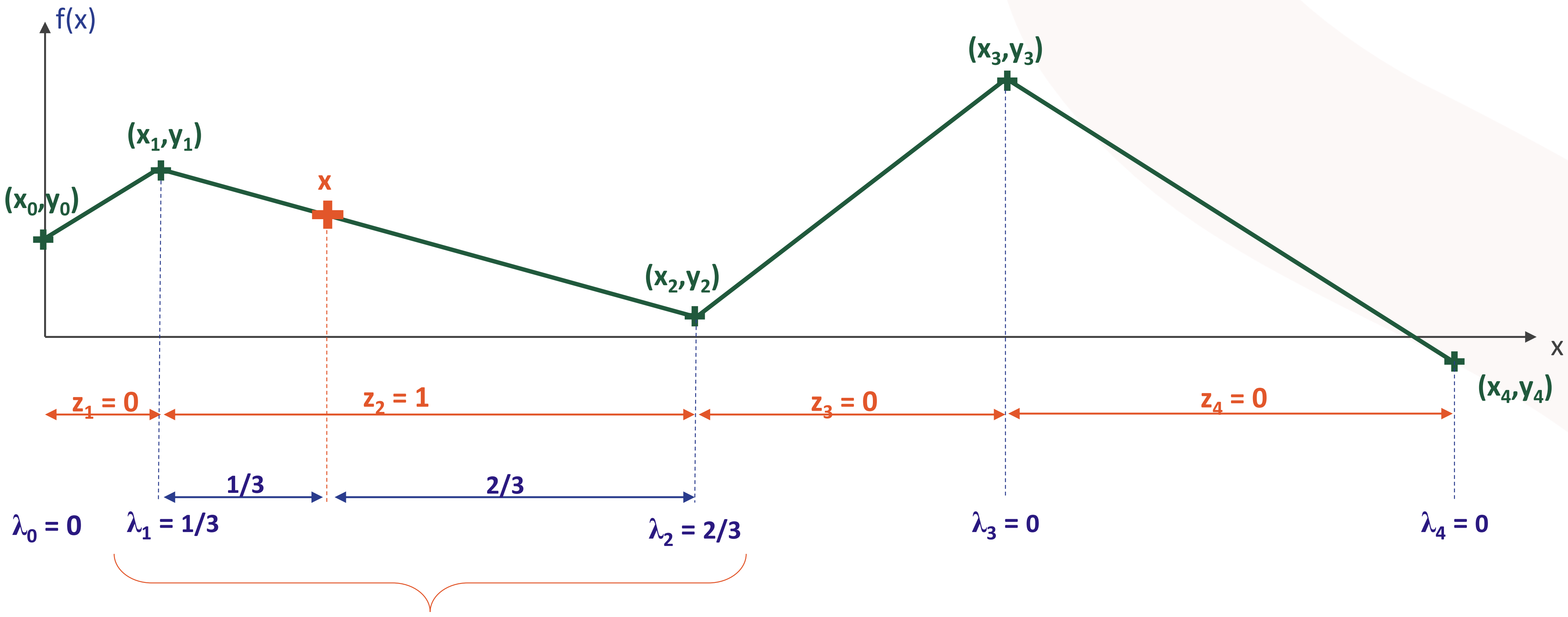
→ *equals 0 if x does not lie in* $[x_{i-1}, x_i]$ *or* $[x_i\ x_{i+1}]$

Constraints

- $f(x) = \sum_{i=0}^{n} \lambda_i y_i$
- $x = \sum_{i=0}^{n} \lambda_i x_i$
- $\sum_{i=0}^{n} \lambda_i = 1$
- $\lambda_i \leq z_i + z_{i+1}$, $i = 1, \dots, n-1$
- $\lambda_0 \leq z_1$
- $\lambda_n \leq z_n$
- $\sum_{i=1}^{n} z_i = 1$
- $\lambda_i \in [0, 1]$
- $z_i \in \{0, 1\}$

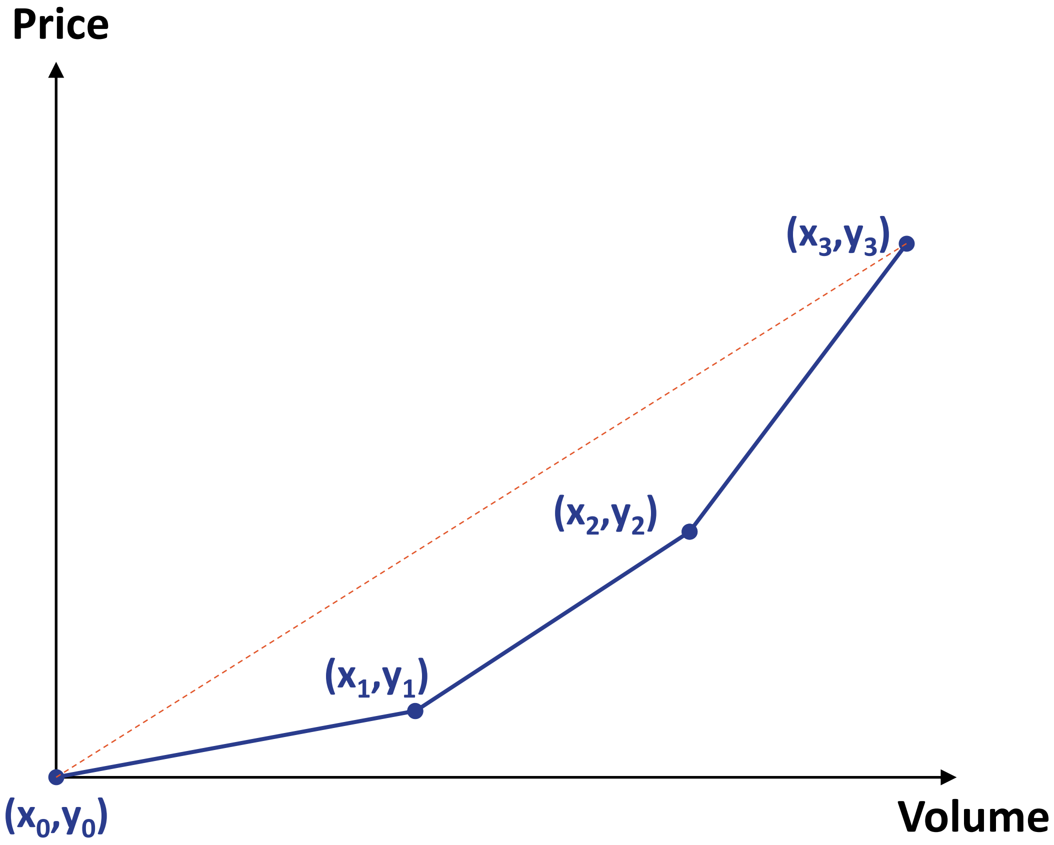
Africa Business School

Illustration



$$f(x) = 1/3\ y_1 + 2/3\ y_2$$

When you try to minimize a convex function (or maximize a concave function), you don't need binary variables any more



Variables

$\lambda_i \in [0,1] \quad \forall i = 0, \dots, n$

→ *indicate the position over* $[x_{i-1}, x_i]$ *or* $[x_i\ x_{i+1}]$

Constraints

Min $\sum_{i=0}^{n} \lambda_i y_i$

- $x = \sum_{i=0}^{n} \lambda_i x_i$
- $\sum_{i=0}^{n} \lambda_i = 1$
- $\lambda_i \in [0,1]$

The model will pick the right values of λ_i since they are the ones that minimizes the objective value

Africa Business School

Tips 6 : Special Ordered Sets (SOS)

SOS1

A set of variables, **at most one of which can take a non-zero value**, all others being at 0
→ choose at most one from a set of possibilities.

Ex : what size of factory to build, when we have a set of options, perhaps small, medium, large or no factory at all

SOS2

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

Special Ordered Sets of type 2 are typically used to model non-linear functions of a variable in a linear model.

What if the solution is not satisfying ?

Unfeasible

Identify the smallest posible subset of mutually incompatible constraints

This feature is provided by some solvers. Ex : « file.ilp »

2 possibilities

- **Manageable number : Remove them**
- **Huge number : Add them one by one**

Not Optimal

Force the solution that you think should be better

Use « Settings constraints »

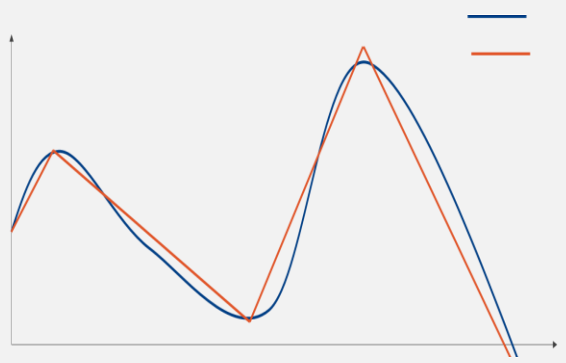
2 possibilities

- **Unfeasible**
- **Worse optimal value**

Mixed-Integer Linear Programming in a nutshell

Extremely powerful

- All analytical problems can be approximated with MILP



But difficult to solve

- Unfortunately very hard (NP-hard) to solve, even for a small number of binary variables (> 1000)
- Not an unique formulation. Generally many of them that can be more ou less efficient

May require alternative methods

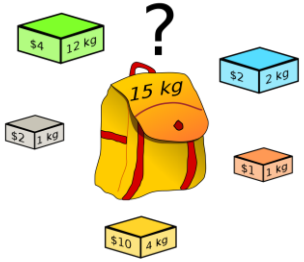
- Decomposition-coordination
  - In the model
  - During the resolution
- Algorithms without garantee of optimality (metaheuristics)

Formulation

Classical optimization problem

Knapsack problem	Transportation problem	Blending problem	Set covering problem
Packing problem	Shortest path	Scheduling	Warehouse location
Assignment problem	Traveling Salesman problem	Assortment problem	Portfolio investment
Min distance / Least-square	Vehicle Routing problem	Inventory management = Lot-sizing	Revenue management
Sorting			

Knapsack



Description

- You are the owner of a knapsack (=resource) with a limited size (=capacity)
- You want to fill it (=use it) with the best possible set of items, among a given list {$item_1$, …, $item_N$}
- « Best » means the set that will bring the maximal value, since each item has a given value
- You cannot take parts of an item. « all or nothing »

Data

- Capacity : C
- i=1, …, N possible items
- Value : V[i]
- Weight : W[i]

Capacity	500	
i	**W[i]**	**V[i]**
1	61	46
2	55	32
3	49	42
4	76	32
5	88	21
6	56	9
7	12	17
8	69	19
9	40	20
10	91	42
11	85	12
12	83	1
13	32	45
14	85	49
15	89	43
16	19	17
17	26	7
18	58	21
19	73	24
20	71	37

Application

- Hiking
- Business planning
- Investment

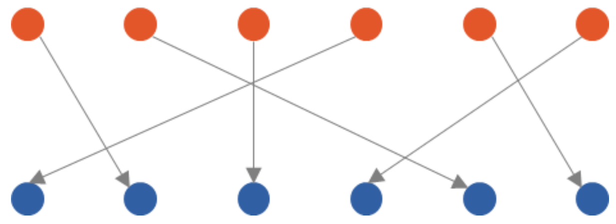
Limitations / extensions

- Multiple constraints
- Combined values

Questions

- **Imagine a greedy heuristic for solving this problem**
- **What would be the decision variables ? What would be their nature ?**

Assignment problem



Description

- You have a list of **supply items** and a list of **demand items** (same number of supply and demand items)
- Each pair supply-demand brings a value
- Find the **matching supply-demand** that maximize the value
- Each supply item shall be allocated once and each demand item shall receive one of them

Data

- i=1, …, N unit of supply
- j=1, …., N unit of demand
- For each i,j : value V[i][j]

Supply i	Demand j	V[i][j]
1	1	5
1	2	19
1	3	7
1	4	13
1	5	13
2	1	16
2	2	2
2	3	5
2	4	9
2	5	5
3	1	14
3	2	21
3	3	13
3	4	13
3	5	11
4	1	10
4	2	20
4	3	22
4	4	15
4	5	2
5	1	13
5	2	13
5	3	16
5	4	22
5	5	5

Application

- Agent-task assignment (ex : taxi-customers)
- Efficient market
- Pairing (marriage, roomates, ..)

Limitations / extensions

- **Unbalanced assignment**
- **Oversupply / over demand market**
- **Transportation problem**

The assignment problem

Find the **matching X** that maximize the value = $\sum_{[i][j]}$ **v[i][j] X[i][j]**

Such that each supply i is allocated once and each demand receives one of them.

v	Supply[i]			
Demand[j]	5	4	7	5
	8	5	3	8
	9	2	3	7
	4	6	2	9

X	Supply[i]			
Demand[j]	0	0	1	0
	0	1	0	0
	1	0	0	0
	0	0	0	1

v	Supply[i]			
Demand[j]	5	4	7	5
	8	5	3	8
	9	2	3	7
	4	6	2	9

Total = 30

Warehouse location = Facility location problem

Description

- You must decide **which of *n* warehouse to open** for meeting the demands of *m* customers
- Once you've decided to operate a warehouse, you have to decide **how much to ship** from this warehouse to the customers
- Your goal is to minimize the cost, made of two lines :
  - **Fixed operating cost** per warehouse i, if opened
  - **Per unit** sum of operating cost at warehouse i and shipping cost from i to customer j
- Good can be shipped from a warehouse only if it opened

Data

- i=1, …, N potential warehouse location
- j=1, …., N unit of demand
- For each j : demand d[i]
- Fixed cost : c_f[j]
- Variable cost : c_v[i][j]

Warehouse i	Demand j	cv[i][j]
1	1	7
1	2	8
1	3	13
1	4	7
2	1	2
2	2	15
2	3	21
2	4	7
3	1	9
3	2	9
3	3	7
3	4	5

Warehouse i	cf[i]
1	10
2	15
3	8

Demand j	d[j]
1	8
2	5
3	11
3	17

Applications

- **Value-chain design**

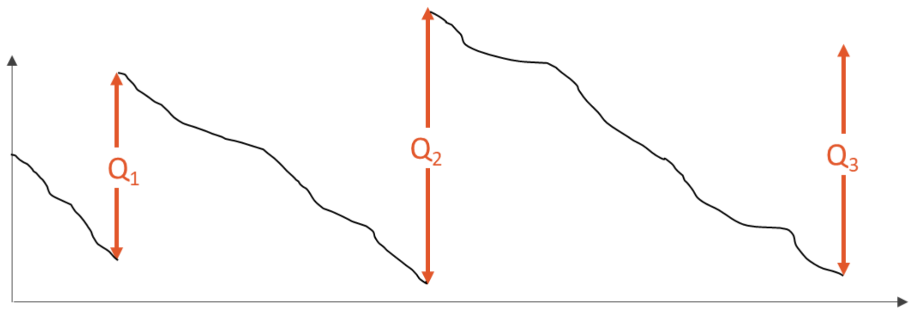
Questions

- **What would be the decision variables ? What would be their nature ?**
- **What would be the objective function ?**

Inventory management / Lot-sizing problem

Description

- You have to decide how much, and at which time, replenishing your inventory
- Your goal is to **satisfy the demand** (which can be deterministic or uncertain) at least cost



Variants

Supply	Procurement	Production: W/O changeover	Production: With changeover	
Demand	Steady-state demand	Dynamic demand	Uncertain demand: Shortage	Uncertain demand: Backlog
Leadtime	None	Deterministic	Uncertain	
Costs	Order cost	Holding cost	Supply cost	
Location	One single warehouse	Parallel warehouses	Sequence of warehouses	
Policy	Periodic	Threshold	Periodic + threshold	

Inventory management : The Economic Order Quantity (EOQ) problem

Problem statement	You are the manager of a warehouse. You know for sure the demand for the upcoming year, which is constant (global volume D). You want to minimize your fixed order costs, but without exploding your storage costs: What is x the optimal quantity to order at each order ?
Costs data	▸ Fixed order cost C_c ($ /order) ▸ Storage cost C_s ($ /average volume of storage on the year) ▸ Purchase cost C_p ($ /unit of purchased volume)

Inventory management : The Economic Order Quantity (EOQ) problem

Problem statement

You are the manager of a warehouse. You know for sure the demand for the upcoming year, which is constant (global volume D).

You want to minimize your fixed order costs, but without exploding your storage costs: What is x the optimal quantity to order at each order ?

Costs data

- Fixed order cost C_c (\$ /order)
- Storage cost C_s (\$ /average volume of storage on the year)
- Purchase cost C_p (\$ /unit of purchased volume)

Solution

The cost $f(x) = C_c \frac{D}{x} + \frac{C_s}{2} x$ and $f'(x) = -\frac{C_c D}{x^2} + \frac{C_s}{2}$

So : $f'(x) = 0 \quad \Rightarrow \quad 2 C_c D = c_s x^2 \quad \Rightarrow \quad x^* = \sqrt{\frac{2 C_c D}{C_s}}$

The second derivative at x^* is strictly positive so x is a local minima.

Scheduling is the allocation of shared resources over time to competing activities

Description

Machine scheduling problem :

- **Activities → jobs = a set of operations**
- **Resources → machines** that can process at most one operations at a time

Objective :

- Min makespan
- Min cost
- Max flowtime
- ....

Inputs

- A set of **jobs** & a set of **machines**
- Each job = a set of operations to be processed on dedicated machines(s) & their associated processing time
- **Various constraints** (precedence, changeover, …)

Scheduling

Outputs

An allocation of jobs x resources over time

= Assignment of each operation on a resource

+ Sequencing

+ Starting time

4 majors fields of applications of scheduling problems

1

Production

Flexible manufacturing

Assembly problems

With or without transportation

2

Computer science

CPU Scheduling

Mono or multi processor systems

3

Project management

Multi-resource scheduling

Precedence constraints

PERT

4

Workforce management

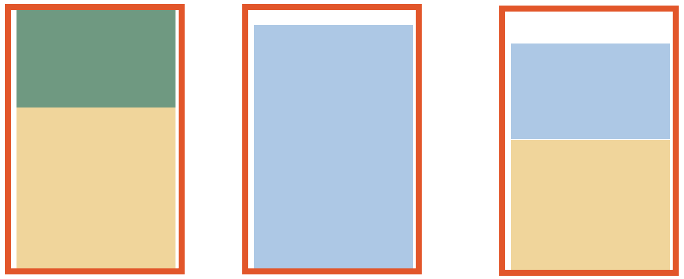
Timetabling scheduling problems (education, health, transport, …)

What's the difference between planning & scheduling ?

Planning

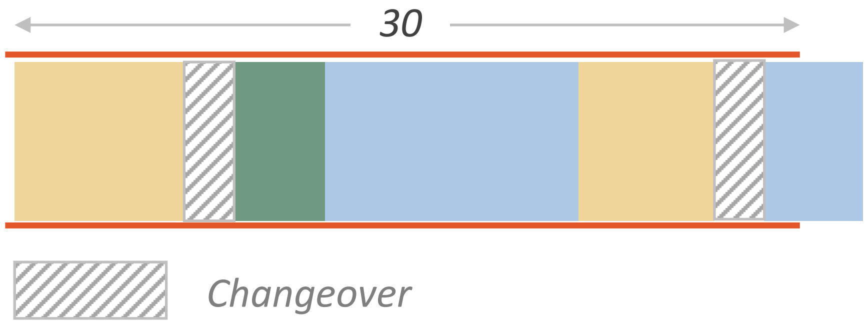
- **Tactical (mid-term)**
- **Discrete time**
- For each time step, look if the capacity is sufficient to complete a series of task
- "Think globally" = the task will be done during the time step but we don't know exactly when
- In the case where the order impacts the capacity to produce (typically: changeover) → risk of being over-optimist (~ constraint relaxation)
- Small time step implies better precision but higher computation time

3 time-steps of 10



Scheduling

- **Operational (short-term)**
- **Continuous time (**No need to discretize)
- Each task is allocated to the resource within a precise time slot
- Make it possible to consider changeover
- 2 key **decision variables** :
  - Loading = assignment to resources
  - Sequencing = order in which they are carried out (notion of sucessor and/or predecessor)



Key characteristics

Operations characteristics

- **Preemption or not**
- **Processing time**

Machine type

- **Identical**
- **Uniform (different yield)**
- **Homogeneous**

Constraints

- **Earliest date or deadline**
- **Changeover or setup**
- **Same machine**
- **....**

Operations per job

- **Single**
- **Ordered**
- **Precedence** (represented through an acyclic graph)
- **Non ordered** = totally free

Work process = Possible machine per operations

- **Only one dedicated machine**
- **Several but with different processing time**
- **Several with same processing time**

Objective

Let C_i be the completion time of job J_i : objective = f(C)

- **Cost function**
- **Makespan**
- **Weighted flowtime = $\Sigma_i w_i C_i$**
- **…**

A simple scheduling example

Jobs

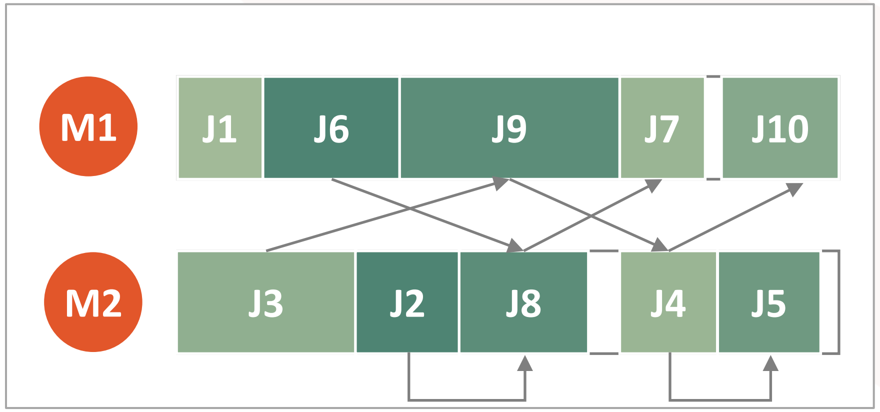
- 10 jobs to allocate either on M1, or on M2
- Each job j can be done on the two machines but with different duration d_{1j} or d_{2j}

Machine

M1 M2

Constraints

- Non preemptive
- Precedence constraints



Scheduling : 3 possible formulations as a MIP

- j=1,…,J jobs of duration *d[j]*
- m=1,…,M machines
- t=1,…,T time step

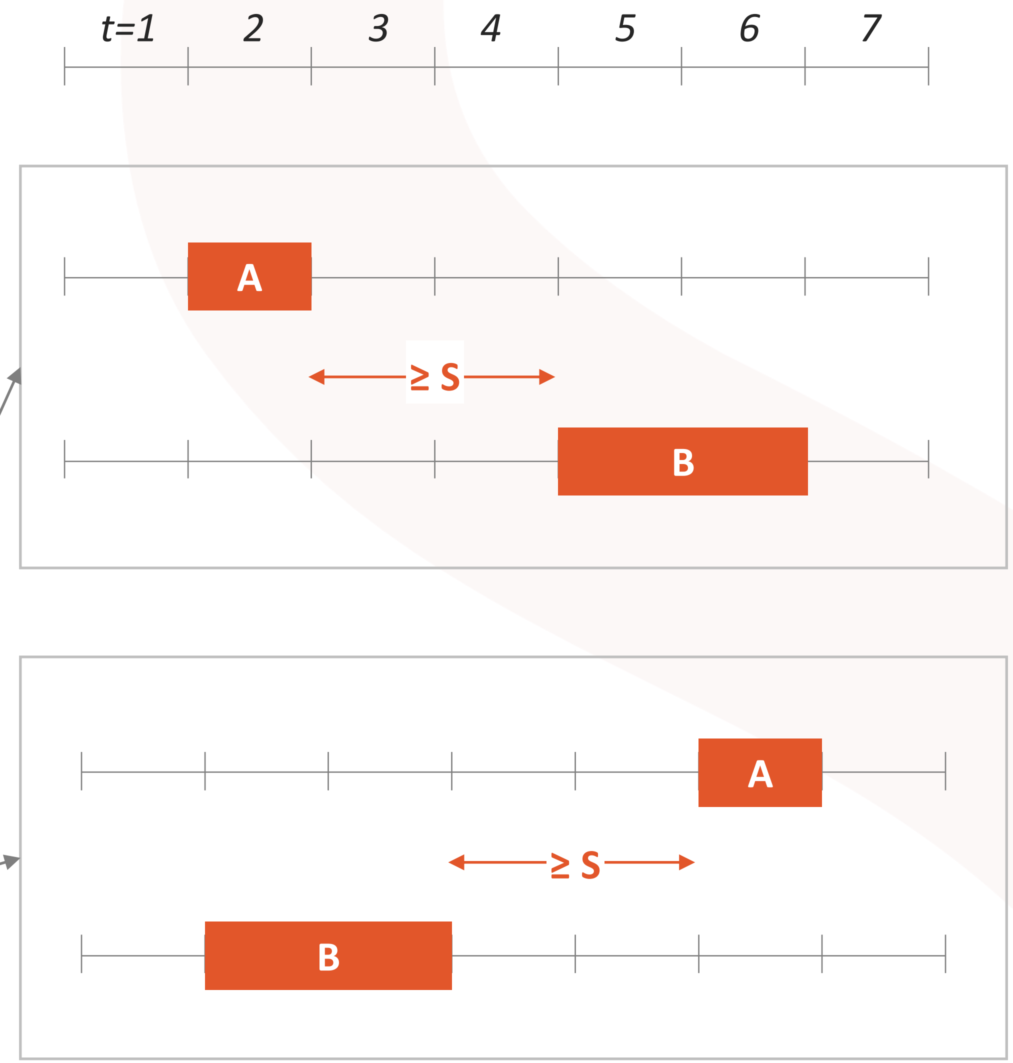
	Variables	Key constraints
Time-indexed	• **X[t][j][m] in {0,1}** *#equal to 1 if job j starts at time t at machine i.*	• Σ[j][t–*d[j][m]* +1 ≤ t' ≤ t] X[t][j][m] ≤ 1 *# for all j, m*
Rank-indexed	• **X[j][m][k] in {0,1}** #equal to 1 if job j is scheduled at the k-th position on machine I • **Y[m][k]** #start time of the job at the k-th position of machine i	• Y[m][k] + Σ[j] *d[j][m]* X[j][m][k] ≤ Y[m][k+1] *# for all m, k*
Precedence	• **X[j][j'] [m] in {0,1}** *#equal to 1 if j precedes j' on machine m* • **Y[j][m]** *#start time of job j on machine m*	• Y[j][m] + *d[j][m]* ≤ Y[j'][m] + *M* X[j][j'][m] *#for all j, j', m*

Disjunctive constraints

- **Problem statement** : 2 events A and B, of duration d_A and d_B, to schedule with a minimum space S :
- **Decision variables :**
  - A[t] = 1 if the event A starts at t
  - B[t] = 1 if the event B starts at t
- **Starting dates** express as :
  - Σ_t t A[t]
  - Σ_t t B[t]

Minimal space S constraint :

- Either **A is before B** then : $\Sigma_t\, t\, B[t] - (\Sigma_t\, t\, A[t] + d_A) \geq S$
- Or **A is after B** then : $\Sigma_t\, t\, A[t] - (\Sigma_t\, t\, B[t] + d_B) \geq S$



Disjunctive constraints : 3 possible formulations

	Variables	Key constraints
The most natural (BigM)	• X in {0,1} equals 0 if A is before B	• Σ_t t B[t] - Σ_t t A[t] ≥ d_A + S – M X • Σ_t t A[t] - Σ_t t B[t] ≥ d_B + S – M(1-X)
2 to 2 exclusion		• A[t] + B[t’] ≤ 1, for all (t, t’) that violate the space constraint
Quadratic		• **Either** Σ_t t B[t] - Σ_t t A[t] - d_A - S ≥ 0 **and** Σ_t t A[t] - Σ_t t B[t] - d_B - S ≤ 0 • **Or** Σ_t t B[t] - Σ_t t A[t] - d_A - S ≤ 0 **and** Σ_t t A[t] - Σ_t t B[t] - d_B - S ≥ 0 It suffices to impose that the product be negative : **(Σ_t t B[t] - Σ_t t A[t] - d_A - S) (Σ_t t A[t] - Σ_t t B[t] - d_B - S) ≤ 0**

Africa Business School

Multi-criteria optimization

Which criteria to guide the decision ?

Quantitative

Qualitative

Various KPIs

Margin

Service level

Safety

Risk

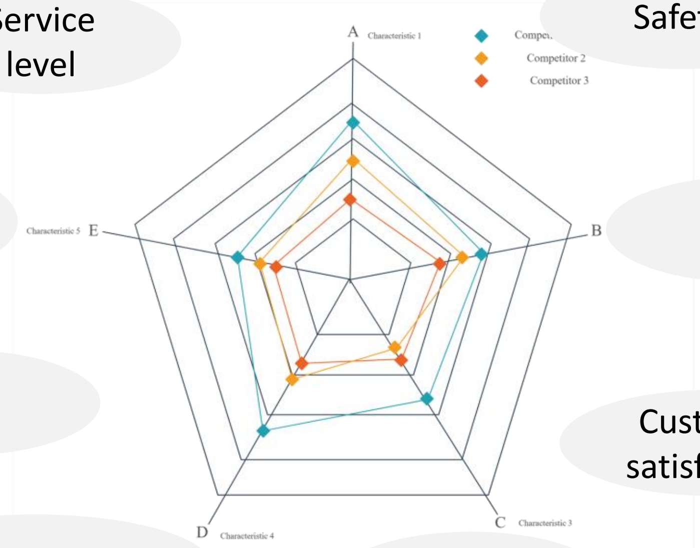
Delay

Cost

Customer satisfaction

Sustainability

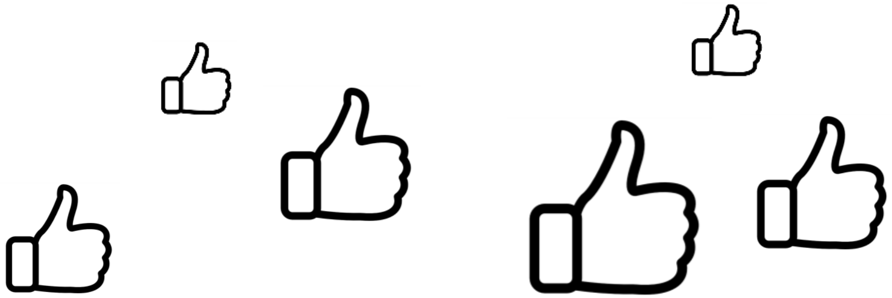
Revenue



Time horizon

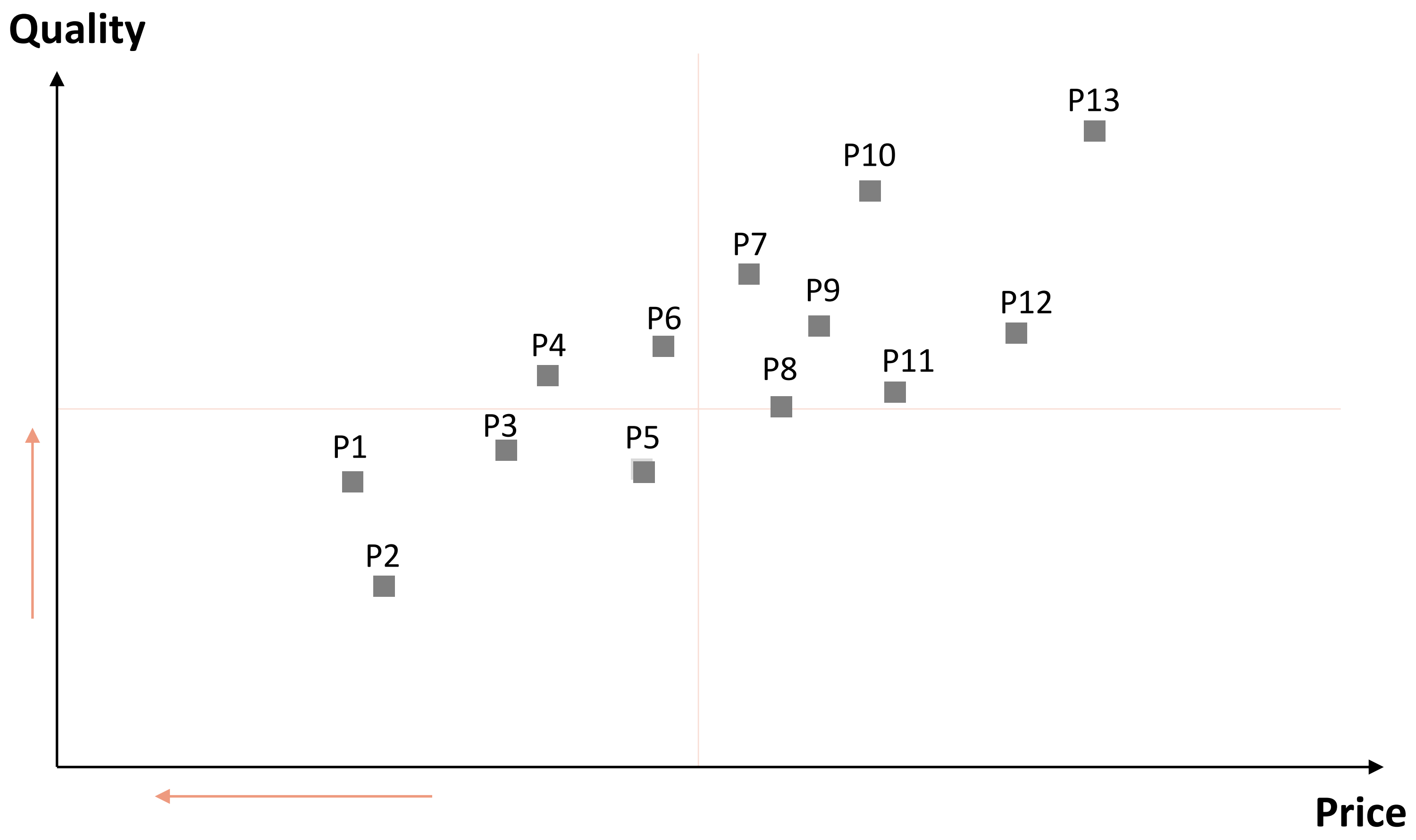
Short-term → Mid-term → Long-term

Several people preferences



Example with 2 criteria

You have to buy a product. On the market, you find 13 offers (P1, … P13) that differs in price and quality as represented on the following graph :

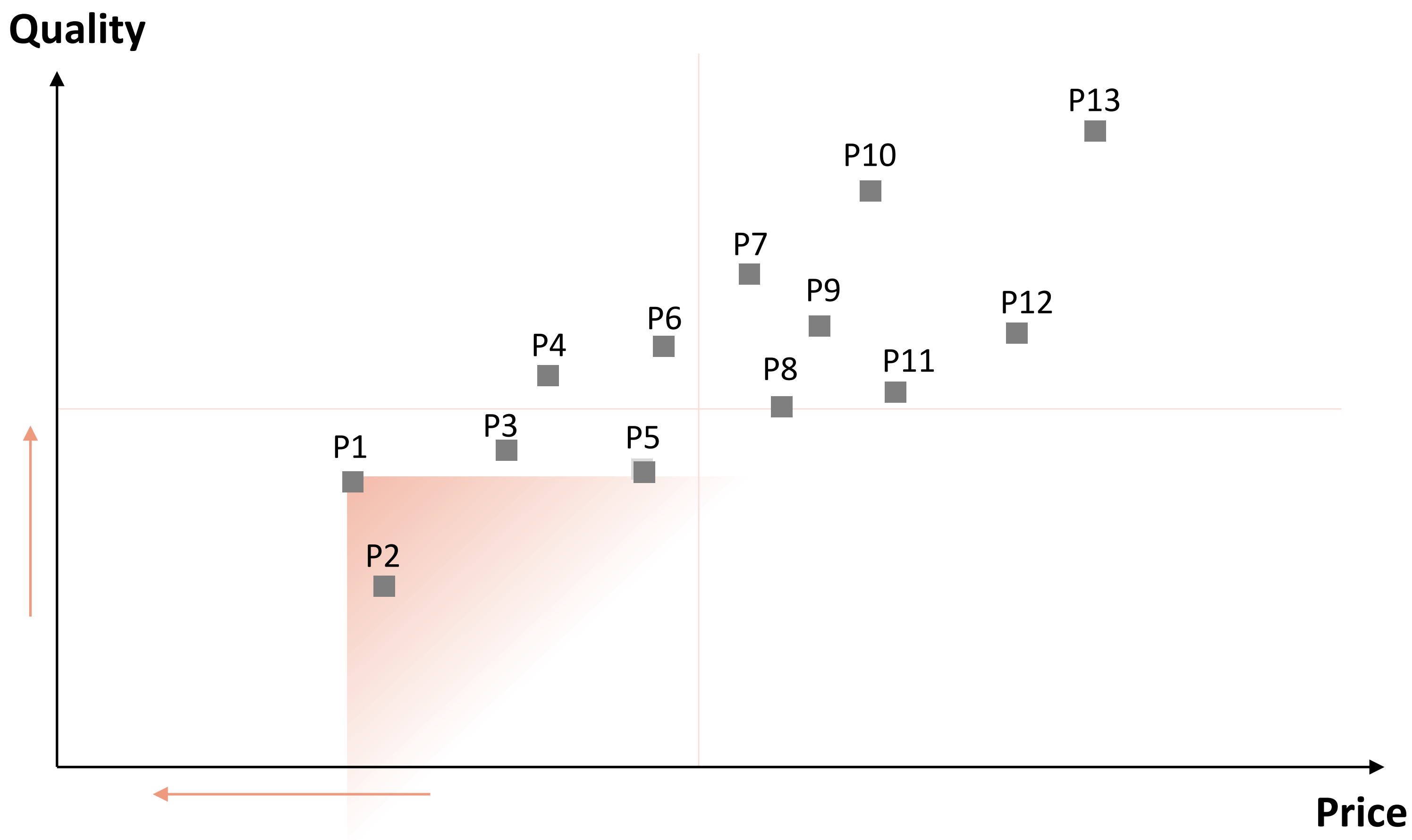


2 questions :

- Which product would you choose ?
- In which quadrant would be your ideal solution ?

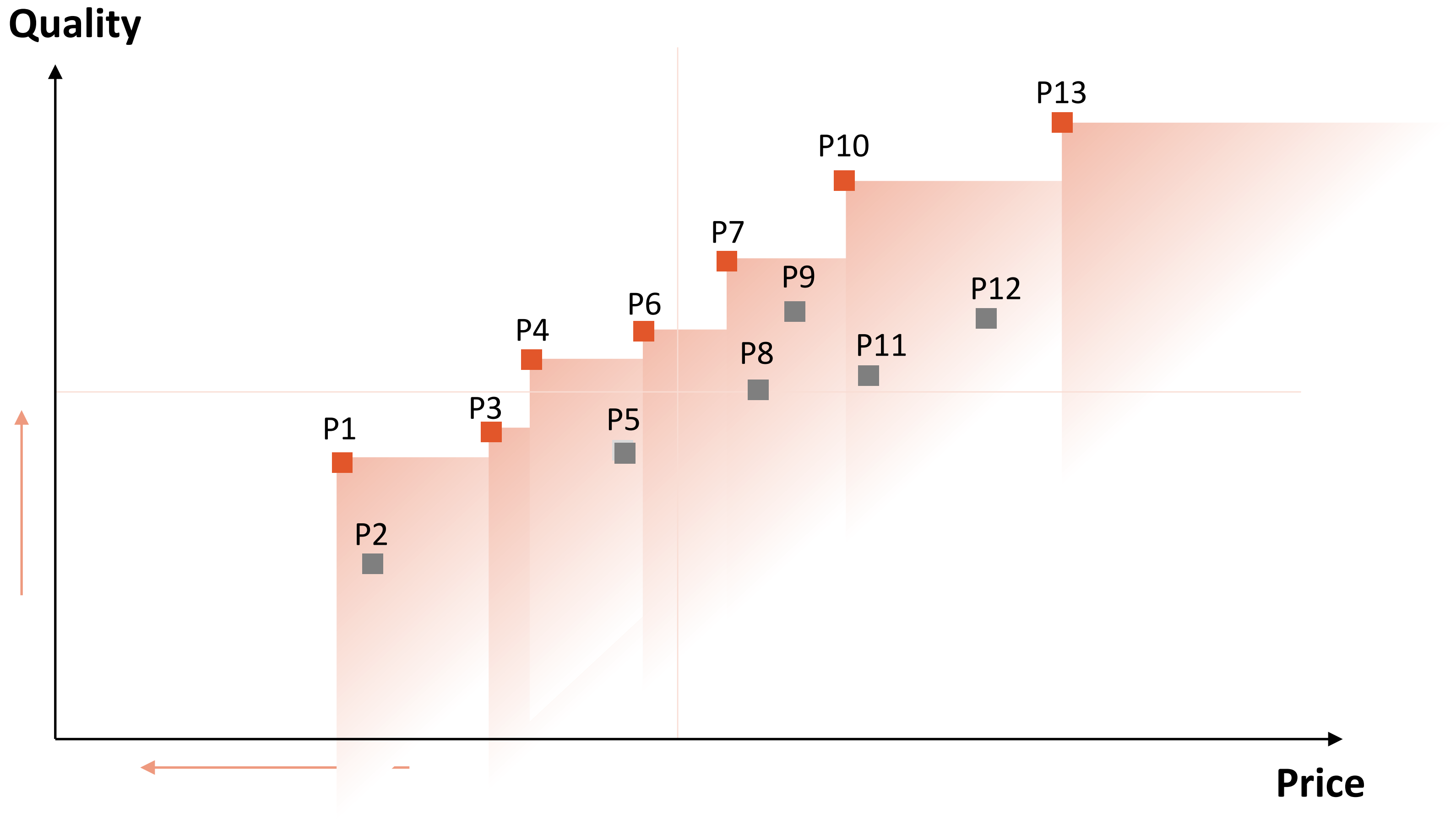
First, we remove the solutions that are dominated by another one

P1 dominates P2 since $\text{quality}_1 > \text{quality}_2$ and $\text{price}_1 < \text{price}_2$

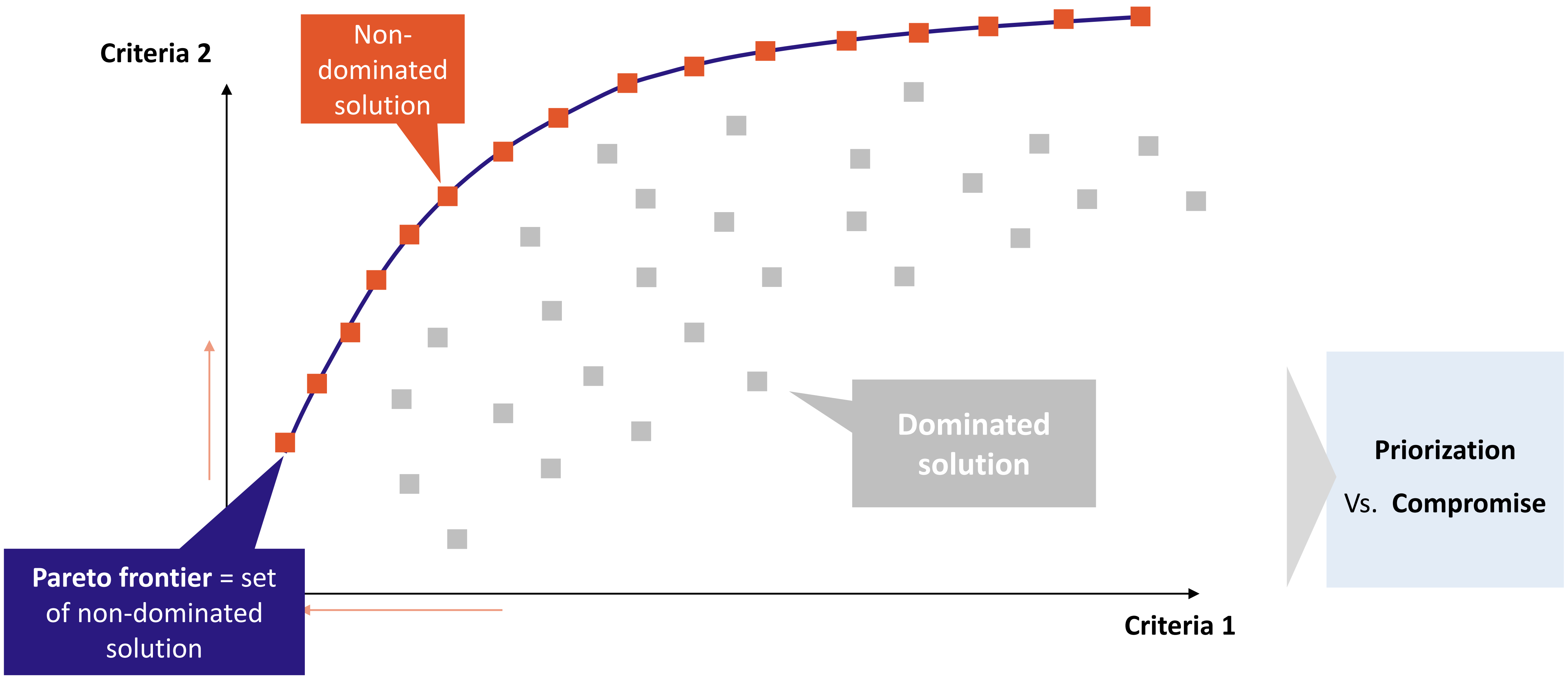


First, we remove the solutions that are dominated by another one

Pi dominates Pi if and only if **$quality_i > quality_j$** and **$price_i < price_j$**

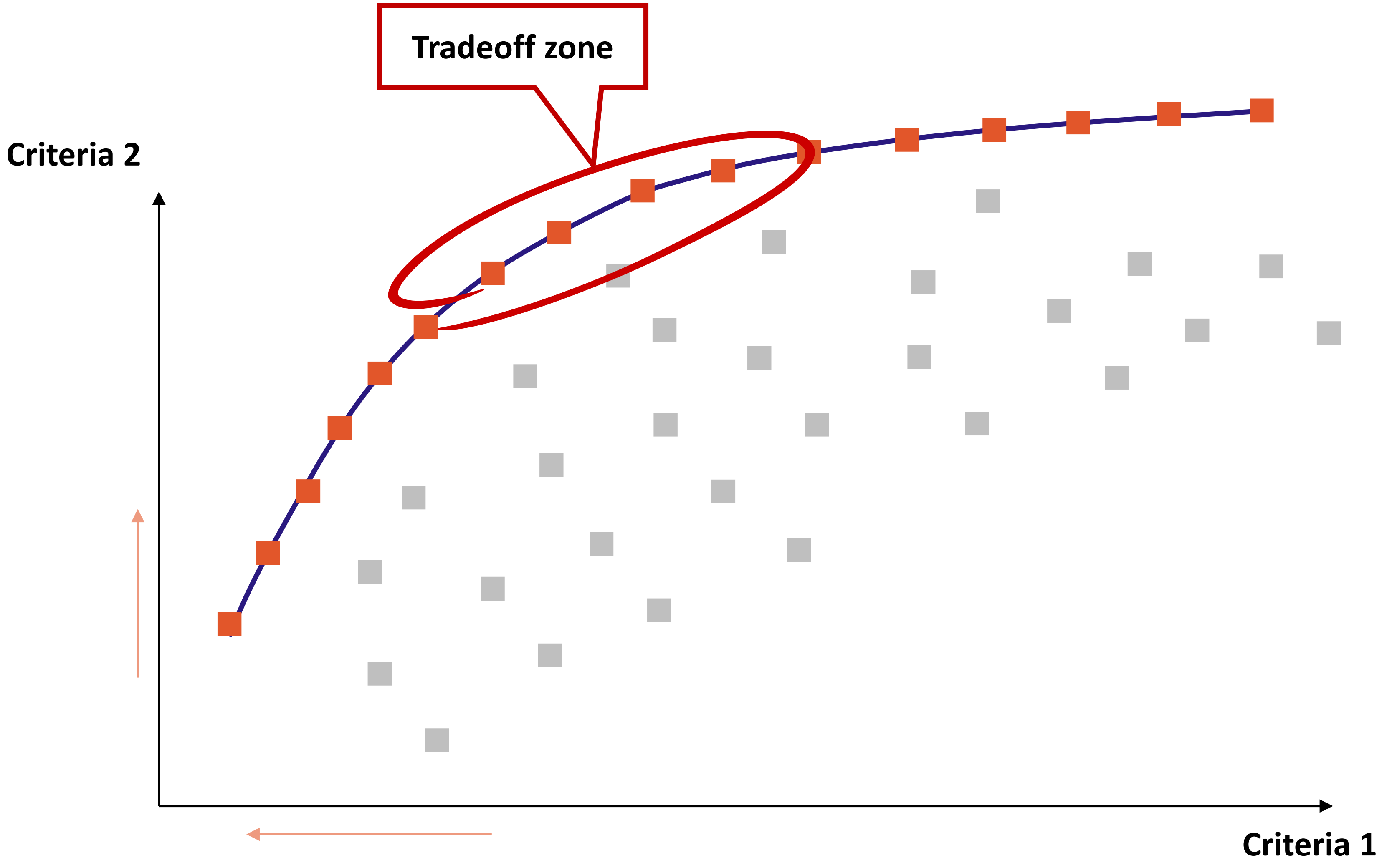


The set of non-dominated solutions form the Pareto frontier

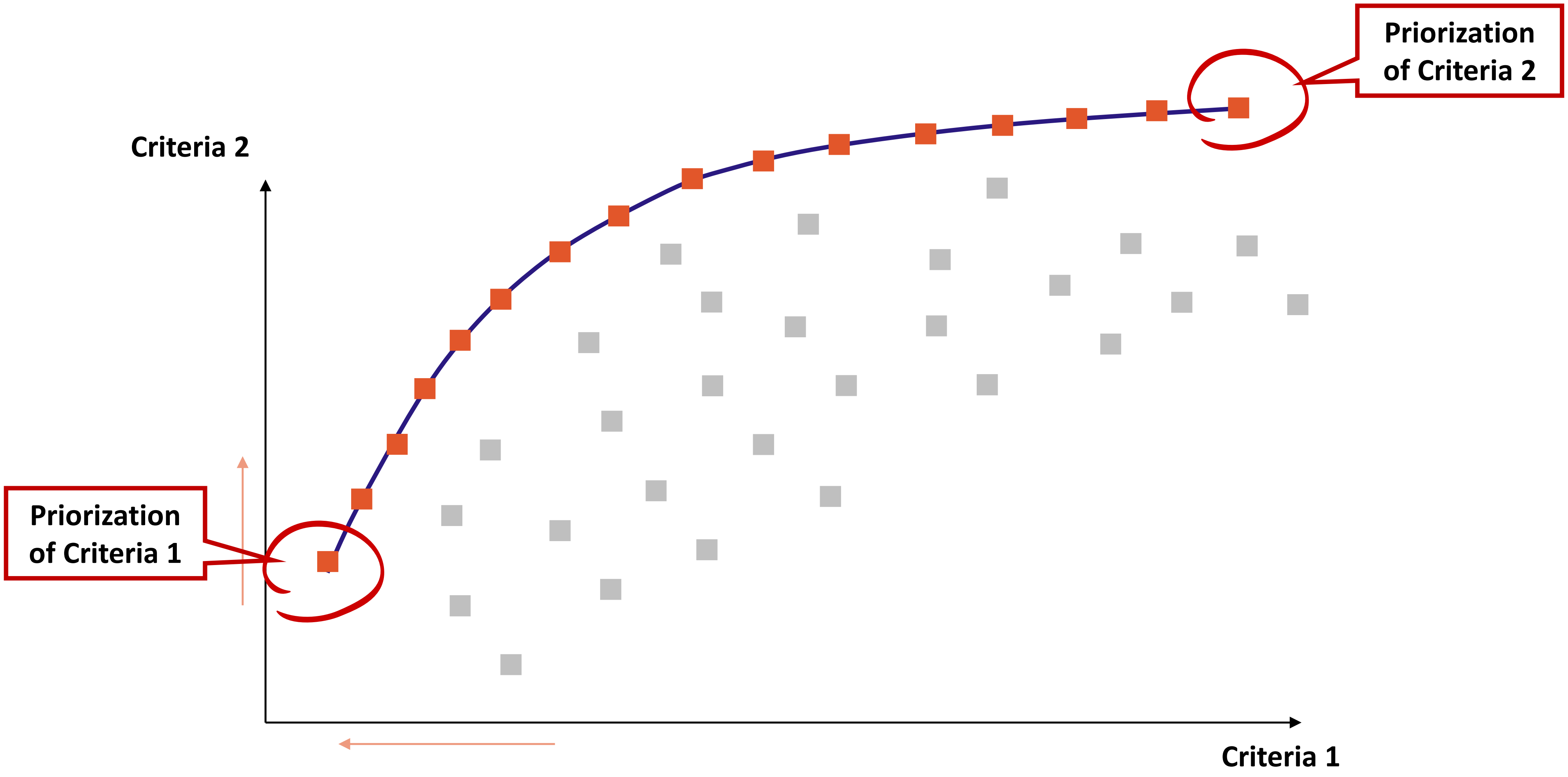


Generally, this set is hard to identify

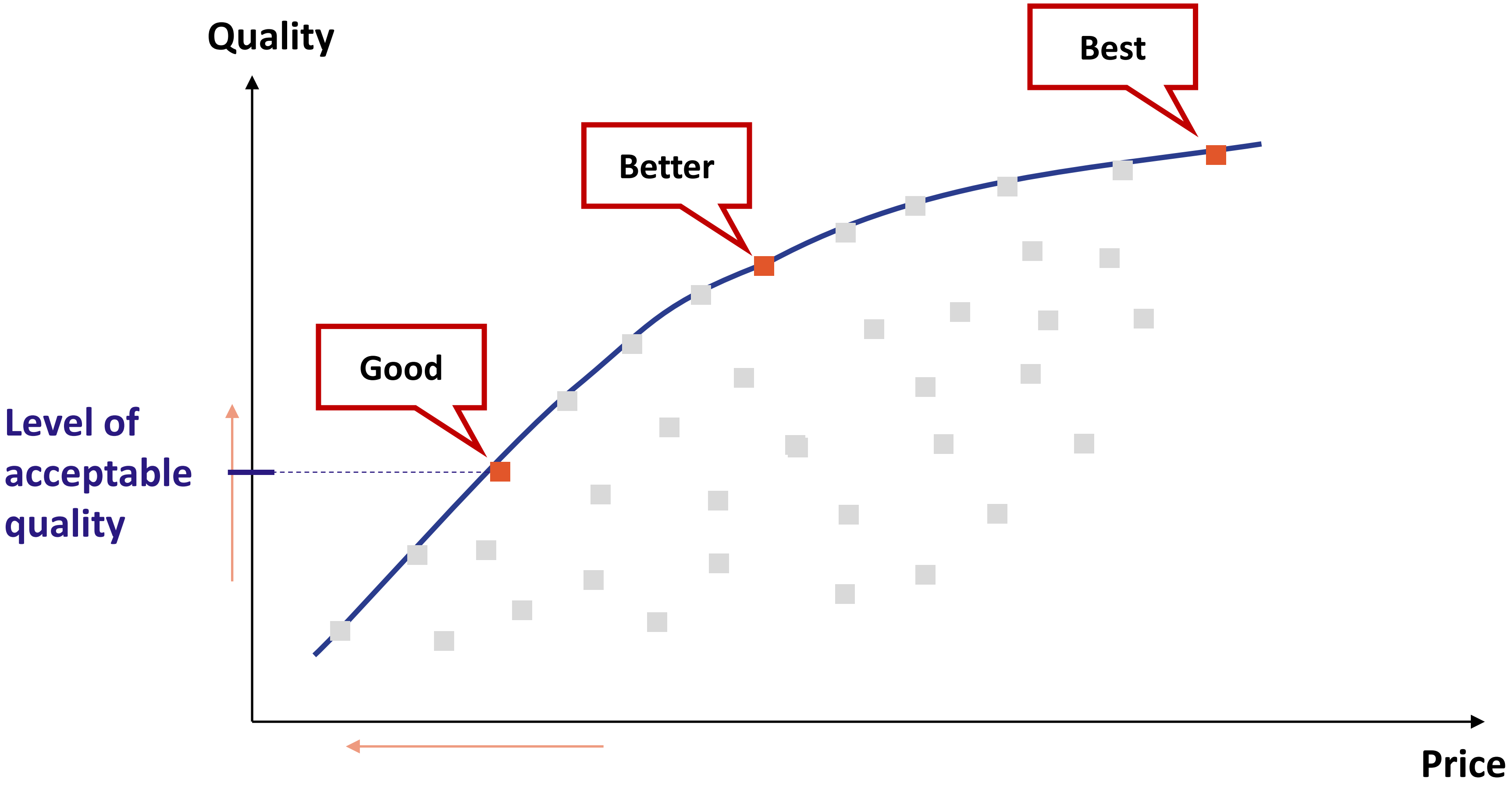
Then select your best solution among the Pareto frontier



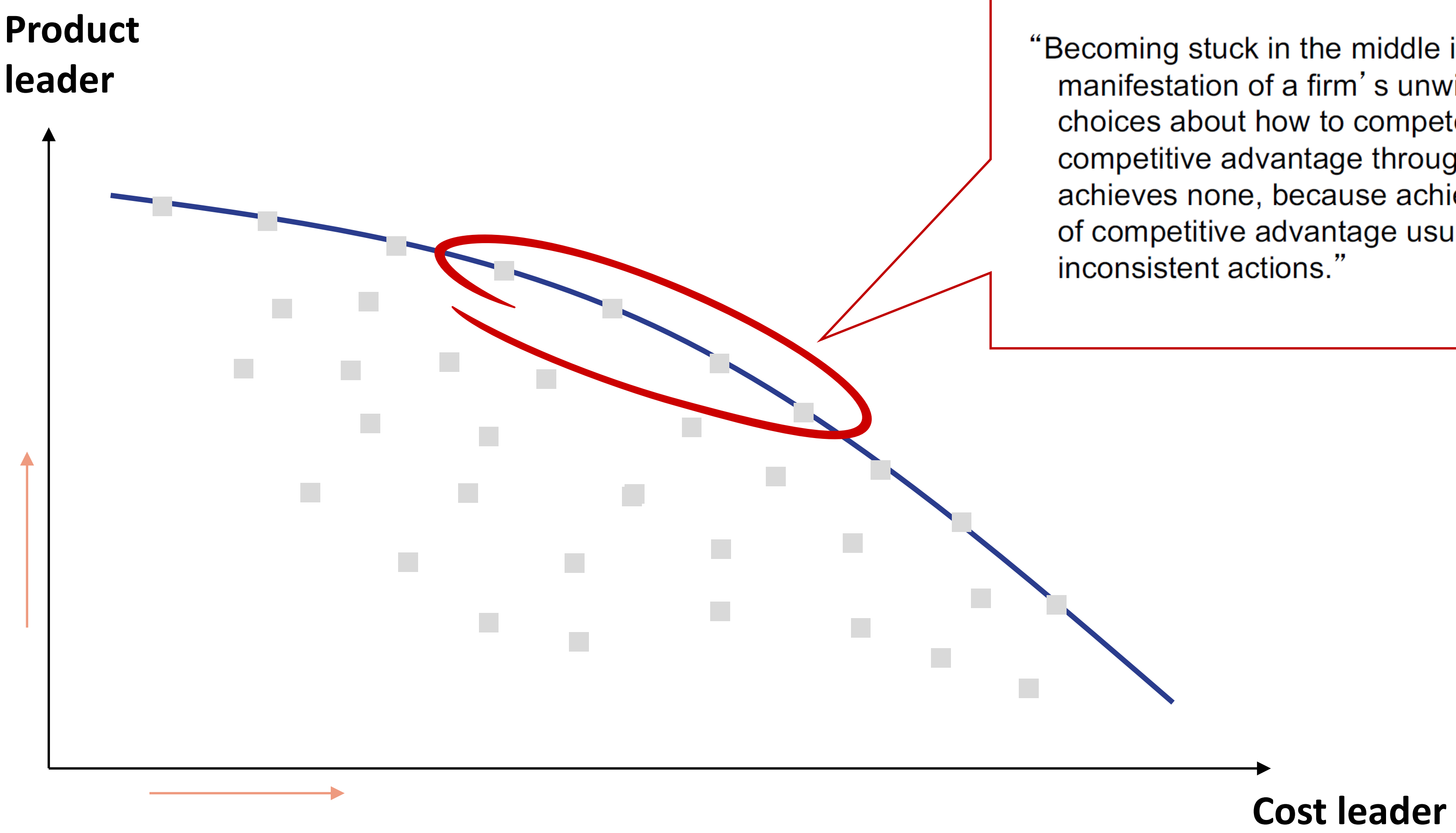
Then select your best solution among the Pareto frontier



The « Good-Better-Best » pricing strategy from a multi-criteria perspective



Sometimes prioritizing is better than compromising
Strategy example

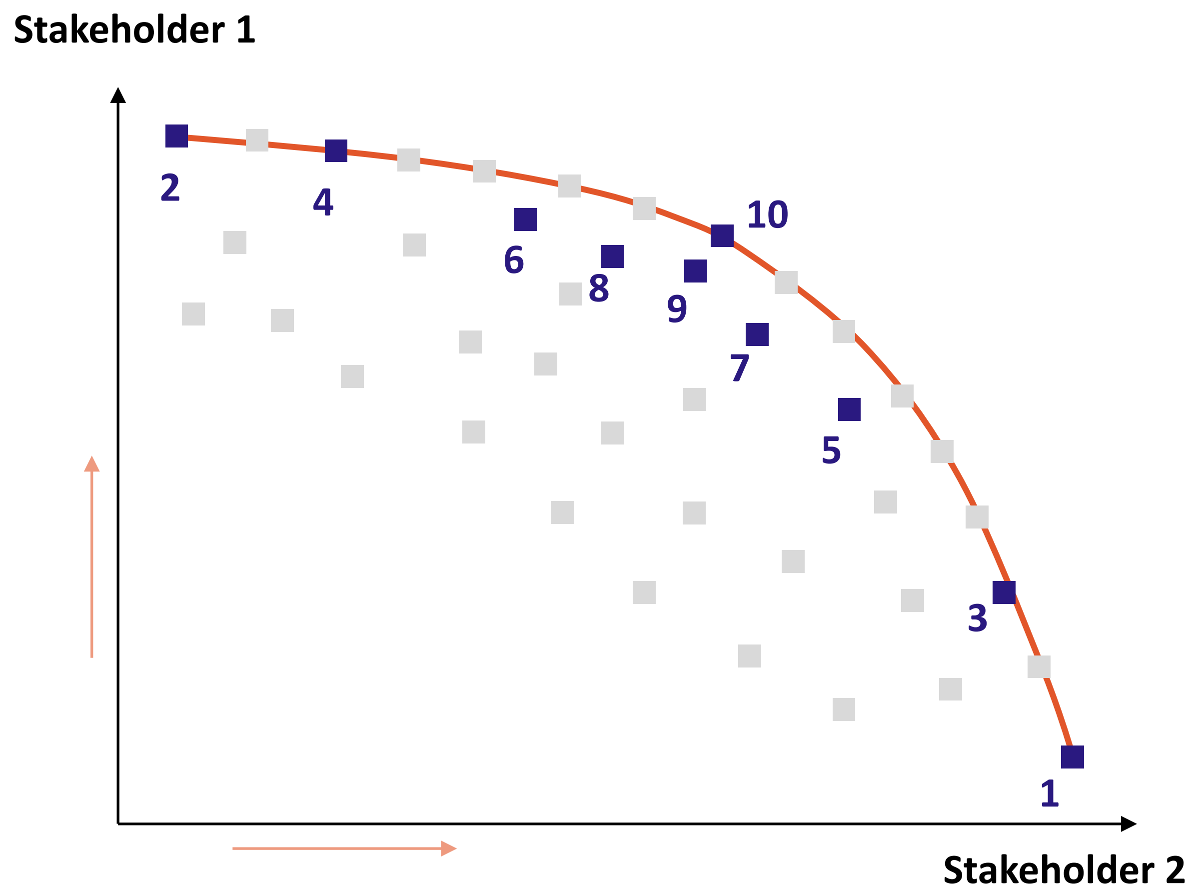


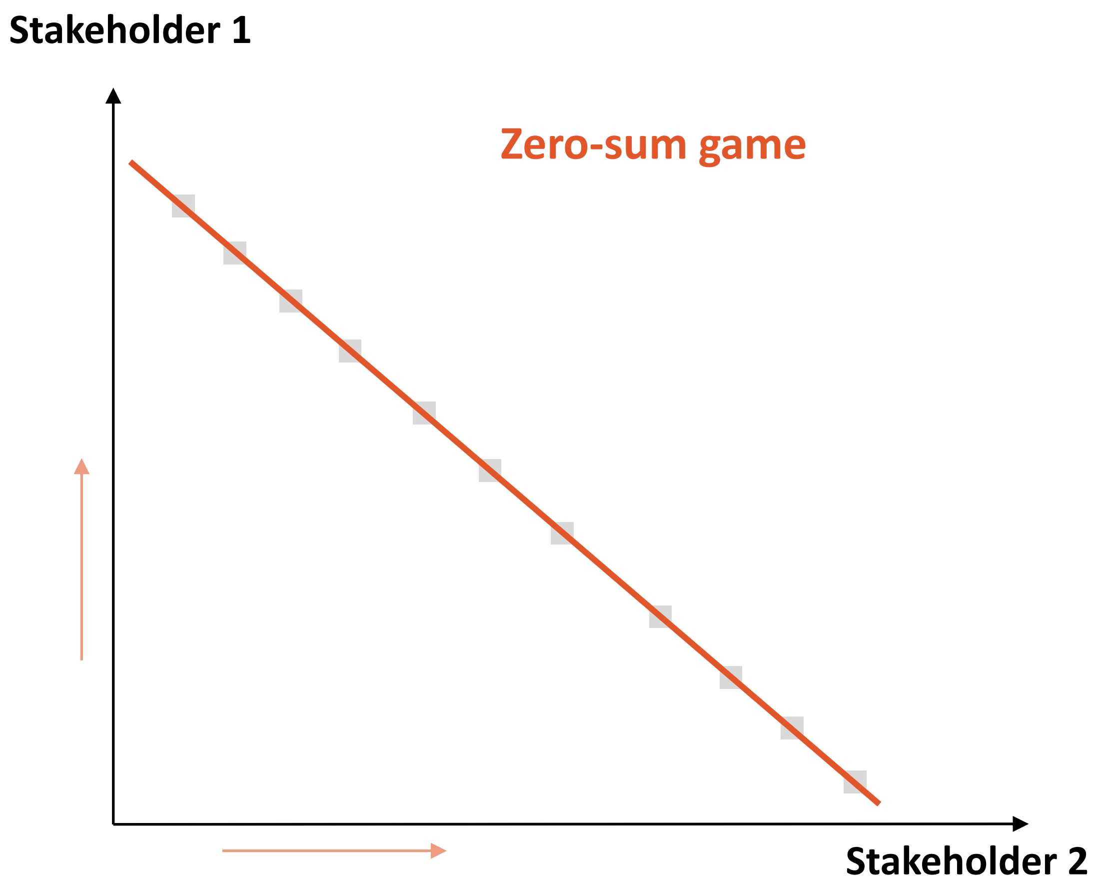
“Stuck in the Middle”

“Becoming stuck in the middle is often a manifestation of a firm’s unwillingness to make choices about how to compete. It tries for competitive advantage through every means and achieves none, because achieving different types of competitive advantage usually requires inconsistent actions.”

Michael Porter

Negociation aims to explore this field and to identify a non-dominated tradeoff





How to deal with multiple criteria ?

Multiple criteria to optimize

- **Identify the Pareto frontier**
  1. **Let the user choose** — *Not compatible with automation*
  2. **Convert criterias as constraints** — *One of the criteria will be kept as a criteria, the other are converted as constraints*
  3. **Successive optimizations** — *You start by optimizing over a criteria, then you optimize on a second one, by contraining the model to keep the same value on the first one, or slightly worsening it*
- **Combine the criteria within a « Meta criteria »**
  4. **Weighted sum** — *Weight the different criteria with weights that translates the decision maker preferences*
  5. **Other combinations…** — *Combine them through a different function*

How to consider multiple objective in an optimization problem ?

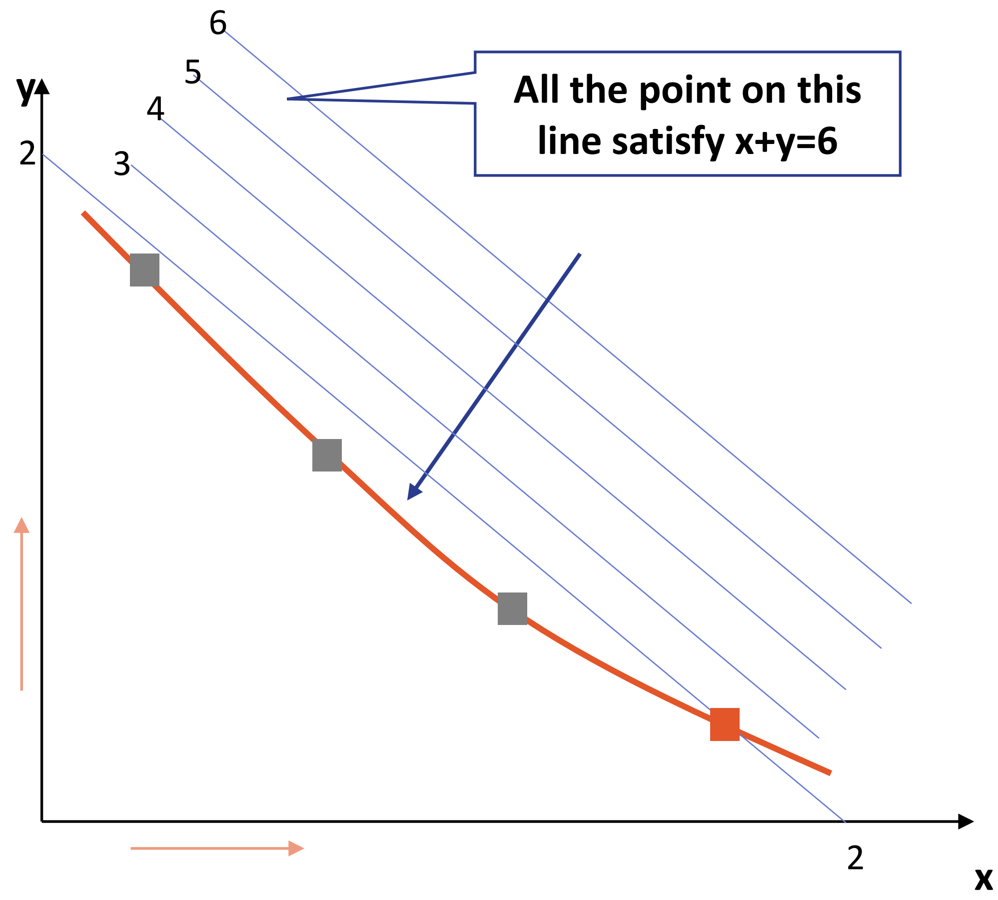
2 Convert criterias as constraints

3 Successive optimizations

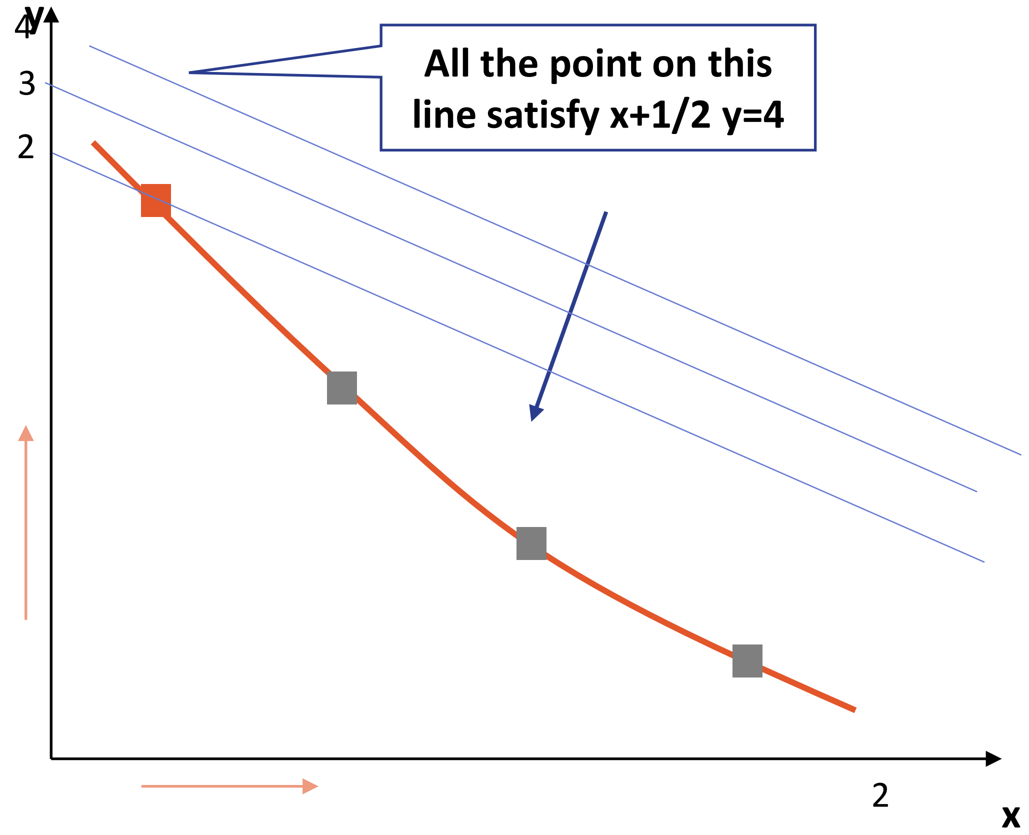
Using a weighted sum can be risky

Due to the convex shape of the curve, midpoints can not be reached

Ex : Criteria = w_1 x + w_2 y with $w_1 = w_2 = 1$



Ex : Criteria = w_1 x + w_2 y with w_1 = 1 et w_2 = 1/2

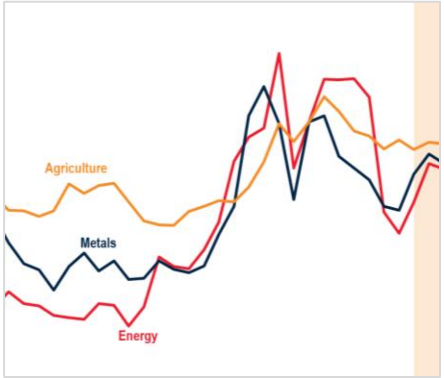


Optimization under uncertainty

Uncertainty is everywhere …

“To be uncertain is to be uncomfortable, but to be certain is to be ridiculous.”
Chinese proverb*

Weather



Prices

Failure

Customers’ behaviors, competitors reactions, R&D success, …

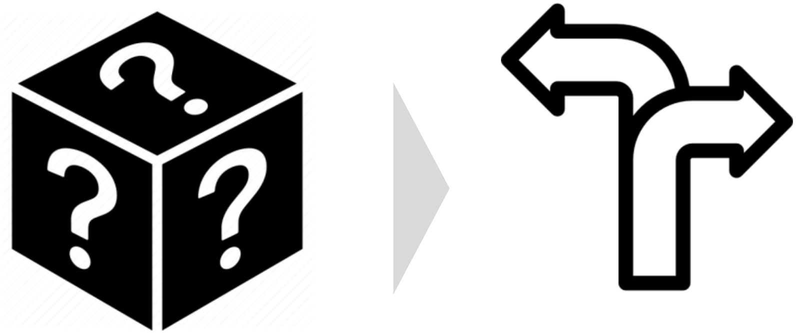
And so many others

… making rational decision-making
(and therefore optimization) much harder

** From A. Ben-Tal and L. El-Ghaoui. Robust Optimization. Princeton University Press, 2009.*

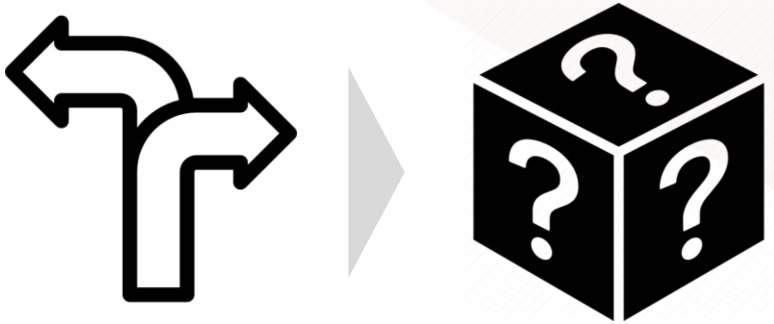
Decision under uncertainty : do I really have to make my decision before knowing what will happen ?

Cas 1 : hazard-decision



Deciding not to decide right now may have a cost

Cas 2 : decision-hazard



Sometime we have to decide right now

In the case « decision – hazard », 2 possible ways of making decisions

I take a bet on the future

I take a bet on one scenario (the most probable, or the expected) and **I make my decision based this scenario**

I don't even consider what might happen in other scenarios.
If my prediction fails, I'll loose a lot

« I believe I know »

Back to deterministic decision-making

The effort focuses on identifying the right scenario, through intelligence, forecasts, …

I consider all the possibilities in my decision

I optimize in order to **min-aximize the expected outcomes or another risk measures (standard deviation, …)**

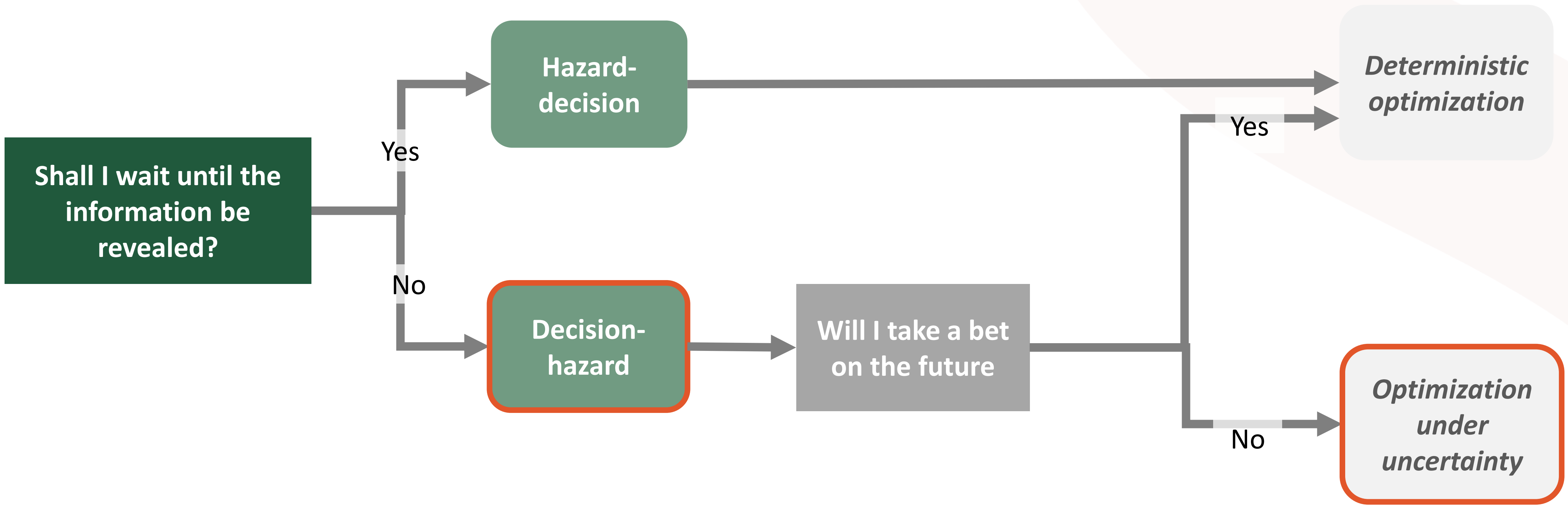
I consider all the possibility in my decision
I aim at remaining close to the optimum regardless of the scenario that finally occurs

« I know that I don't know »

Welcome to stochastic & robust optimization

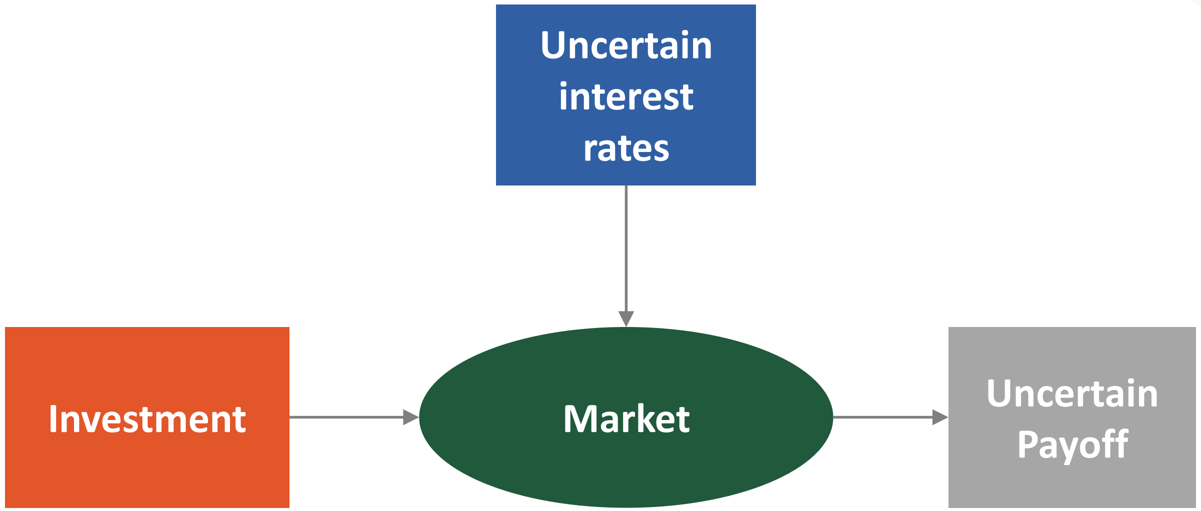
The effort focuses on structuring the available information (without extrapolating it) and using all this information for common decision-making

In the case « decision – hazard », 2 possible configurations



Typical example : the portfolio problem

You're an investor and you have to choose between investing in stocks, bonds or the money market



How considering uncertainties in an optimization problem ?

Generic framework

Consider a classical decision problem	**Max** Criteria (x) *Over the set of possible x*		
	Uncertainty ξ		
Add uncertainties ξ in data	**Max** Criteria (x, ξ) *Over the set of possible x*		**Ill-defined**
	Measure H		
Get rid of the uncertainty by applying a risk measure H	**Max** H(Criteria (x, ξ)) = f(x) *Over the set of possible x*		**Well-defined**

Investment example

Max Payoff (x)
For x in {stock, bond, money market)

Uncertain interest rates

Max Payoff(x, interestRates)
For x in {stock, bond, money market)

Measure H

Max EV[Payoff (x, interestRates)]
For x in {stock, bond, money market)

More generally, how to consider uncertainty in an optimization problem ?

Reminder : what is a mathematical program ?

$$\begin{cases} \text{Min} \quad f_0(x) \\ \text{S.t.} \quad f_1(x) \leq 0 \\ \quad\vdots \\ \quad f_m(x) = 0 \\ \boldsymbol{x} \in \mathbb{R}^{\boldsymbol{n}} \end{cases}$$

Consider uncertainties in data

$$\begin{cases} \text{Min} \quad f_0(x,\xi) \\ \text{S.t.} \quad f_1(x,\xi) \leq 0 \\ \quad\vdots \\ \quad f_m(x,\xi) = 0 \\ \boldsymbol{x} \in \mathbb{R}^{\boldsymbol{n}}, \xi \in \mathbb{R}^{\boldsymbol{p}} \end{cases}$$

Ill-defined

Operators $h_0,\ldots,h_m$

3 criteria for the choice of h_j:

- Reflect decision-maker's risk profile
- Make the best use of our knowledge of ξ
- Lead to a tractable deterministic counterpart

Determine its deterministic counterpart :

$$\begin{cases} \text{Min} \quad g_0(x) = h_0(f_0(x,\xi)) \\ \text{S.t.} \quad g_1(x) = h_1(f_1(x,\xi)) \leq 0 \\ \quad\vdots \\ \quad g_m(x) = h_m(f_m(x,\xi)) = 0 \\ \boldsymbol{x} \in \mathbb{R}^{\boldsymbol{n}}, \xi \in \mathbb{R}^{\boldsymbol{p}} \end{cases}$$

Well-defined

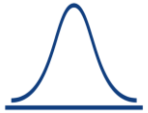
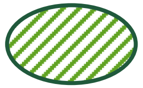
Ex : if the uncertainty is represented by a discrete set of scenarios

You're an investor and you have to choose between investing in stocks, bonds or the money market :

		Scenarios			Possible criterias		
		Interest rates rise	Static rates	Interest rates fall	Expected payoff*	Worst payoff**	Variance*
Probability		20%	50%	30%			
Possible decisions	Stocks	-4	4	12	$\boxed{4,8}$	-4	31,4
	Bonds	-2	3	8	3,5	-2	12,3
	Money market	3	2	1	1,9	$\boxed{1}$	$\boxed{0,5}$

** Requires probability*   *** Do not require probability*

This leads to two commonly used paradigms to consider uncertainty in a decision problem

	 Stochastic optimization	 Robust optimization
Available information	Probability distribution of ξ	Support = all possible values for ξ
Risk measure	**H = Expected value + Quantile**	**H = worst-case**
Philosophy	**Average approach** *Requires a good knowledge of the uncertainties*	**Worst-case approach** *Very conservative*
Typical application field	 Finance	 Engineering

Chance-constraint or probability constraints : satisfying a constraint with a given probability

Deterministic constraint	$f(x, \xi) \leq a$
Chance-constraint (or probability constraint)	$P_{\xi}[\, f(x, \xi) \leq a \,] \geq 1-\varepsilon$ Or equivalently : $P_{\xi}[\, f(x) > a \,] \leq \varepsilon$
Example	P [power shortout] ≤ 1%

How to formulate a constraint in an uncertain optimization problem where the uncertainty is represented by a set of scenarios

Problem statement

Consider the constraint : $f(x, \xi) \leq a$

With ξ in $\{\xi_s\}_{s=1, \ldots, n}$ → n possible scenarios for ξ with probability p_s, s = 1, …, n

Robust constraint

Satisfy the constraint in the worst-case ≡ Satisfy the constraint on all cases :

$$f(x, \xi_s) \leq a, \quad \text{for all } s=1, \ldots, n$$

Chance-constraint (or probability constraint)

$$P_\xi\,[\, f(x) > a \,] \leq \varepsilon$$

Define a binary variable per scenario y_s such that the non-satisfaction of the constraint on this scenario implies $y_s=1$ (BigM constraint) :

$$f(x, \xi_s) \leq a + M\, y_s$$

Then limit the number of y_s =1 in order to satisfy the required probability :

$$\Sigma_s\, p_s\, y_s \leq \varepsilon$$

3 choices to make

How to model uncertainty ?

- Continuous distribution
- Discrete distribution (scenarios)
- Fuzzy logic

Preferences of the decision maker

- Risk-neutral
- Risk-averse
- Risk-lovers

How to deal with uncertain constraints

- Penalisation
- Chance constraints
- Worst-case
- Recourse action

The efficient portfolio

The efficient portfolio

Problem statement

- You want to invest your budget (a total amount of 1) into different assets and you're looking for the optimal fraction of each asset.
- The rate of return of asset i is a random variable with expected value m_i
- Assets are not independent, as quantified by the covariance matrix C
- Your objective function combines the expected rate of return and a risk indicator.

Questions

- Formulate the problem as a 2-criteria optimization problem, combining the expected return and the variance as risk indicator
- How do you compute the variance of the portfolio return ?
- How could you determine the efficient frontier ? *Hint : apply KKT*

Refresher

$$\mathrm{Var}(aX + bY) = a^2\mathrm{Var}(X) + b^2\mathrm{Var}(Y) + 2ab\mathrm{Cov}(X, Y)$$

The efficient portfolio - Solution

Problem statement

Let C the covariance matrix and m the mean returns of stocks. We are looking for a solution that satisfies two criteria :

- Minimize the risk, computed as $x^T Cx$
- Maximize the return, computed as $m^T x$

Problem formulation

In order to explore the efficient frontier, we convert one of the criteria as a constraint, and therefore solve the following problem for different values of m_{target} :

$$(P) \begin{cases} \min\limits_{x} & x^T Cx \\ \text{s.t.} & e^T x = 1, \\ & m^T x \geq m_{target} \\ & x \in \mathbb{R}_+^n \end{cases}$$

The efficient portfolio – Step 1

The linear problem

In order to define the range for m, we start by looking for the best possible value of m (independantly of the risk criteria) :

$$(P)\begin{cases} \max\limits_{x} & m^T x \\ \text{s.t.} & e^T x = 1, \\ & x \in \mathbb{R}_+^n \end{cases}$$

KKT conditions

Stationarity conditions :

x is optimal if and only if there exists $\mu \in \mathbb{R}$ and $\lambda \in \mathbb{R}_+^n$ such that :
$\nabla_x L(x, \lambda, \mu) = 0$, with $L(x, \lambda, \mu) = -m^T x + \mu(e^T x) - \lambda^T x$.

$\nabla_{x_i} L(x, \lambda, \mu) = -m_i + \mu - \lambda_i = 0 \Leftrightarrow -m_i + \mu = \lambda_i \geq 0$ so $\mu \geq m_i, \forall i$

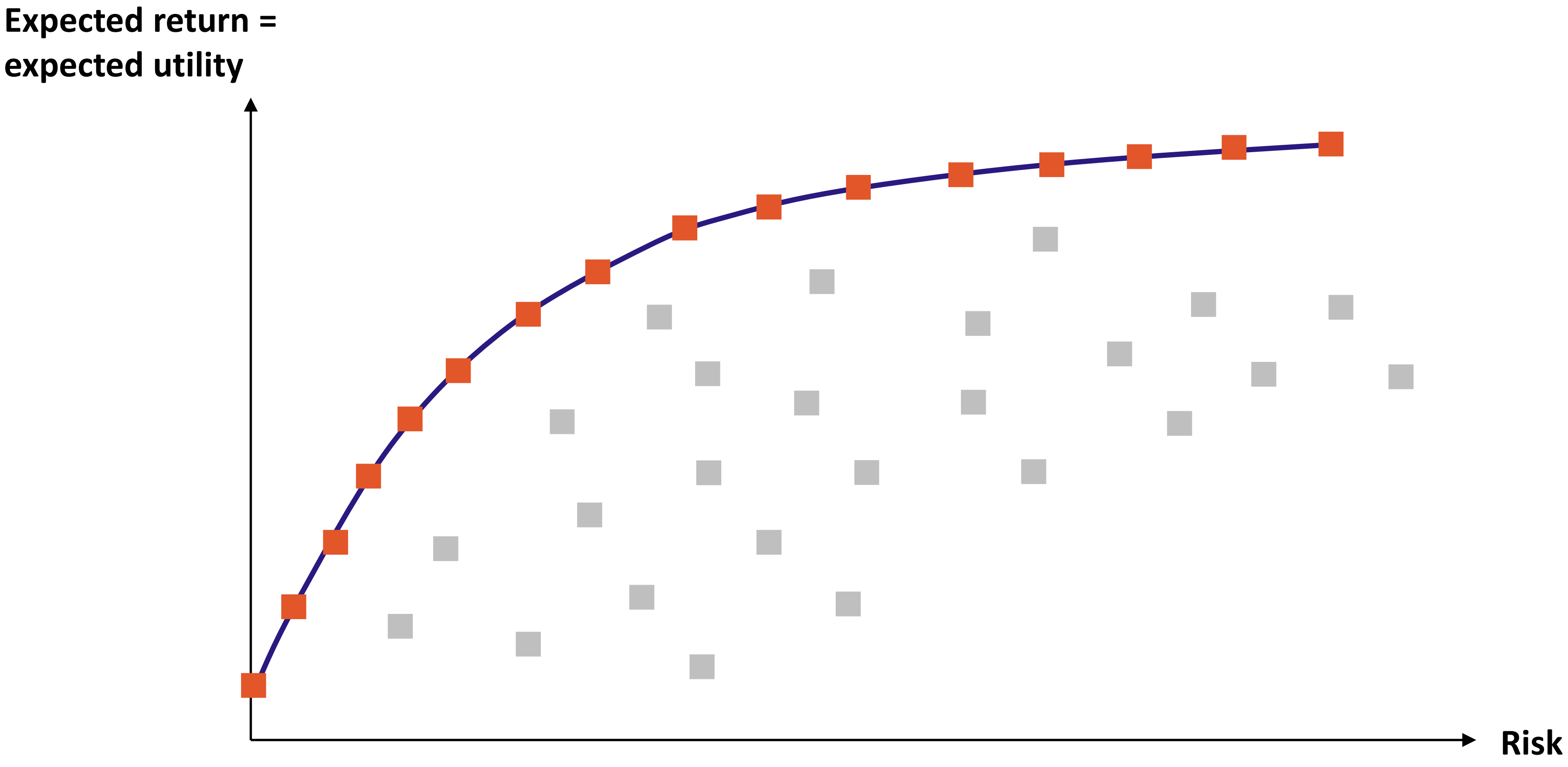
Complementarity slackness :

$\lambda_i x_i = 0$: either $x_i = 0$, or $\lambda_i = 0$.
As $e^T x = 1$, there is at least one value of i for which $x_i > 0$.
Let's i^* be such that $x_{i^*} > 0$, then $\lambda_{i^*} = 0$ so $\mu = m_{i^*}$ and therefore $m_{i^*} \geq m_i, \forall i$, so $i^* = \arg\max_i m_i$. In the sequel, let'assume that $i^* = 1$.

The efficient portfolio – Step 2

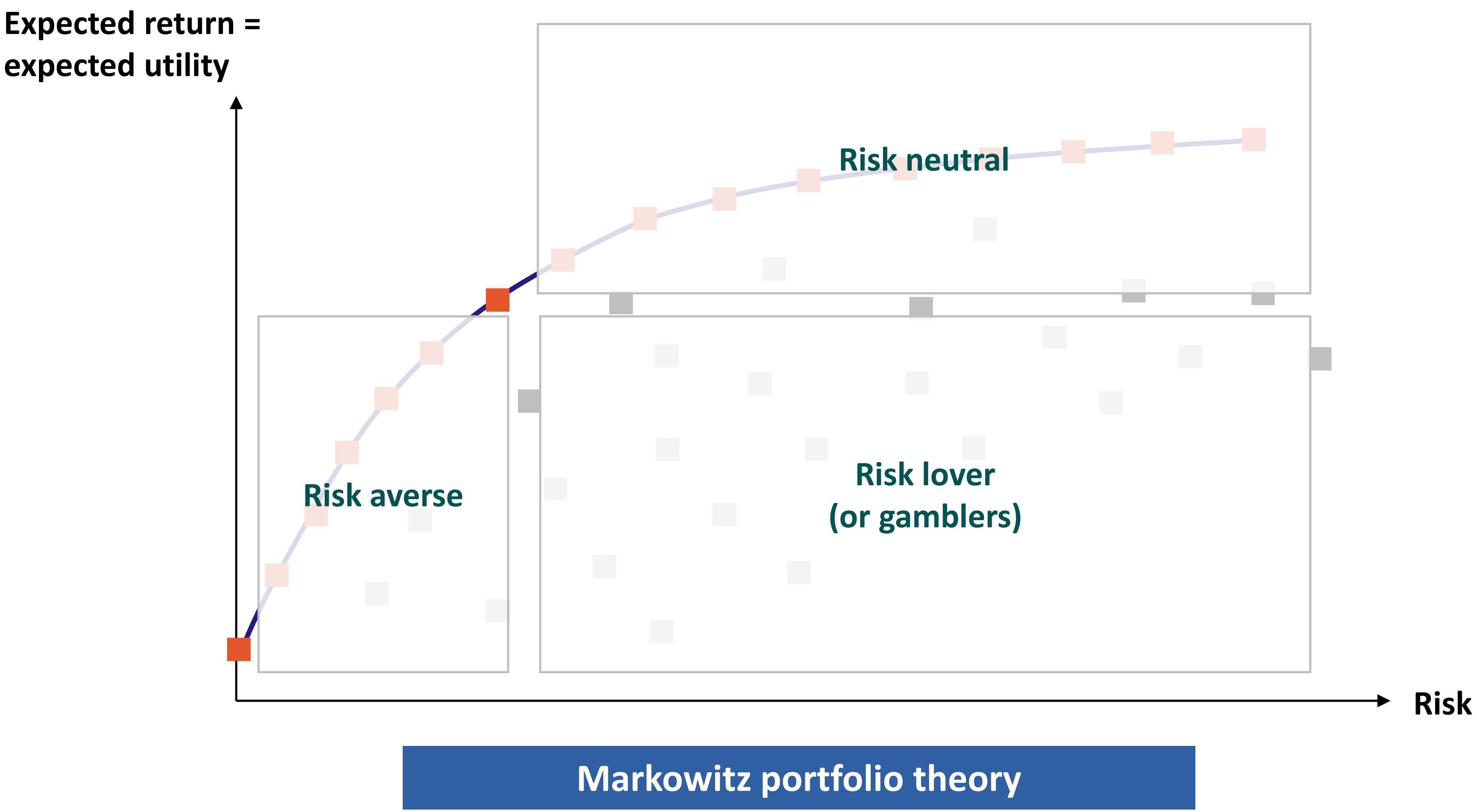
The quadratic problem	Iterate on $k = 1, \cdots, N$: $m_{target} = \frac{k}{N} m_1$: $$(P)\begin{cases} \max\limits_x & x^T C x \\ \text{s.t.} & e^T x = 1, \\ & m^T x = m_{target} \\ & x \in \mathbb{R}^n_+ \end{cases}$$
KKT conditions (w/o inequality constraint)	Solve the linear system : $\begin{bmatrix} C & e & m \\ e^T & 0 & 0 \\ m^T & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \mu_1 \\ \mu_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ m_{target} \end{bmatrix}$ Then check whether $x \geq 0$. If $x_i < 0$, add the constraint $x_i = 0$ and add the corresponding $\lambda \geq 0$ variable in the system.
KKT conditions (with 1 inequality constraint)	Solve the linear system : $\begin{bmatrix} C & e & m & e_{(j)} \\ e^T & 0 & 0 & 0 \\ m^T & 0 & 0 & 0 \\ e_{(j)}^T & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \mu_1 \\ \mu_2 \\ \lambda_j \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ m_{target} \\ 0 \end{bmatrix}$ with $e_{(j)}$ is the null vector except the j-th component that equals to 1. Add non-negativity constraints conjointly if required

Connecting the dots between Markowitz portfolio & expected utility theory

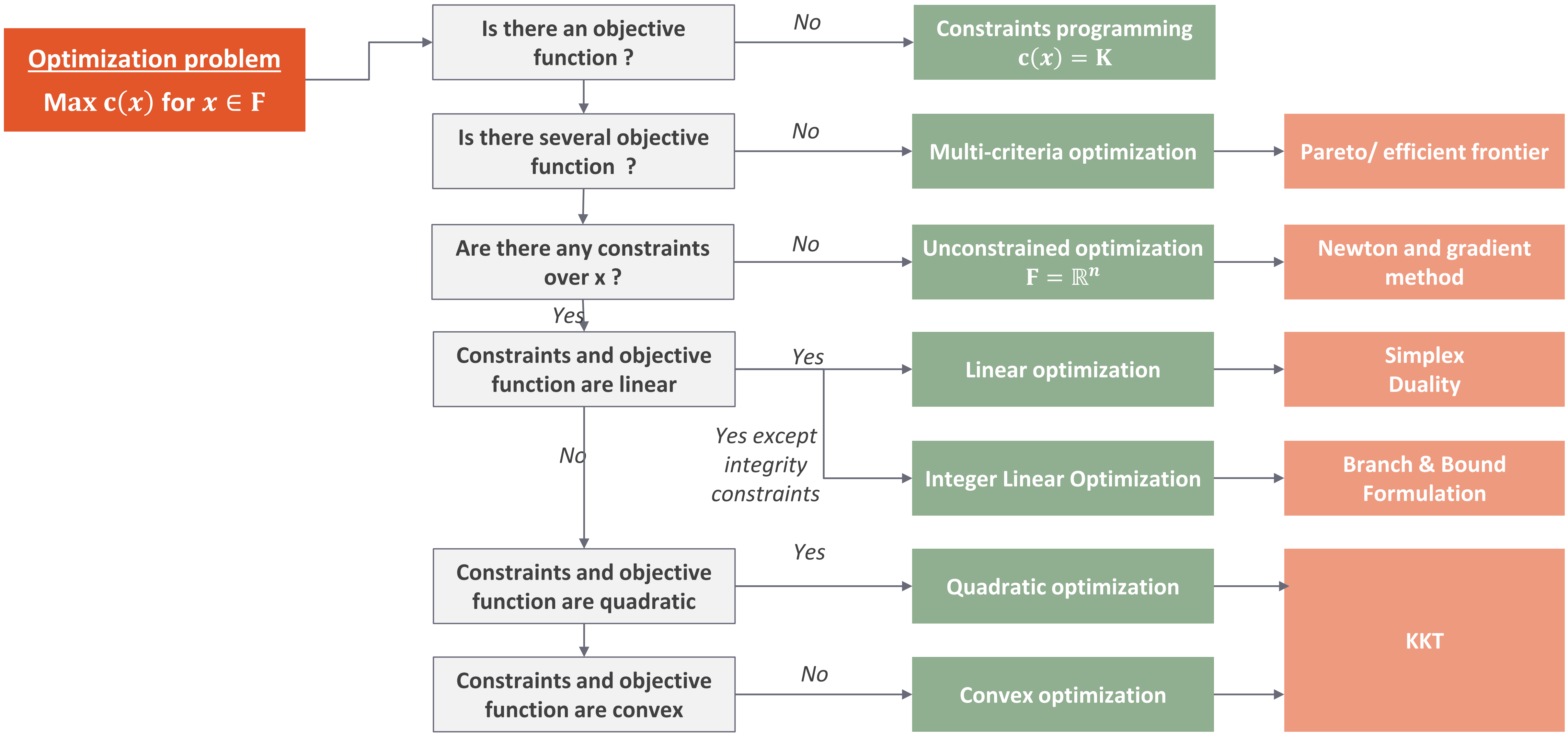


Markowitz portfolio theory

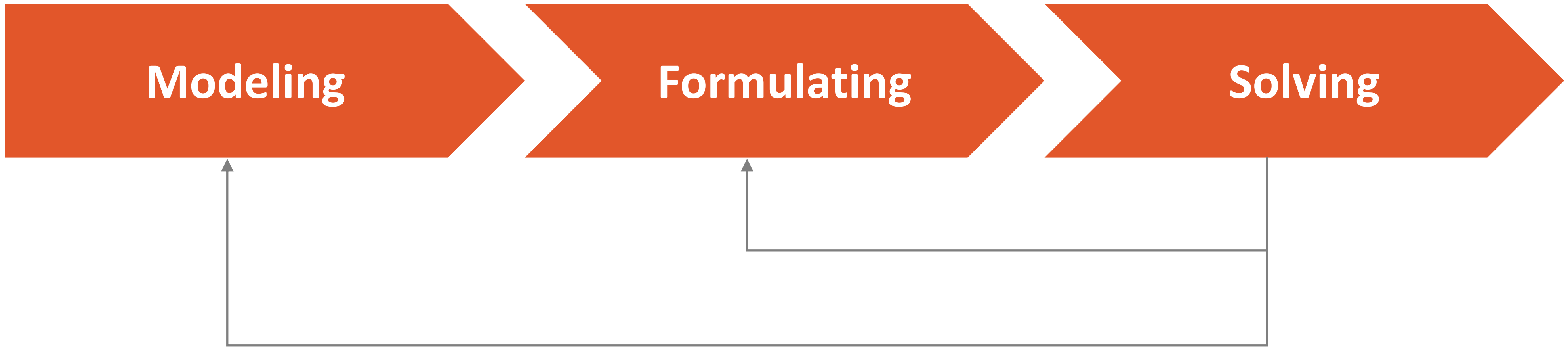
Connecting the dots between Markowitz portfolio & expected utility theory



WRAP-UP on optimization



3 tasks



Conclusion

Static paradigm

- I have a random vector that depend one my decision : $\xi(x)$
- I use risk measures (such as expected value, quantile, CVaR, worst-case…) to get rid of the uncertainty
- I have 3 ways of representing uncertainties :
  - Continuous distribution
  - Uncertainty sets
  - Discrete distribution (scenarios)

Dynamic paradigm

- Some of my decisions are made after the uncertainty is known (recourse variables)
- In order to consider them in an optimization problem, uncertainties has to be represented through scenarios

Optimization under uncertainty brings additional difficulties compared to deterministic optimization

1 Modeling

How to model uncertainty ?

- Uncertainty set : $\varpi \in \Omega$
- Random variables (with probability)
  - Continuous
  - Discrete but countable infinite
  - Discrete and finite (scenarios)
- Fuzzy logic, ….

How to model decision dynamics

Which dependency to upcoming information ?

- Static
- Dynamic :
  - With recourse
  - Multi-stage

2 Solving

Uncertain problems are significantly **more difficult to solve** than their deterministic version :

- **Larger scale** (more variables, more constraints, more inputs data, …)
- Converting the stochastic problem into their deterministic counterpart generally introduces more **complex mathematical structure** (not linear, …)

The different choices to make

How to model uncertainty ?

- Continuous distribution
- Discrete distribution (scenarios)
- Fuzzy logic

Preferences of the decision maker

- Risk-neutral
- Risk-averse
- Risk-lovers

How to intertwine decision with upcoming information ?

- Static
- Dynamic
  - With recourse
  - Multi-stage

How to deal with uncertain constraints

- Penalisation
- Chance constraints
- Worst-case
- Recourse action

In the case « decision – hazard », 2 possible configurations