Challenges for Real Applications

Data Science and Engineering (I)
Master's Degree in Computer Engineering

UNIVERSIDAD DE MÁLAGA

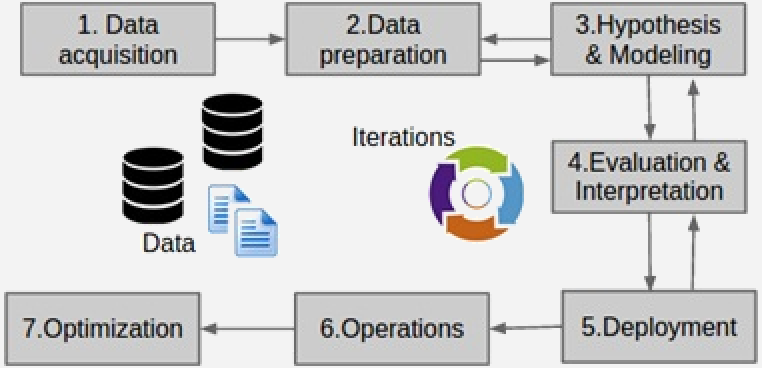
E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

Challenges for real products

1. **Managing** a data science and engineering project
2. Search, **optimization**, and learning
3. **Needs of real projects**: scalability, dynamism, robustness, multiple objectives, restrictions, and self-control
4. **Examples** of real products and services
5. **Modern techniques** for real applications

Management of a DS project (I)

- The typical data science project is an **engineering procedure**: start, steps, end
- Full of **informed decisions** on whether to continue based on pre-defined criteria
- **Goal:** optimize **resource** utilization, get high-quality **results** and maximize **benefits**
- **Money** is an issue, but **realistic** hypotheses and ideas are a must
- **The data science life-cycle**:
  1. Data acquisition
  2. Data preparation
  3. Hypothesis and modeling
  4. Evaluation & Interpretation
  5. Deployment
  6. Operations
  7. Optimization



Management of a DS project (II)

1. **Data acquisition** – acquiring data from **internal** and **external** sources

2. **Data preparation** ("data wrangling") - involves cleaning the data and reshaping it into a readily **usable**

3. **Hypothesis and modeling** – **applying ML techniques** to all data (MS: model selection). MS involves to identify **training/test** sets

4. **Evaluation and interpretation** – comparing model performances

- *Steps 2-3-4 are repeated; as the understanding of data and business becomes clearer*

Management of a DS project (III)

5. **Deployment** – the project is run in a production environment. It could include fast-tweaks *after* deployment, based on the **continuous deployment** model.

6. **Operations** (maintenance) – This phase could also follow a **DevOps** model which gels well with the continuous deployment model, given the rapid time-to-market requirements in big data projects. Steps 5 and 6 are mixed usually (like agile software in software engineering).

7. **Optimization** – This could be triggered by failing performance, or due to the need to add new data sources and retraining the model, ...

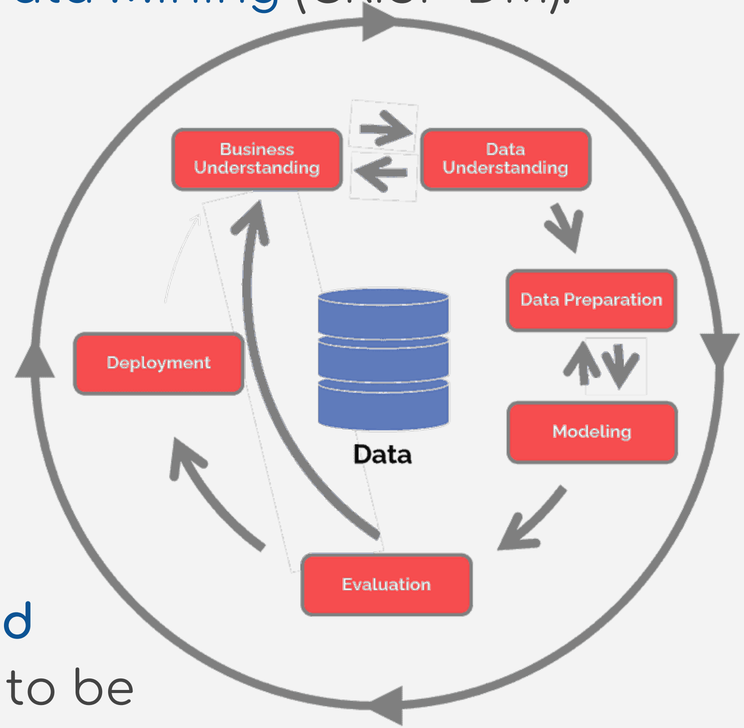
Examples

In which step is the activity done? (1. Data acquisition, 2. Data preparation, 3. Hypothesis and modeling, 4. Evaluation & Interpretation, 5. Deployment, 6. Operations, and 7. Optimization)

A.- Removing outliers
B.- Calculating vehicle speed from its positions
C.- Running the complete system on a Docker infrastructure for testing
D.- Splitting the dataset into train and test set
E.- Deploy improved version of the model
F.- Rebooting the complete system after an unrecoverable failure
G.- Applying cross-validation
H.- Tuning the model parameters
I.- Examining which models can be applied to the data
J.- Obtaining values from car’s sensors (OBD-II)

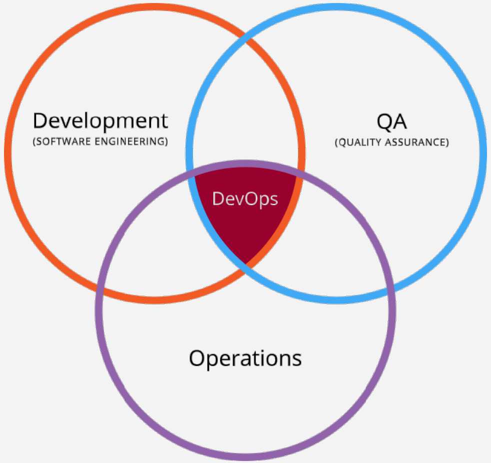
You should now also on (I) CRISP-DM

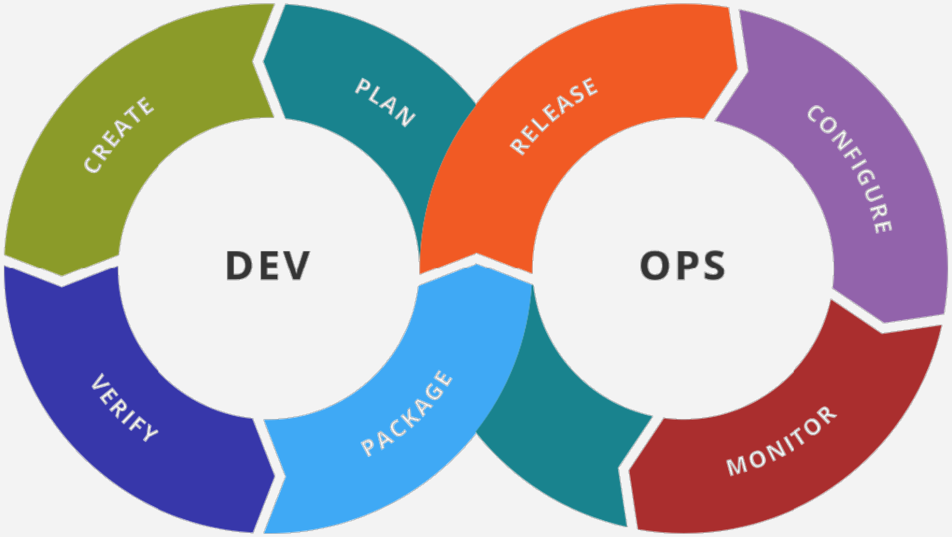
- CRoss Industry Standard Process for Data Mining (CRISP-DM):
  1. Business understanding
  2. Data understanding
  3. Data preparation
  4. Modeling
  5. Evaluation
  6. Deployment
- Library of assets (expertise/maturity):
  1. Library of business use case
  2. Data requirements
  3. Minimum data quality requirements
  4. ...
- Data scientists are likely to have limited business domain expertise. They need to be paired with business people and those with expertise in understanding the data.



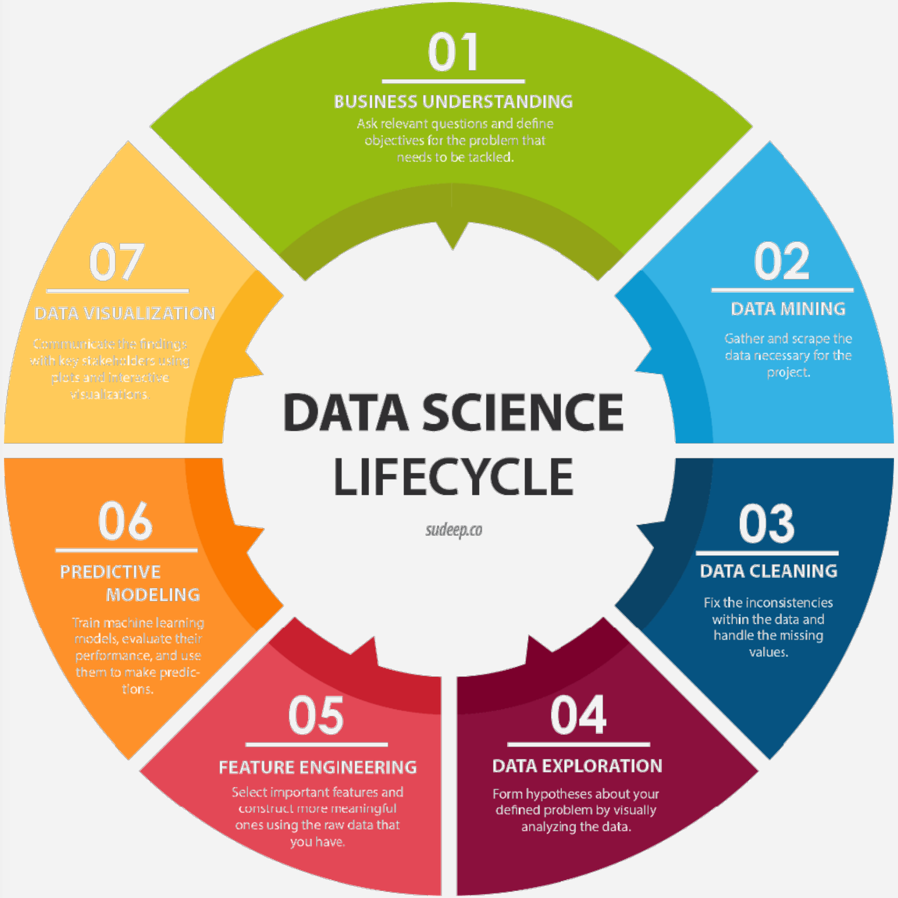
You should now also on (II) DevOps

- Set of practices to **reduce the time** between committing a change to a system and the change being placed into normal production, while ensuring **quality**
- It uses different sets of tools (toolchains) rather than a single one
- **Steps** – Coding + Building + Testing + Packaging + Releasing + Configuring + Monitoring

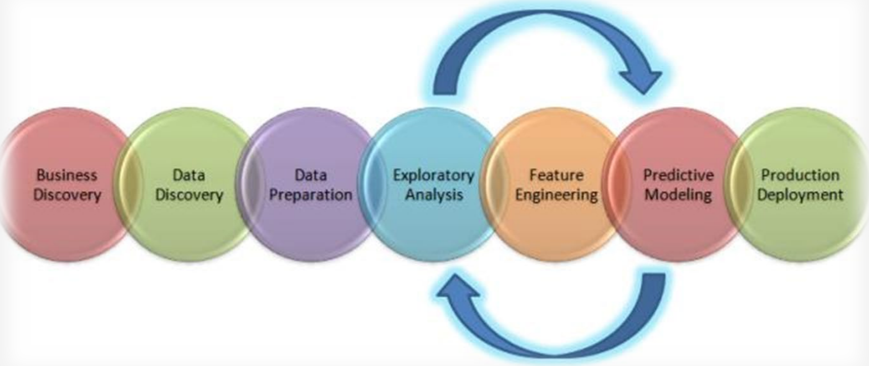




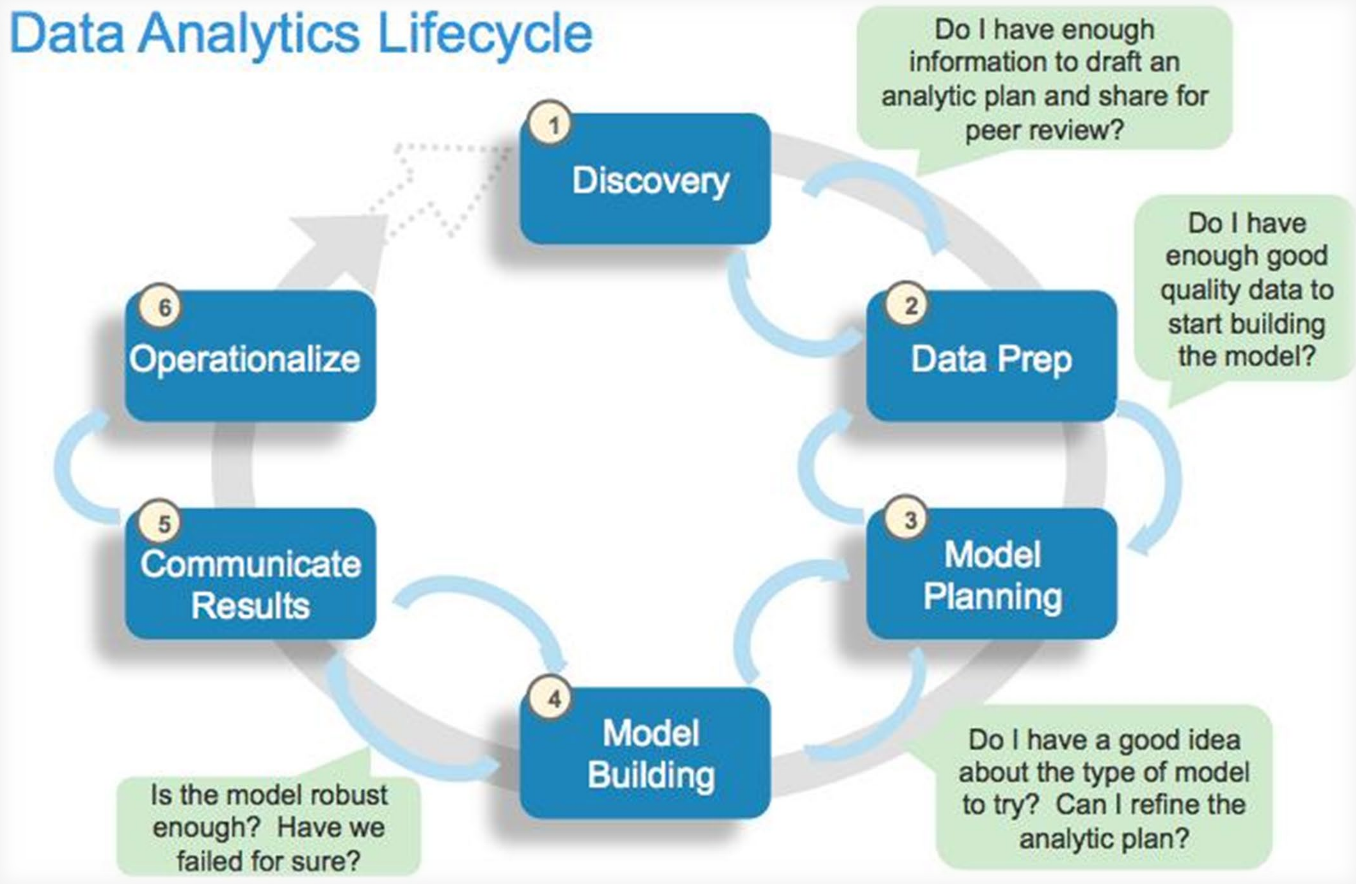
Other models having seven steps



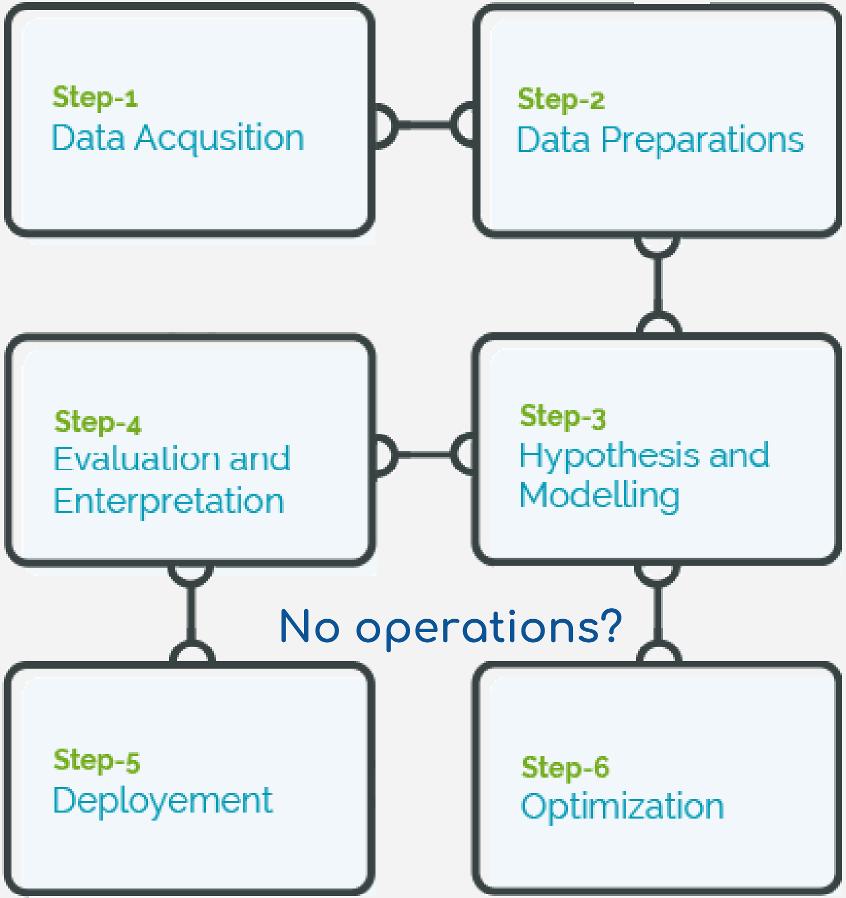
Too new!
Standards just born…



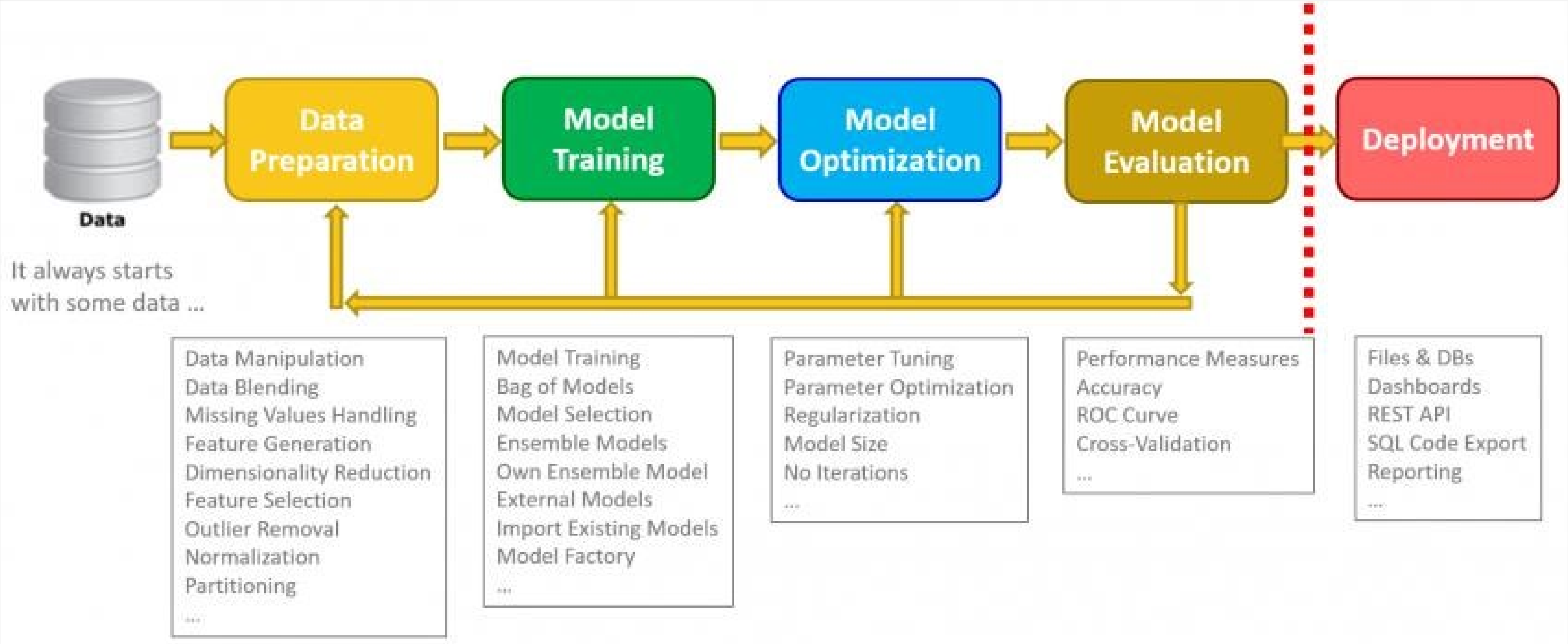
Other models having six steps (I)



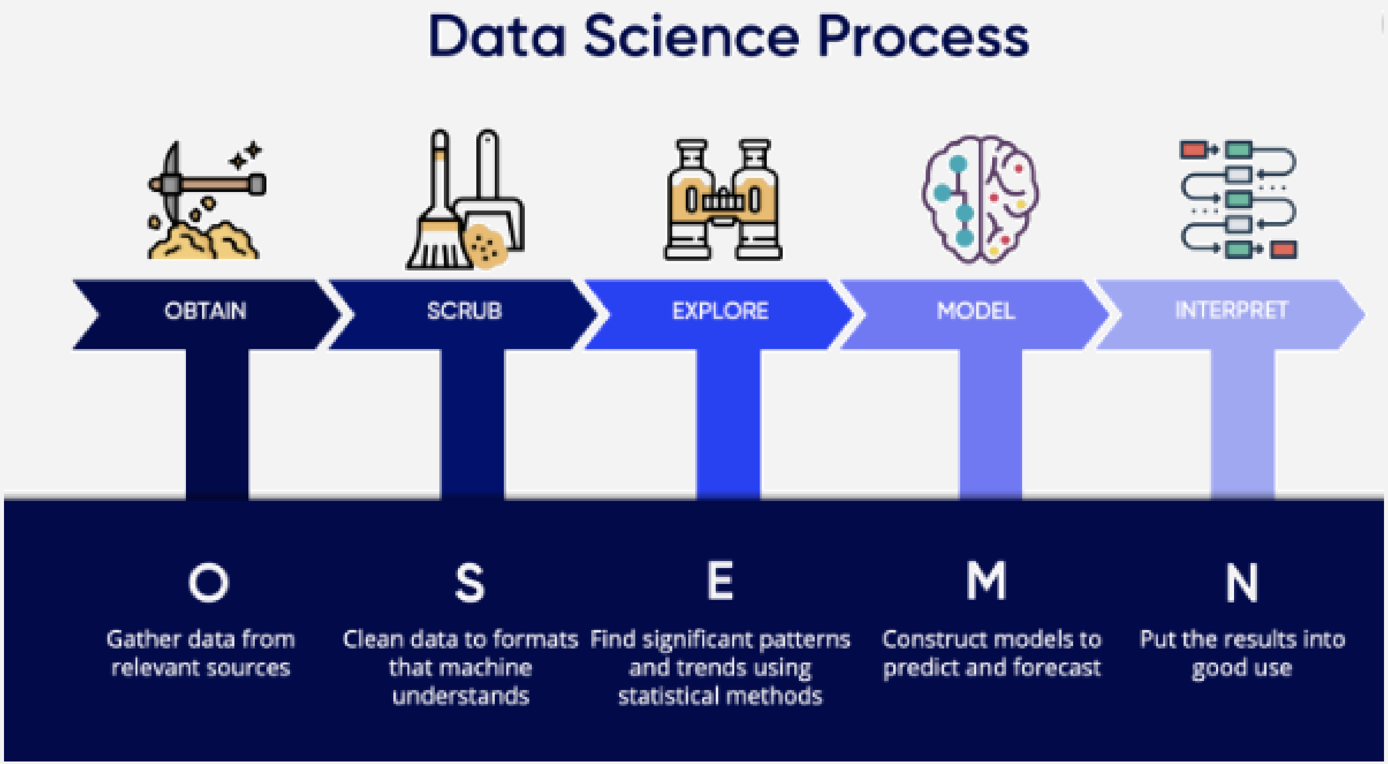
Other models having six steps (II)



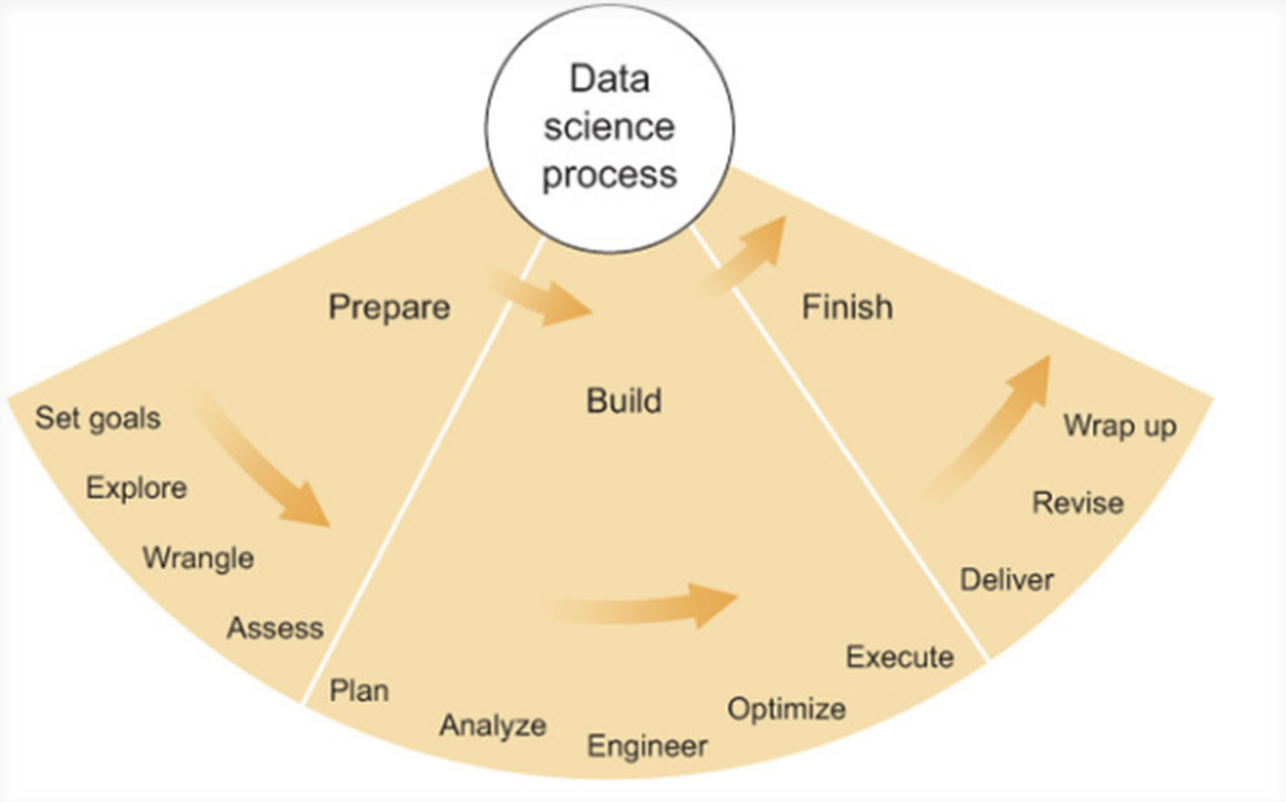
Other models a smaller number of steps



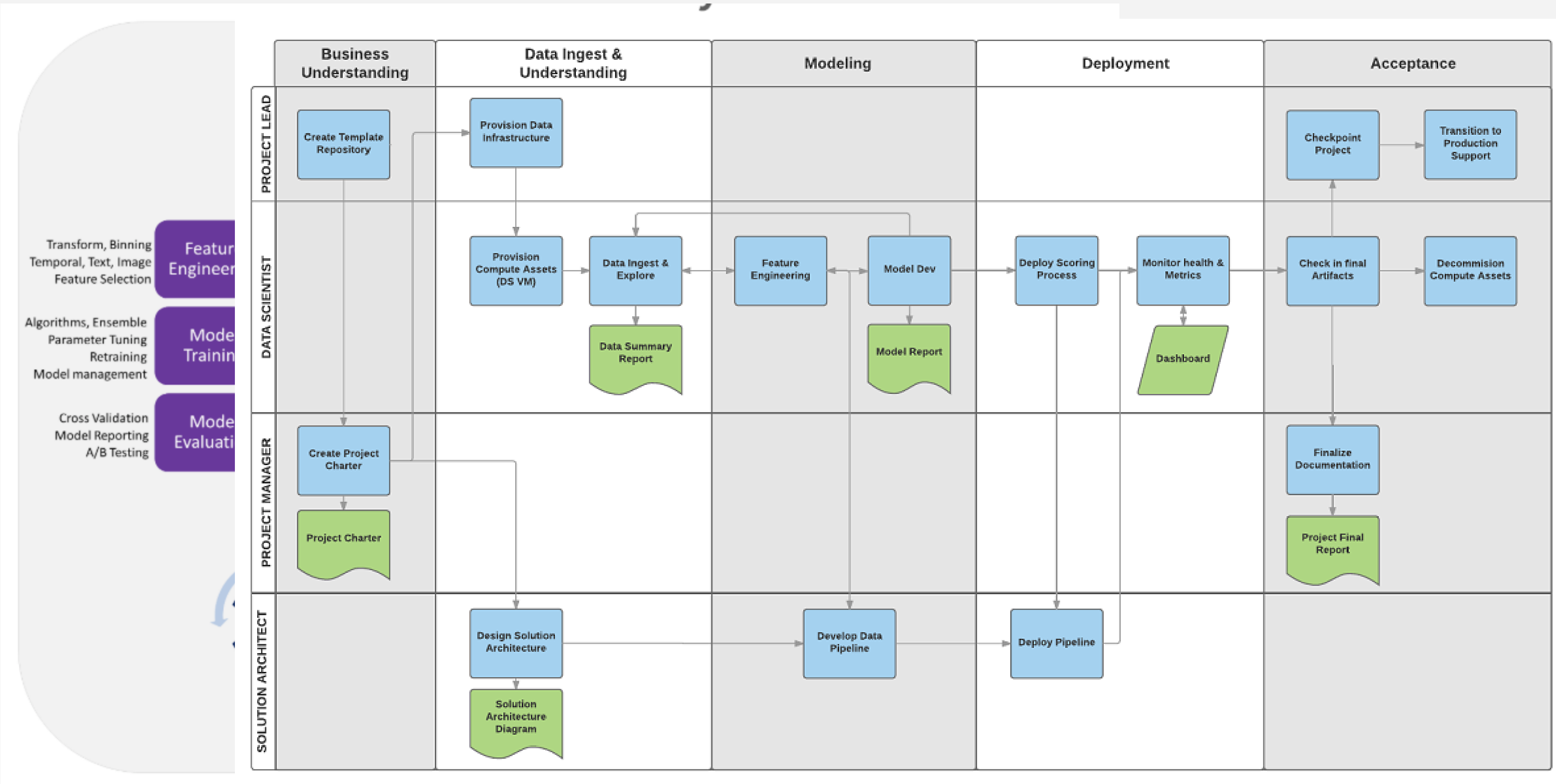
Other models a smaller number of steps



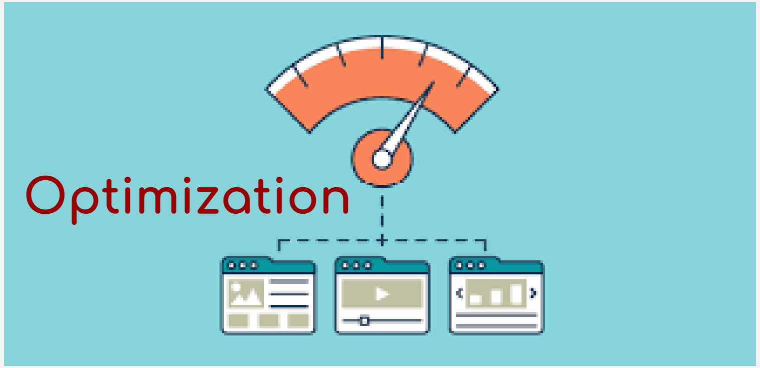
Models having a larger number of steps!

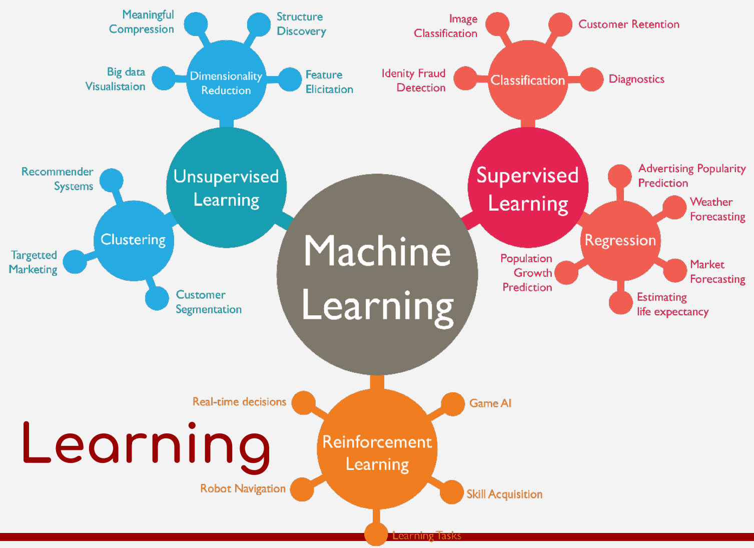


... OK; let's go wild!



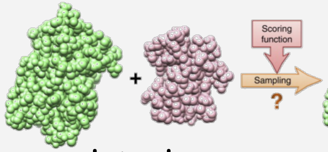
Search, Optimization, and Learning

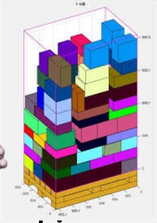


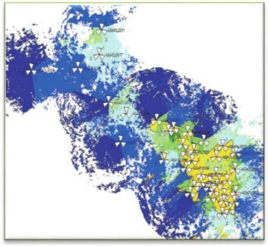


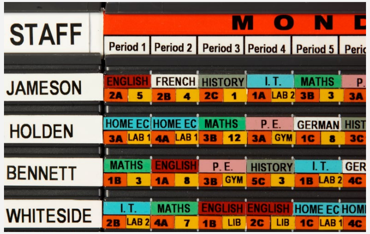
Optimization problems are everywhere!

- Logistics, transportation, supply change management
- Manufacturing, production lines
- Timetabling
- Cutting & packing
- Computer networks and telecommunications
- Health
- Videogames
- Software (SBSE)

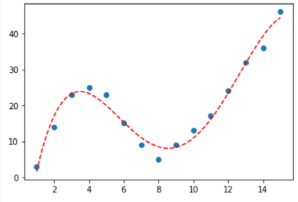


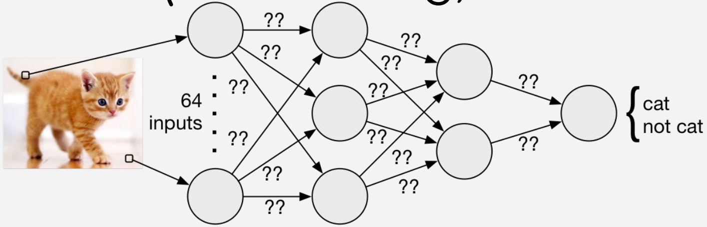


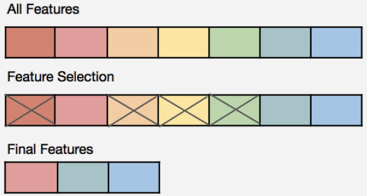




… even in **data science** (data fitting, NN training, feature selection…)







Optimization problems

- Most general form:

$$min_{x \in X_{ad}} f(x)$$

Terminology:

- $f: X_{ad} \to R$: fitness function, objective function, usually real-valued
- $min \leftrightarrow max$ by replacement $f \leftrightarrow -f$
- x: control or optimization parameters
  - integer/discrete, continuous, or mixed-integer problems
- X: usually vector space or unbound set
- $X_{ad} \subset X$: admissible or feasible set
  - $X_{ad} = X$: unconstrained problem
  - $X_{ad} \neq X$: constrained problem

A taxonomy of modern AI techniques

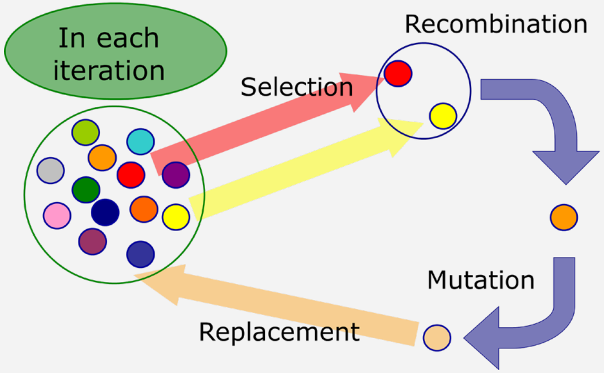
- Optimization Algorithms
  - Exacts
    - Calculus
      - Direct
        - Newton
        - Greedy
      - Indirect
    - Enumerative
      - DP
      - B&B
  - *Ad-hoc* Heuristics
  - Metaheuristic
    - Trajectory
      - SA
      - VNS
      - TS
    - Population
      - EA
      - ACO
      - PSO

nature inspired in red

Evolutionary Algorithm

- Based on the ideas of Darwinian Evolution theory

```
t := 0
initialize(P(t))
evaluate(P(t))
while  not end condition do
       P'(t) := selection(P(t))
       P'(t) := recombination(P'(t))
       P'(t) := mutation(P'(t))
       evaluate(P'(t))
       P(t+1) := replacement(P(t), P'(t))
       t := t+1
end while
```

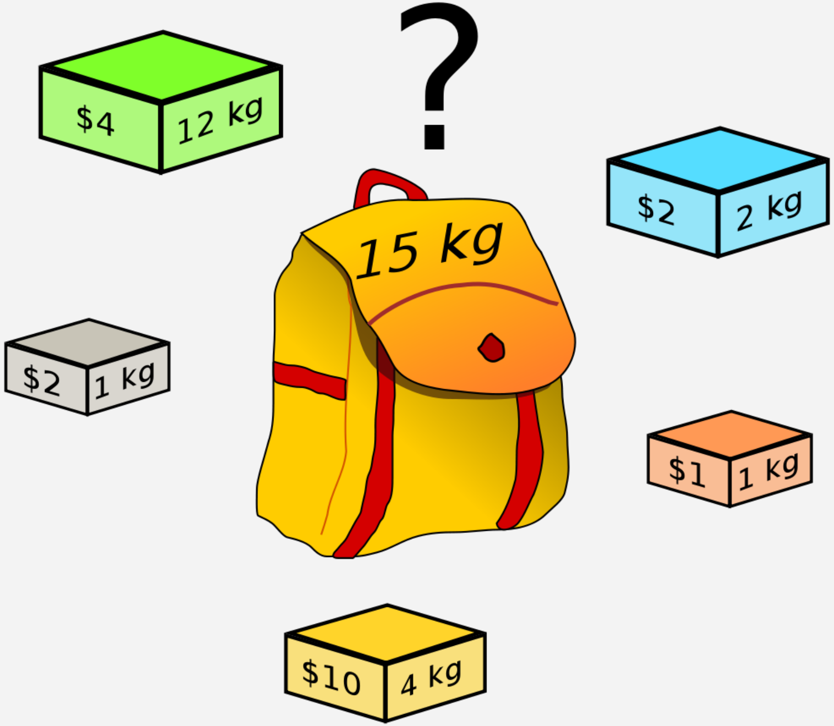


In order to use one EA several steps of instantiation are needed:

- Problem: genotype (encoding) and fitness function
- Operators and their parameters
- Stopping criterion

Evolutionary Algorithm: example (I)

- 0-1 Knapsack problem



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

Subject to $\sum_{i=1}^{n} w_i x_i \leq W$ and $x_i \in \{0,1\}$

Evolutionary Algorithm: example (II)

- **Genotype:** bit string

0	1	1	1	0	0	0	1	1	0	1	0	1	1	1

- **Fitness function:** $$\sum_{i=1}^{n} v_i x_i$$
- **Stop condition:** 100000 evaluations
- **Population:**
  - Size: 100
  - Random generated

0	0	1	1	1	0	0	0	1	1	0	1	0	1	1	1
1	1	1	0	0	1	1	1	0	1	0	1	0	1	1	0
2	1	0	1	1	0	1	0	1	0	1	0	1	0	1	1
	...														
99	0	0	0	1	1	0	1	1	1	0	1	1	1	0	0

- **Selection:** Random
- **Replacement:** Worst

Evolutionary Algorithm: example (III)

- **Crossover / recombination:** SPX (simple point crossover)

0	0	1	1	0	0	1	1	1	1	0	1	1	0	1
0	0	0	1	1	1	0	1	0	0	0	1	0	1	1

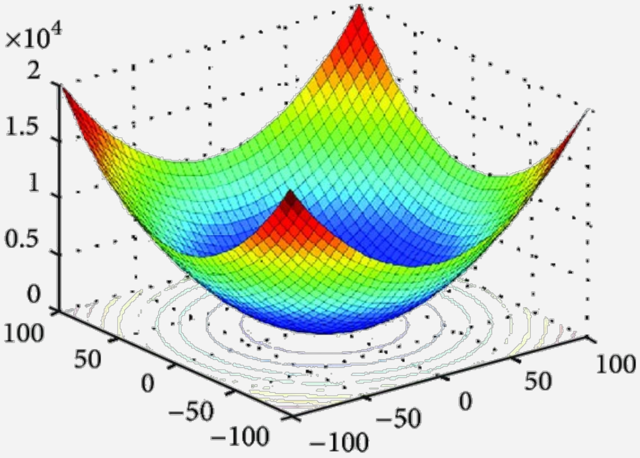
0	0	1	1	0	0	1	1	1	0	0	1	0	1	1
0	0	0	1	1	1	0	1	0	1	0	1	1	0	1

- **Mutation:** bit-flip

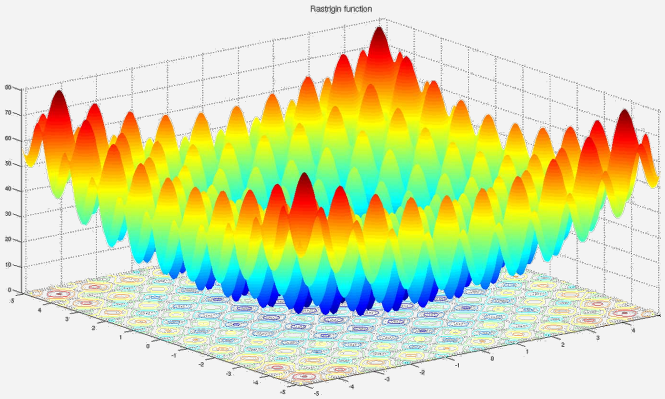
0	0	1	1	0	0	1	1	1	1	0	1	1	0	1

0	0	0	1	0	0	1	1	1	1	1	1	1	0	1

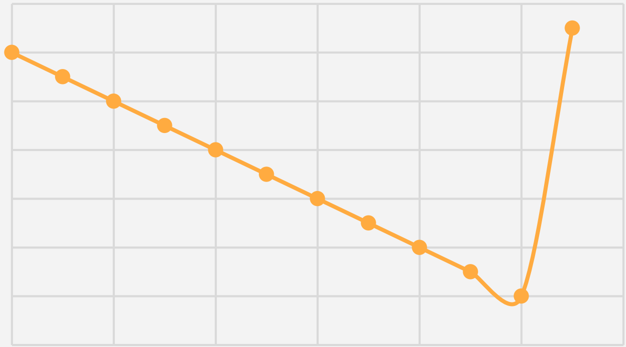
Search space



Unimodal



Multimodal



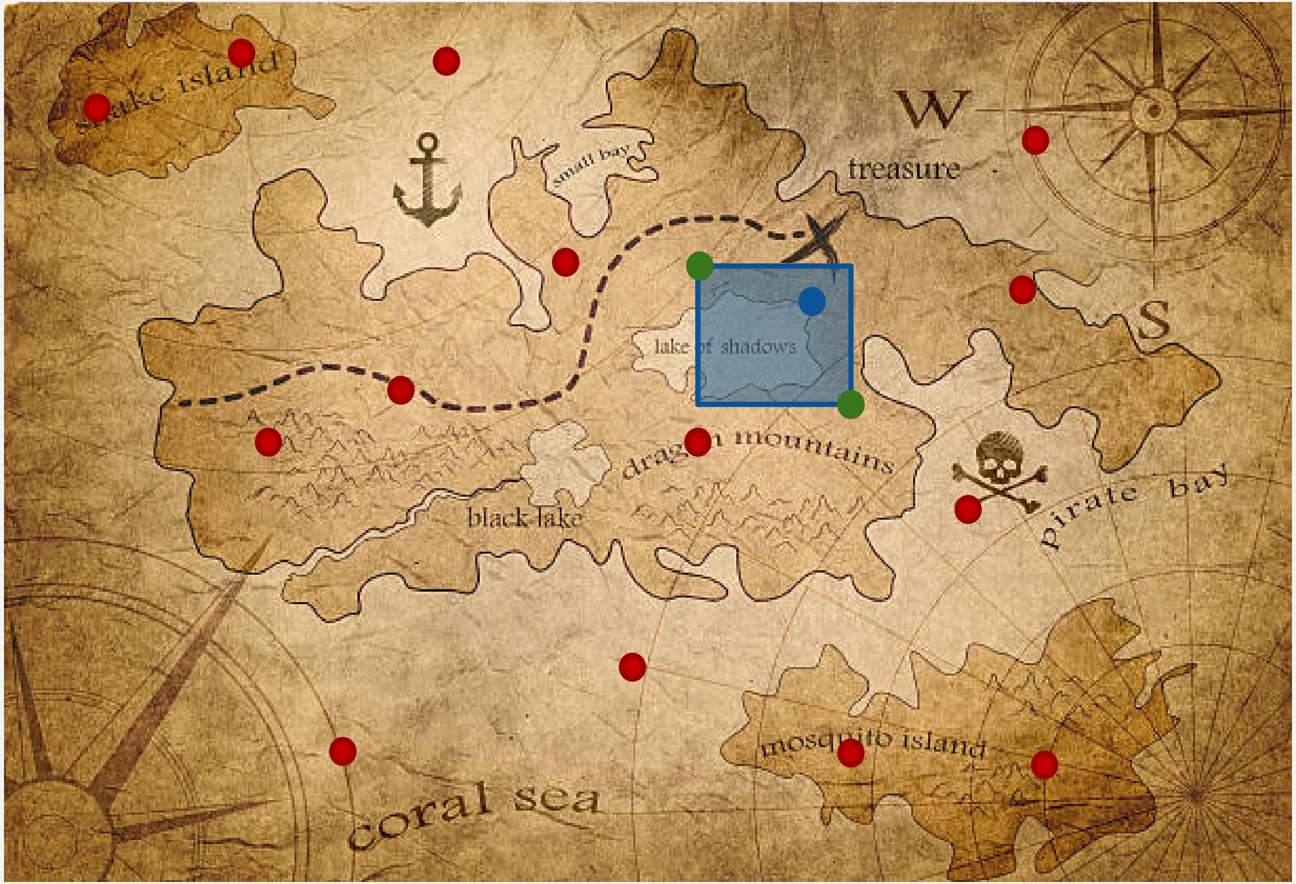
Deceptive

Search space and GA operators

Population generation

Selection

Crossover



Search space and GA operators

Population generation

Selection

Crossover

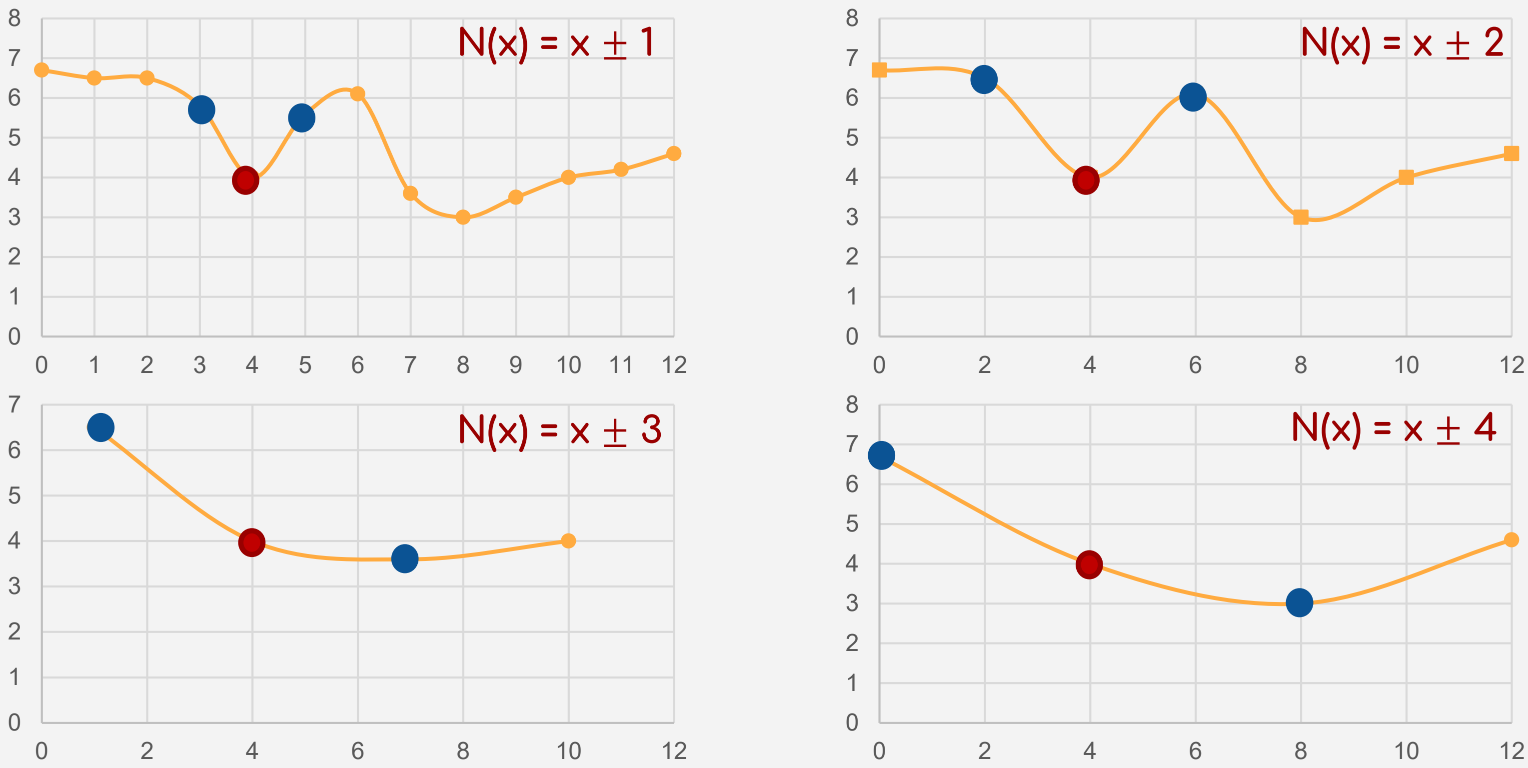
Mutation

Replacement

Search space and Neighbourhood

Current solution

Neighbours



Particle Swarm Optimization (I)

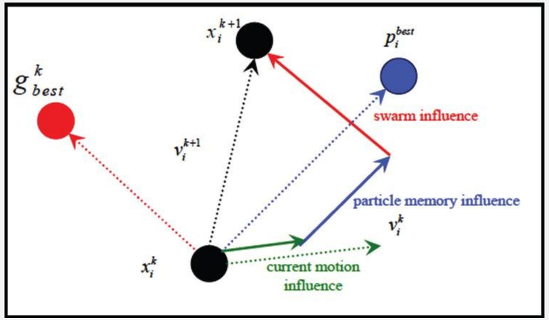
- Particle Swarm Optimization (PSO) is a population based metaheuristic inspired in the social behavior of **birds** within a flock
- It was initially designed for **continuous optimization** problems, but can be used in discrete ones also
- In PSO, each potential solution is called a **particle** and the population of particles is called a swarm
- In this algorithm, each **particle position** p_i is updated each generation k by means of this equation (v^i is its **velocity**):

$$p_i^{k+1} \leftarrow p_i^k + v_i^{k+1}$$

Particle Swarm Optimization (II)

- The velocity of the particle is given by the expression:

$$v_i^{k+1} \leftarrow w \cdot v_i^k + c_1 \cdot r \cdot g_{best}^k + c_2 \cdot r \cdot p_i^{best}$$



Algorithm 1 Pseudocode of PSO

```
initializeSwarm()
locateLeader(b)
while !stopCondition() or g < maxGenerations do
    for each particle x_g^i do
        updateVelocity(v_g^i) // Equation 2
        updatePosition(p_g^i) // Equation 1
        evaluate(p_g^i)
        update(bp_g^i)
    end for
    updateLeader(b_g)
end while
```

Simulated Annealing

- It is based on **annealing** in metallurgy
- SA is a **hill-climbing** method
- It **accepts worse solutions** to avoid getting stuck in local optima, according a criterion (y is new solution and x the old one):

$$rand(0,1) \leq \min\left(1, e^{\frac{f(x)-f(y)}{T}}\right)$$

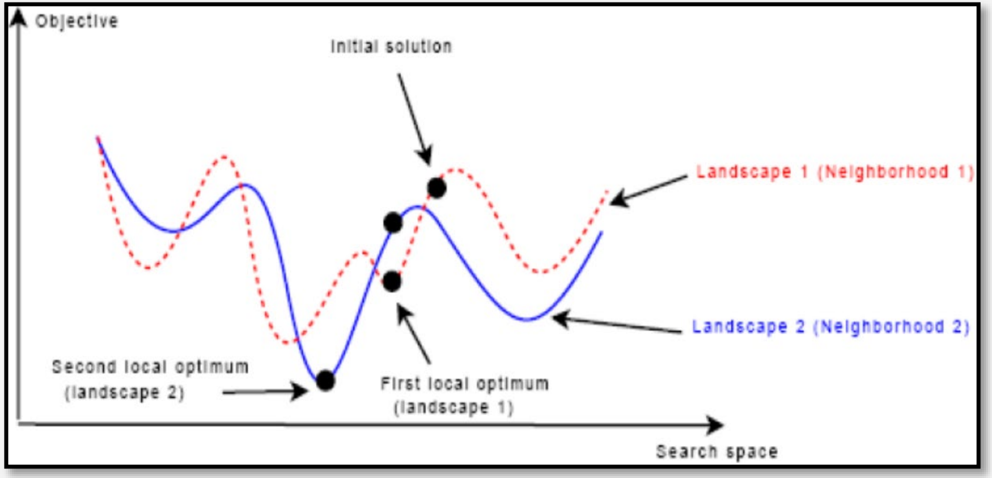
Algorithm 1 Simulated annealing algorithm

procedure SA(f, N, Ω, x^0, T_0)
  $k \leftarrow 0$
  $x_{\min} \leftarrow x^k$
  $f_{\min} \leftarrow f(x_{\min})$
  $T_k \leftarrow T_0$
  while stopping criterion is not satisfied **do**
    $z^k \leftarrow rand\left(N(x^k, T_k)\right)$
    $y^k \leftarrow x^k + z^k$
    if $rand(0,1) \leq \min\{1, \exp\{(f(x^k) - f(y^k))/T_k\}\}$ **then**
      $x^{k+1} \leftarrow y^k$
    else
      $x^{k+1} \leftarrow x^k$
    if $f(x^{k+1}) < f_{\min}$ **then**
      $x_{\min} \leftarrow x^{k+1}$
      $f_{\min} \leftarrow f(x_{\min})$
    $k \leftarrow k+1$
    $T_{k+1} \leftarrow$ temperature is updated
  return $x_{\min}$

Variable Neighbourhood Search

- VNS is a stochastic algorithm with a **set of neighbourhood structures** are defined,
- Each iteration: **shaking**, **local search** and **move**
- VNS **explores** a set of **neighbourhoods** to get different local optima and **escape from local optima**

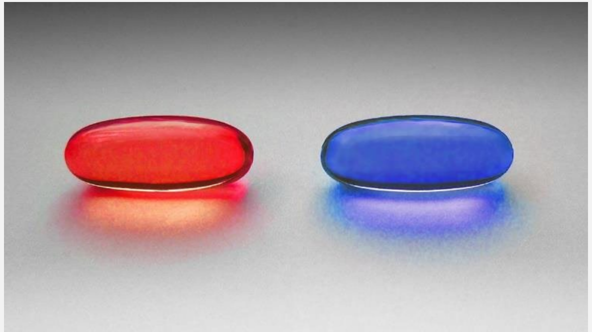
Procedure Algorithm of variable neighborhood search:
begin
  find the best solution found x;
  $k := 1$
  while $(k \leq k_{max})$ **do**
    randomly generate a new solution $y \in N_k(x)$;
    if $(f(x) > f(y))$ **then**
      $x := y$;
      $k := 1$;
    else
      $k := k + 1$;
    endif
  endwhile
end



Real problems pose real challenges

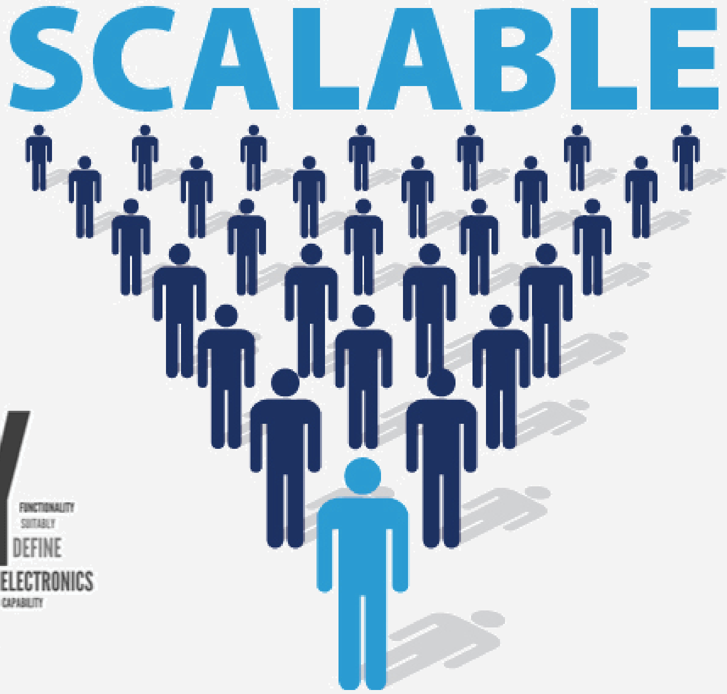
Reality is challenging:

- **Large scale**, every is really big
- **Time** consuming and real time
- **Dynamic**, everything changes in time
- **Uncertainty** in all tasks and phases
- **Complex** relations, interdependences
- Several **goals** at the same time
- **Human** preferences and interfaces
- Lots of **restrictions** (legal, technical...)
- **Mobile** plus **desktop** applications



Scalability

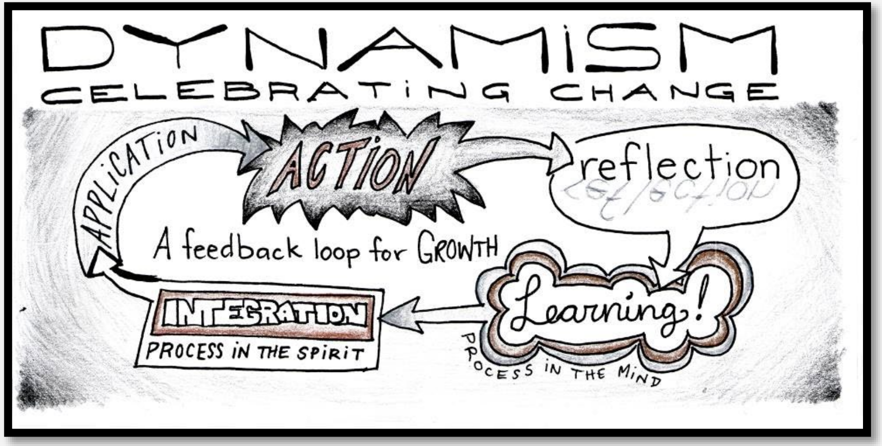
Scalability is the property of a system to handle a growing amount of work by adding resources to the system



Dynamism and Uncertainty

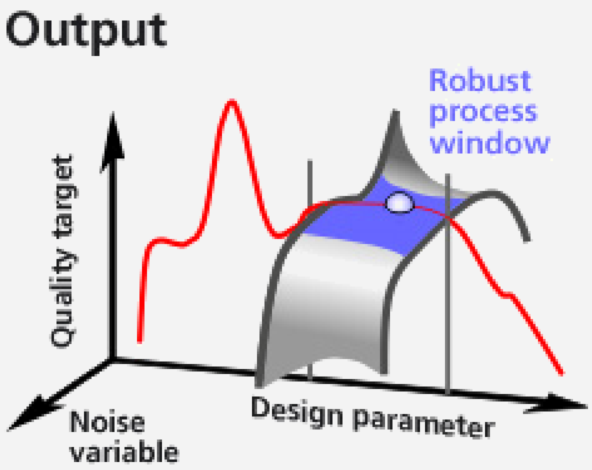
Dynamism: the problem conditions change over the time in an unpredictable way

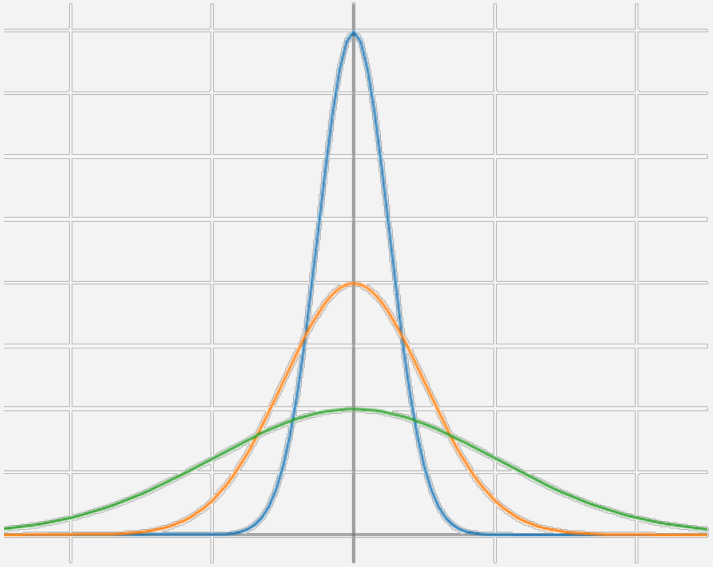
Uncertainty: the problem involves imperfect or unkown information



Robustness

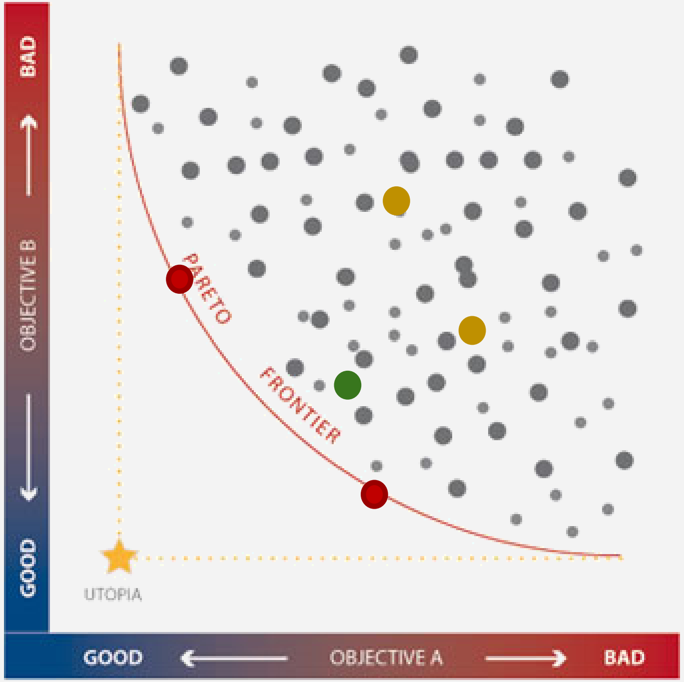
Robustness: the performance is stable after adding some noise to the environment

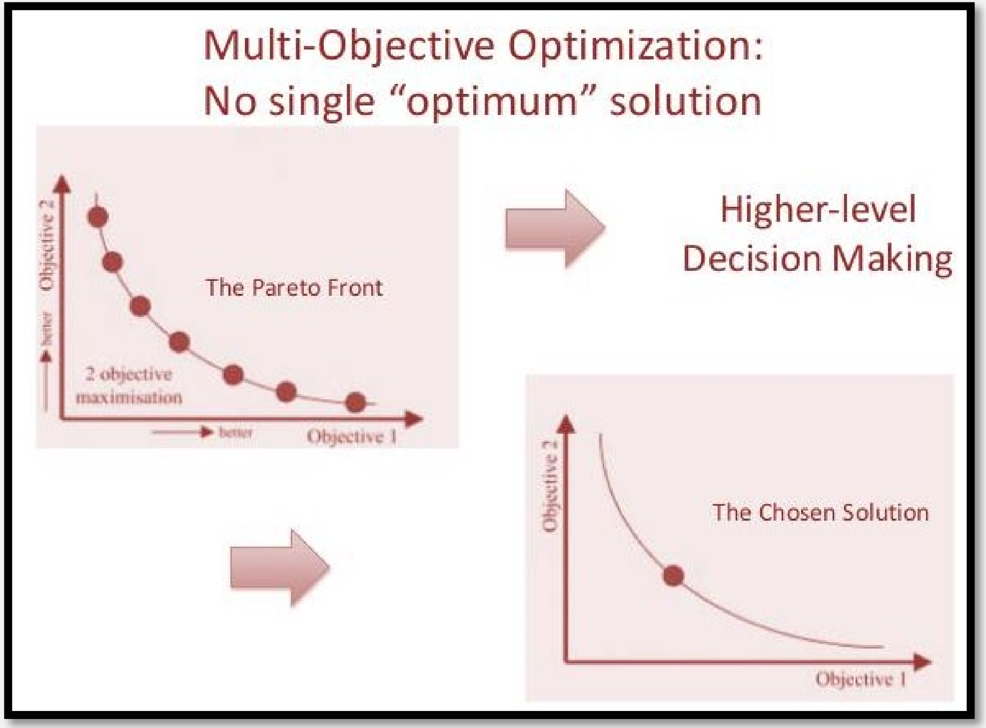




Multiple Objectives

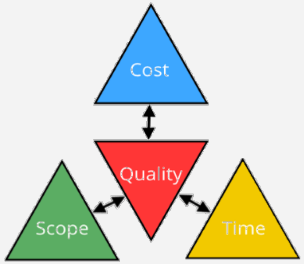
Green dominates yellow. Red are non-dominated.



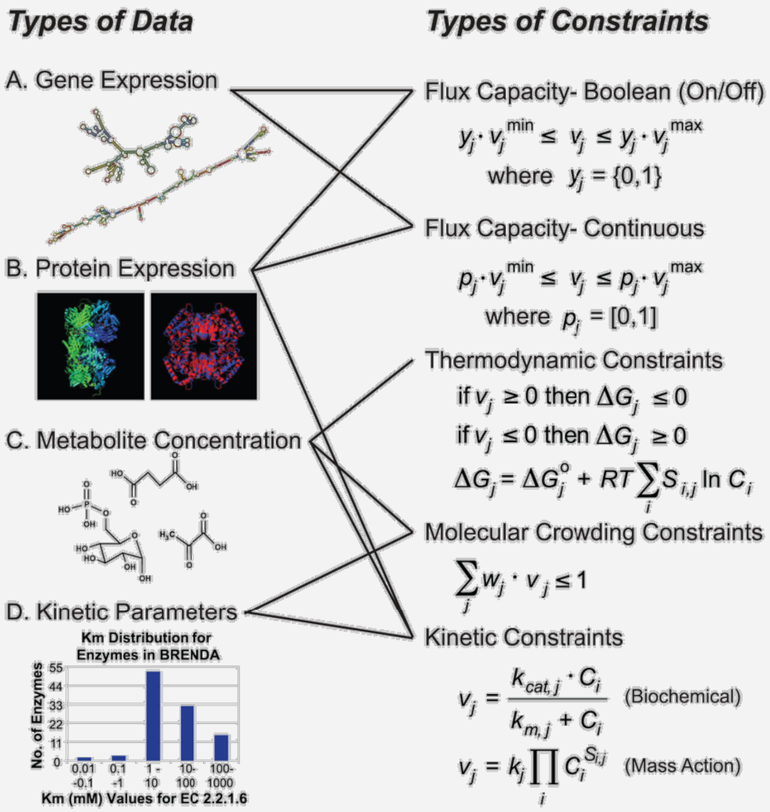


Constraints

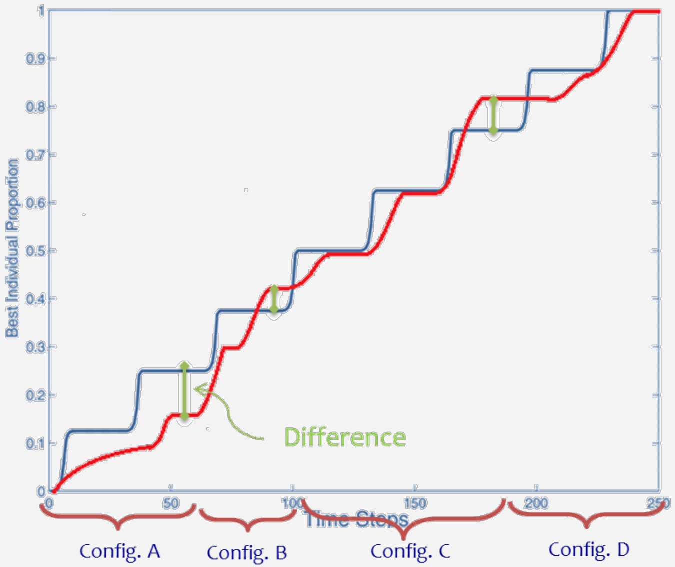
… many types of constrains …



Project Constraints



AutoML



Best Individual Proportion
Time Steps
Difference
Config. A
Config. B
Config. C
Config. D

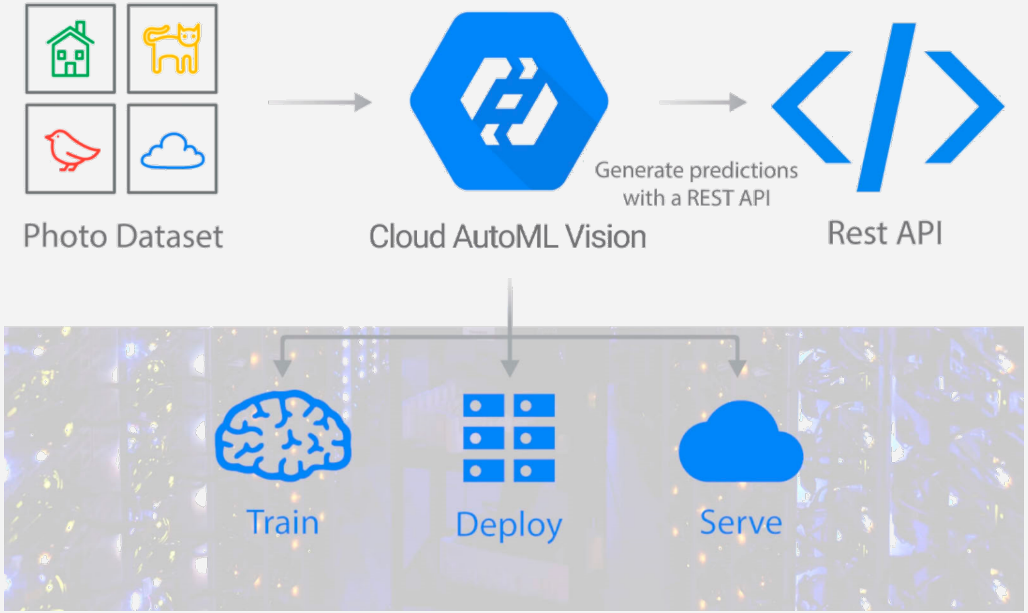


Photo Dataset
Cloud AutoML Vision
Generate predictions with a REST API
Rest API
Train
Deploy
Serve

An example (I)

Problem:

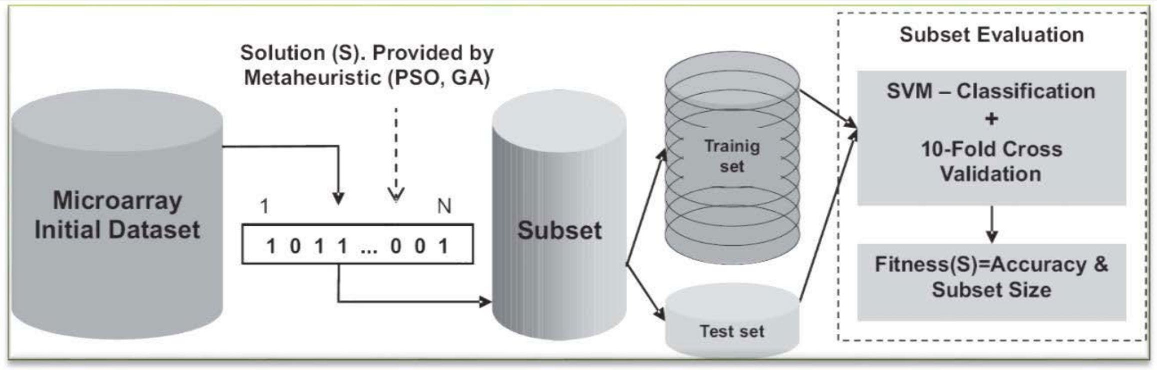
- **Gene selection** and cancer classification of DNA
- Microarray, feature selection

Objectives:

- Maximize accuracy of prediction
- Minimize the number of selected genes
- Maximize sensibility and specificity (ROC factors)

Phases:

- Feature selection
- Training
- Validation
- Fitness calculation



An example (II)

Fitness:

- **Monobjective:** aggregative: (alpha*100/accuracy + beta * #features)
- **Multiobjective:**
  2 (accuracy, #features) or 3 (sensibility, specificity, #features)

Classification:

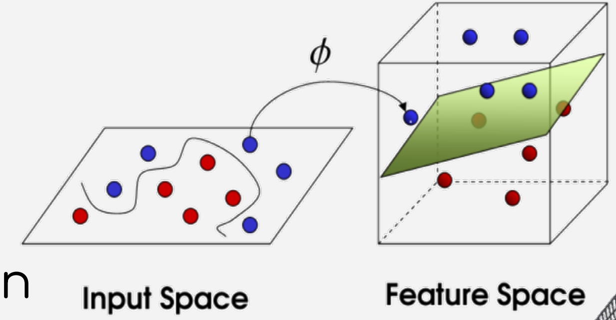
- SVM
- K-means

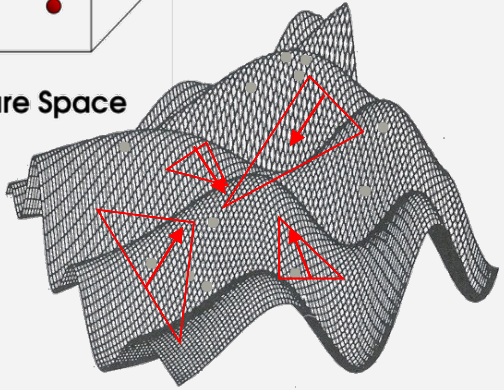
Validation:

- Leave one out cross-validation
- 10-fold cross-validation

Algorithms (for feature selection):

- PSO variant
- GA variant





An example (III)

Instances:

- Large scale datasets of well-known cancer DNA Microarrays: Leukemia, Colon, Prostate, Lung, Ovarian, Breast (e.g. breast 24481 genes and 97 patient samples)

Results: comparison against other techniques (S.O.T.A.)

Dataset	GPSO	GA	Huerta et al.	Juliusdottir et al.	Deb et al.	Guyon et al.	Yu et al.	Liu et al.	Shen et al.
Leukemia	97.38(3)	97.27(4)	100(25)	-	100(4)	**100(2)**	87.44(4)	-	-
Breast	86.35(4)	**95.86(4)**	-	-	-	-	79.38(67)	-	-
Colon	**100(2)**	100(3)	99.41(10)	94.12(37)	97(7)	98(4)	93.55(4)	85.48(-)	94(4)
Lung	99.00(4)	**99.49(4)**	-	-	-	-	98.34(6)	-	-
Ovarian	**99.44(4)**	98.83(4)	-	-	-	-	-	99.21(75)	-
Prostate	**98.66(4)**	98.65(4)	-	88.88(20)	-	-	-	-	-

Leukemia Gene Subset:

PSO: K01383, U03056, J04130 vs **GA**: L40379, S85963, U83192, Z49099

Advanced Tools

Data Science and Engineering (I)
Master's Degree in Computer Engineering

UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

Advanced techniques and technologies

1. **Complex** problems need advanced tools
2. **Measuring** efficacy and efficiency
3. **Parallel hardware**, or how new technology helps
4. **Algorithm hybridization**, or how new techniques can help
5. **Practical Examples**

Real problems need more than you expect / know

- Graduate students know some tools to deal with engineering apps
- Most graduate programs offer a small sample of algorithms and technologies
- Graduate students then only know very basic concepts
- Real problems seldom admit the constraints of basic tools
- A complex real application needs advanced algorithms and technologies
- Research in algorithms, software, AI, and new technologies is full of them
- Just few techniques that can be used as described in books
- To work well, they need to be improved…

How do we improve them?

Important questions

What is the computational complexity of your algorithm?

- Measure it as O(n) O(logn) O(n•logn) $O(n^2)$ $O(n^3)$ … $O(2^n)$ … O(n!) … $O(n^n)$ …

How much simple is a technique? Occam's razor principle applies

- If more complex than one with similar behaviour, then not interesting

How measure complexity: computational, software, understanding…?

- In terms of input, branches, length of description, time to learn it, …

What is defining the limits of a technique or a technology?

- Its complexity, but also its accuracy in solving a problem, its robustness…

How a technique could be improved? And a technology?

- New design (operations, concepts), new implementation, latest hardware, …

Similarities to other existing tools? Do they inspire to improve?

- See the basics of the tool, similar structure, know on cross-fertilization

Can we quantify all the decisions? Identify all the needed steps?

- Data driven decision making, always measure … scientific method

Always measure: efficacy measures

MSE – Mean Squared Error:

- Risk measure (quality of estimator)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Logarithmic Loss:

- Penalising the false classification

$$LogLoss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} * \log(p_{ij})$$

Accuracy

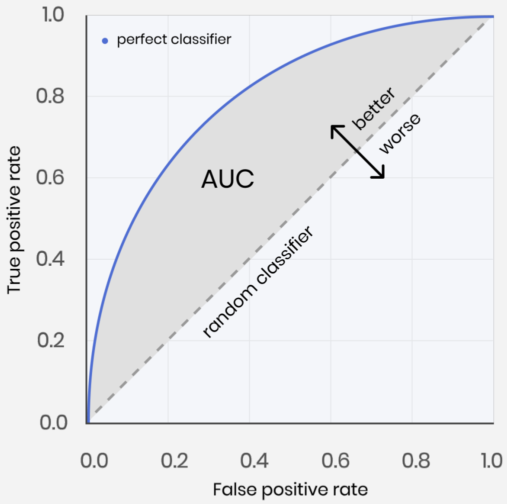
$$Accuracy = \frac{\#\ correct\ predictions}{\#\ input\ samples}$$

Confusion Matrix

		True Class	
		Positive	Negative
Predicted Class	Positive	TP	FP
	Negative	FN	TN

Area under de curve (AUC)

- How well the test separates the group being tested into 2 classes?
- TPR = TP/(TP + FN) FPR = FP/(FP + TN)



Always measure: efficiency measures

Wall Clock Time

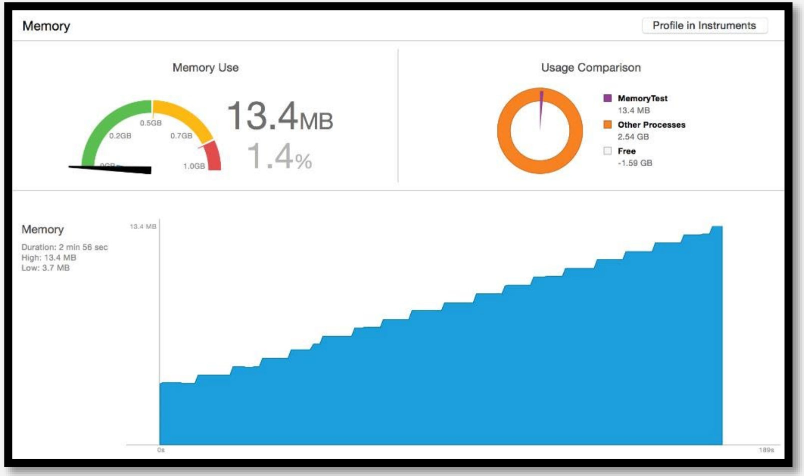
- $T = t_{end} - t_{start}$

User time
CPU time
Communication time
…

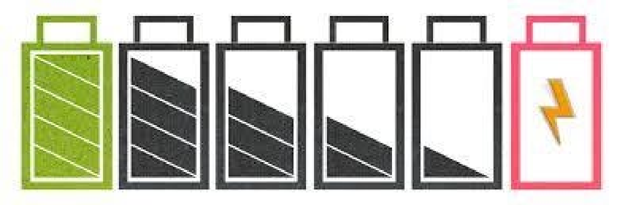
Speedup

$$S_m = \frac{\bar{T}_1}{\bar{T}_m}$$

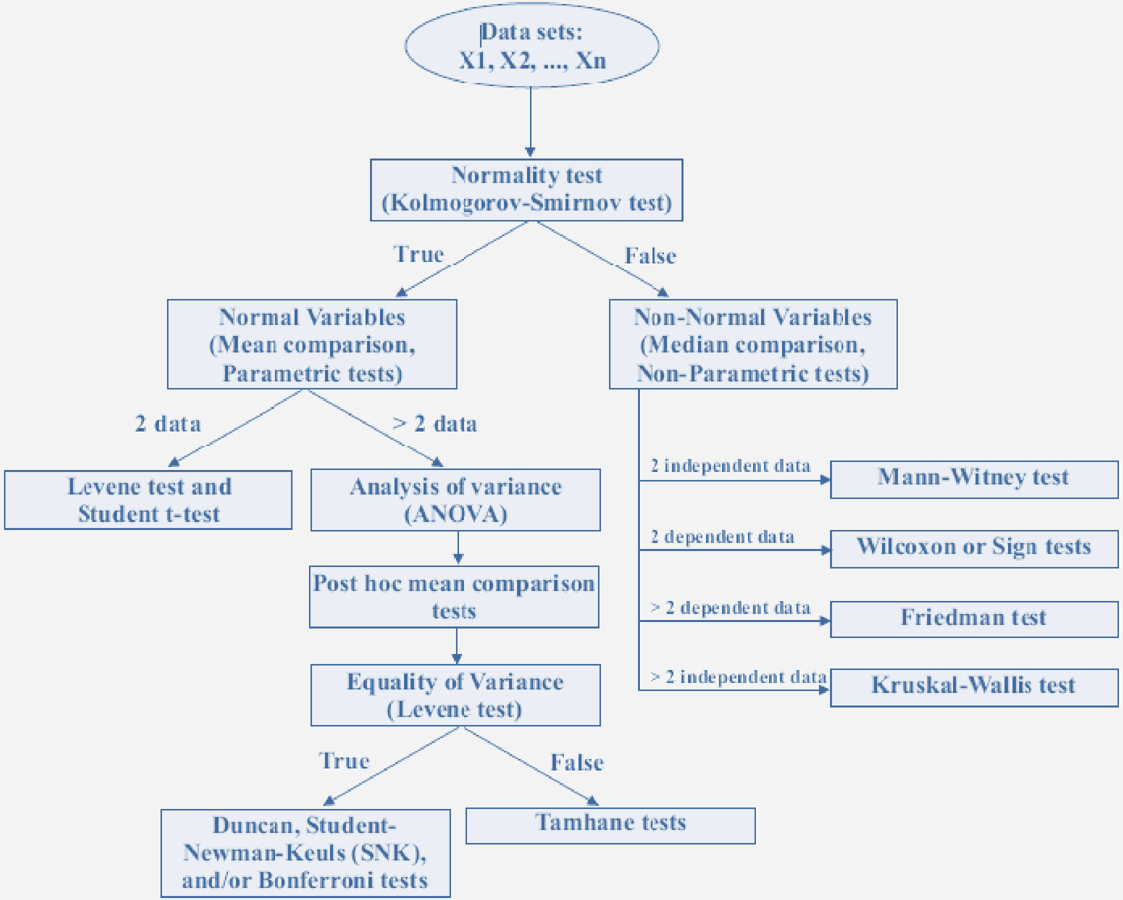
Memory usage
Size of data files



Battery consumption (phone)
Kwh consumption (data center)



Statistical analysis mandatory



Parallelism: parallel algorithms plus HW

- Even very advanced algorithms reach a **maximum efficiency**. This happens in large problem instances, or when using simulators, or in real time scenarios, or in web services for clients, ...
- Advances in **parallel hardware** like clusters, multicores, GPUs, cloud, etc. allow to make more than one step per unit time in the used techniques
- Sometimes you are not only looking for **reduced times**, but for **new types** of techniques that search for different solutions at the same time collaborating
- Sometimes you have a multicore or a lab full of cores: **explode them** !!!
- Thus, you can make new techniques and also run them faster, **both** !!!

Problems not solved before, now become solvable by using parallel tools

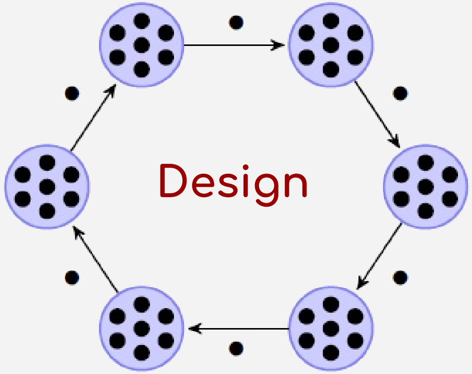
Parallelism: parallel algorithms plus HW

Parallelism and Metaheuristics:

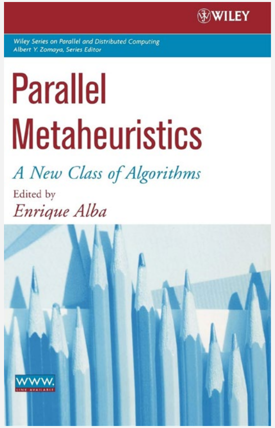
The increasing availability of new kinds of CPUs and the parallel nature of metaheuristics have allowed the fast development of parallel metaheuristics

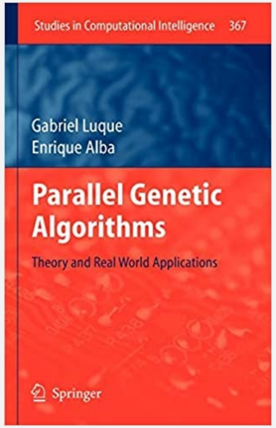
Advantages:

- Allow to tackle more complex problems/instances
- Allow to reduce the execution time
- Allow to improve quality of the found solutions

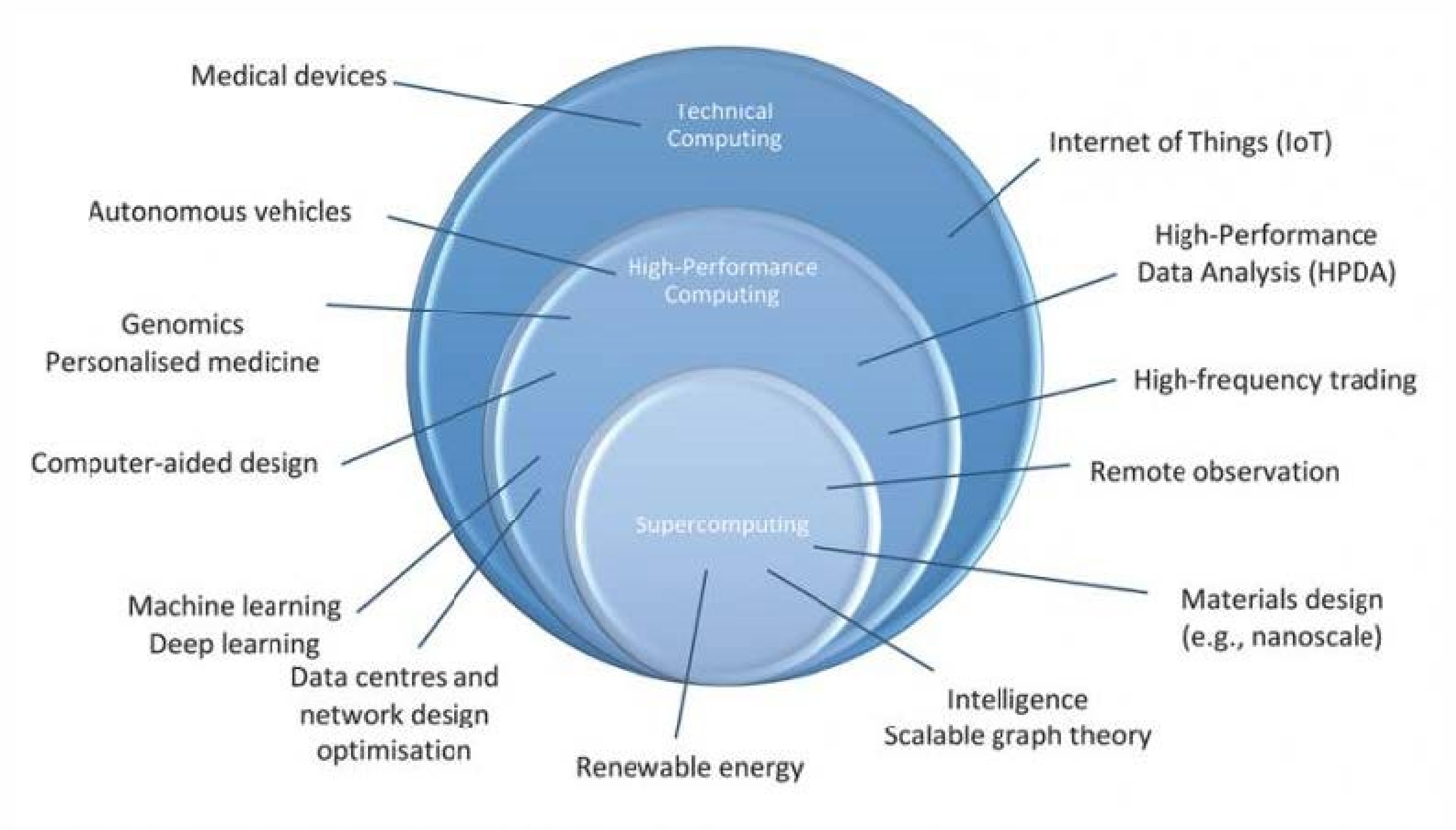


Implementations





High performance computing and folks



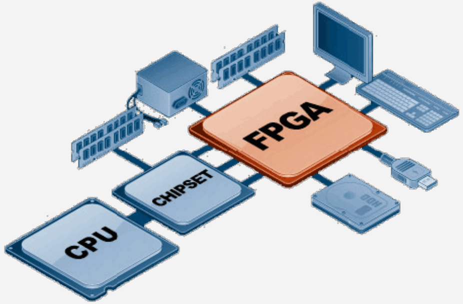
Hardware is important

Grid and cloud computing

Cluster computing

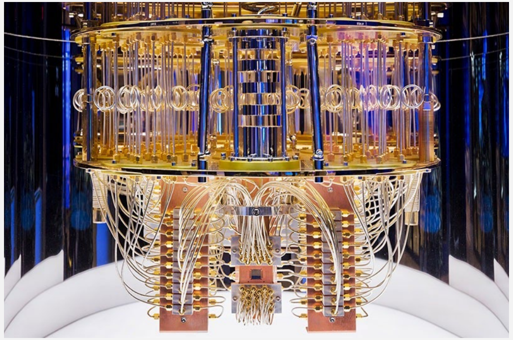
GPU

FPGA



Manycores

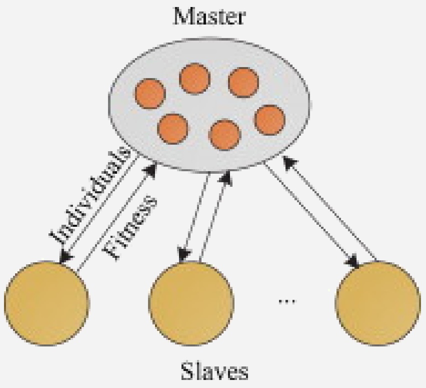
Quantum Computers

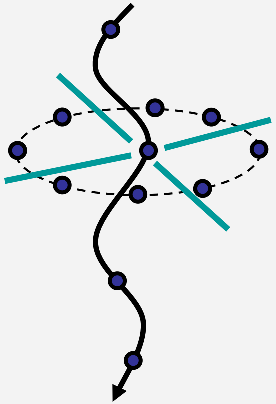


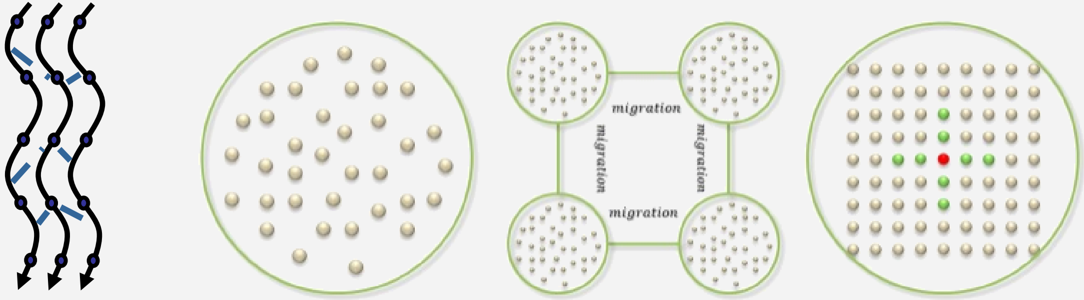
Design vs implementation, not the same

- Node in a distributed EA

```
t := 0
initialize(P(t))
evaluate(P(t))
while not end condition do
        P'(t) := selection(P(t))
        P'(t) := recombination(P'(t))
        P'(t) := mutation(P'(t))
        evaluate(P'(t))
        P(t+1) := replacement(P(t), P'(t))
        <<Communication with neighbours >>
        t := t+1
end while
```





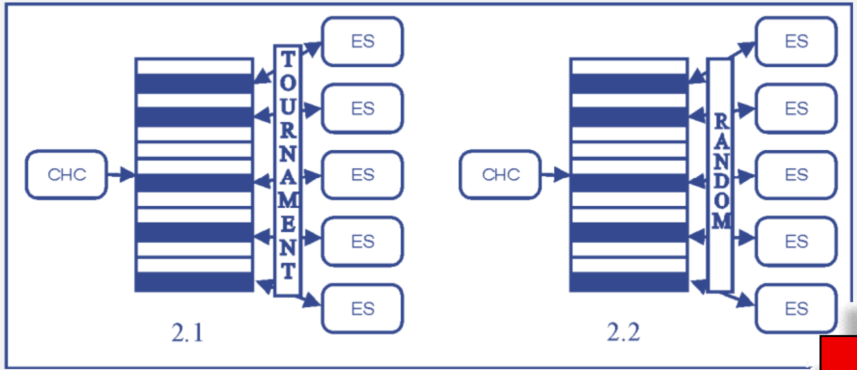


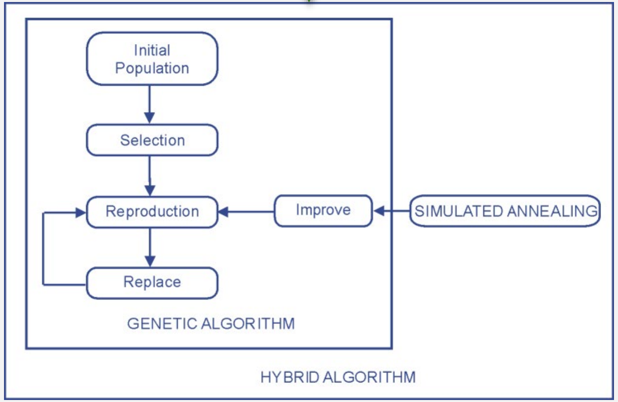
Hybridization: a good way to build techniques

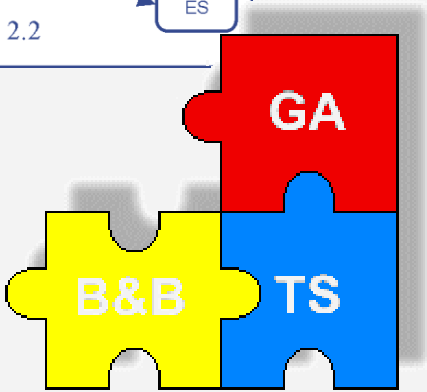
Hybridization is the inclusion of problem-dependent information in the algorithm, but also combining fields, operations, data, technologies, frameworks ...

Types

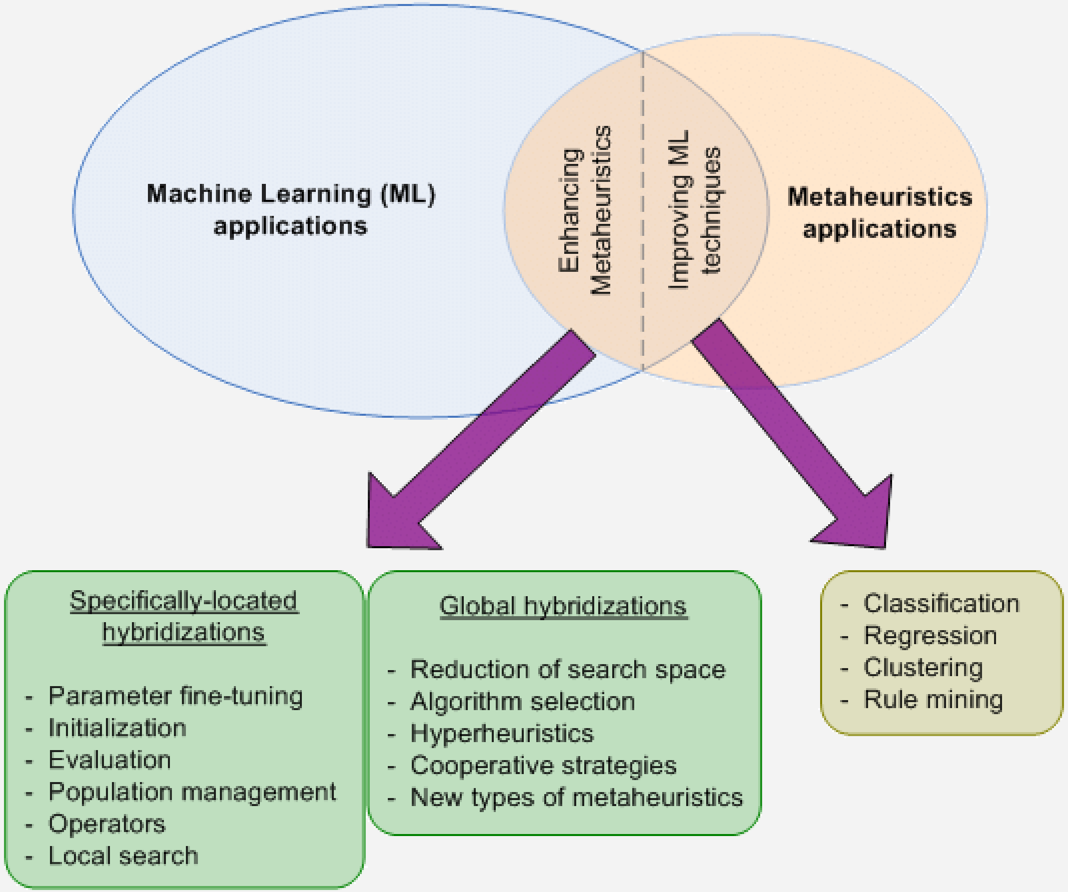
- Weak
- Strong







Hybridizing ML with metaheuristics



Optimization + Learning

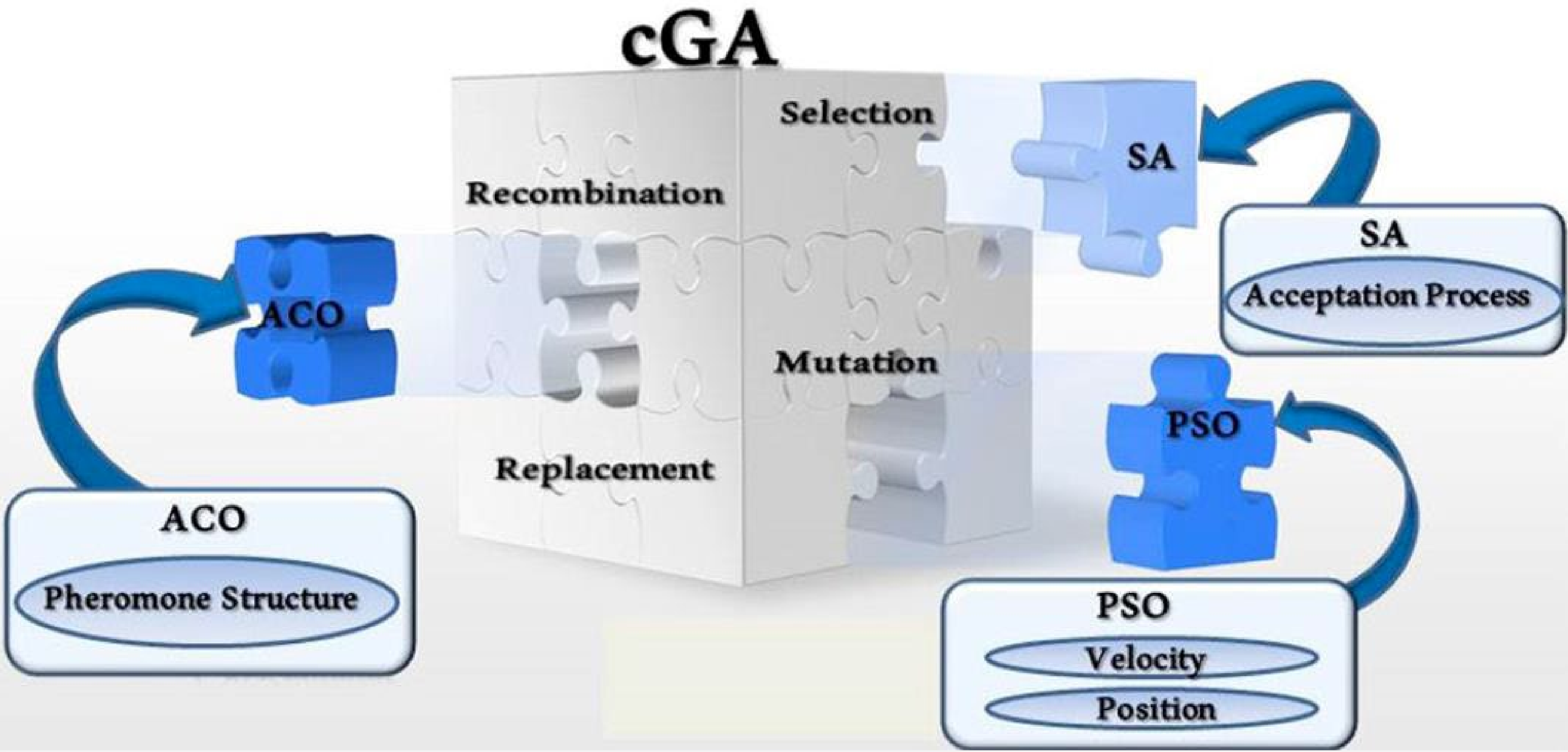
Hybrids = ML + MetaH

Tuning ML tools

Surrogate systems

Collaboration processes

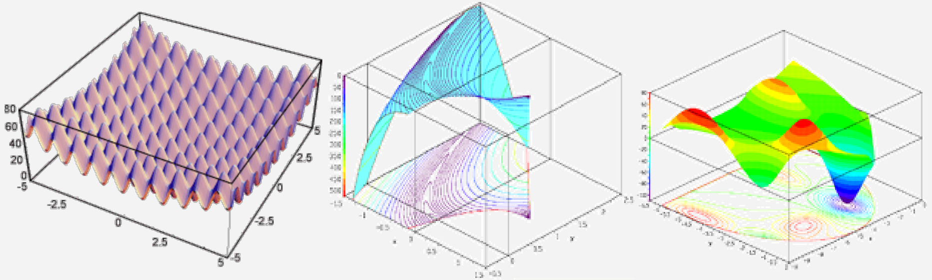
Building Blocks



Academic problem domains

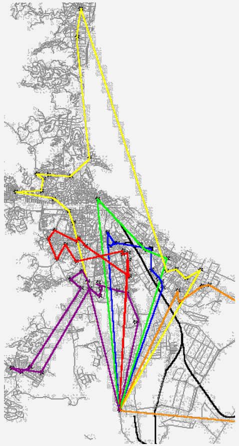
Mathematical Optimization:

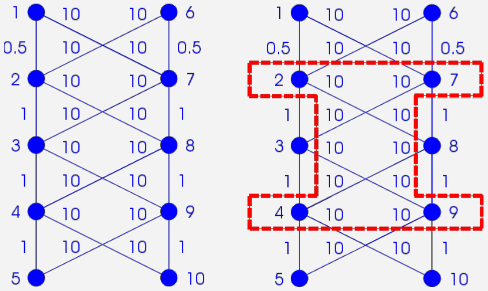
- Rastrigin, Rosenbrock, Mishra's Bird…



Combinatorial optimization:

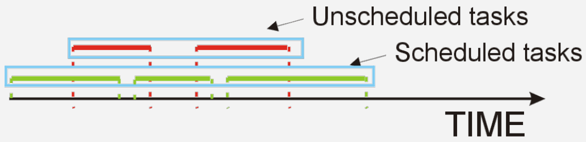
- Routes, scheduling, graphs…





Domain dependent benchmarks:

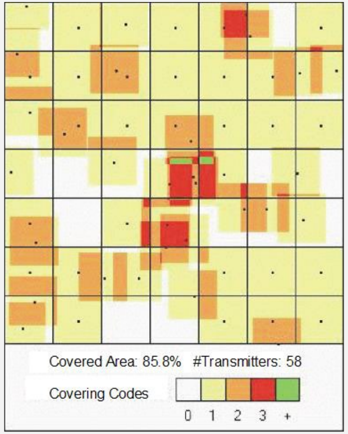
- Multiobjective
- Temporal series
- Data mining
- Neuronal network training



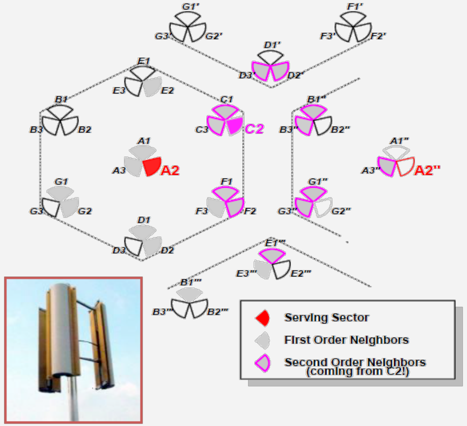
Know on standard benchmarking!!!

Sectoral domains: Telecoms in this case

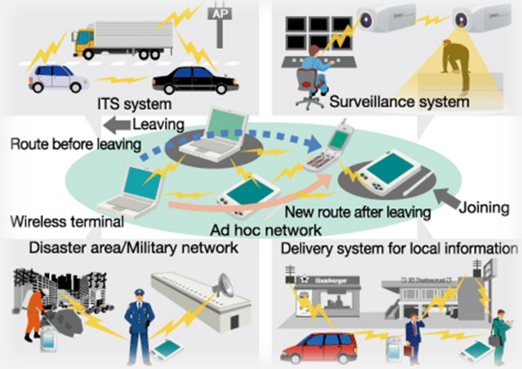
Radio Network Design



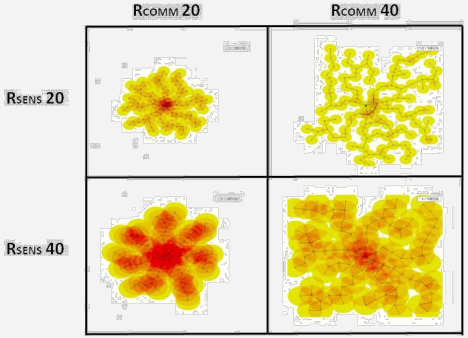
GSM Frequency Assignment



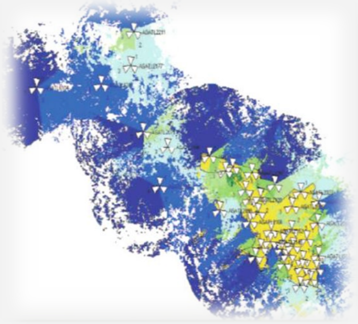
MANETs



Sensor Network Layout



Location Area in 4G/5G

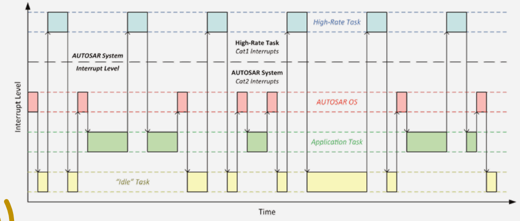


VANETS

Every single domain is a good target

Designing Quantum Circuits

Data Based (Data Mining, Query Optimization)

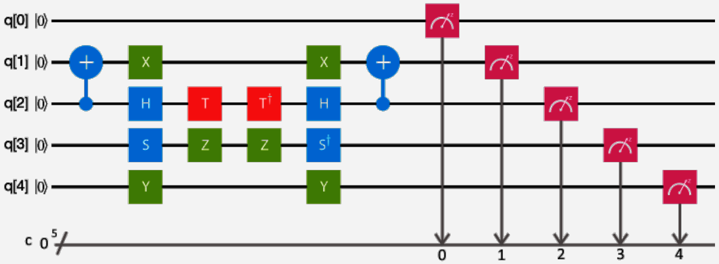


Dynamic Optimization Problems (DOPs)

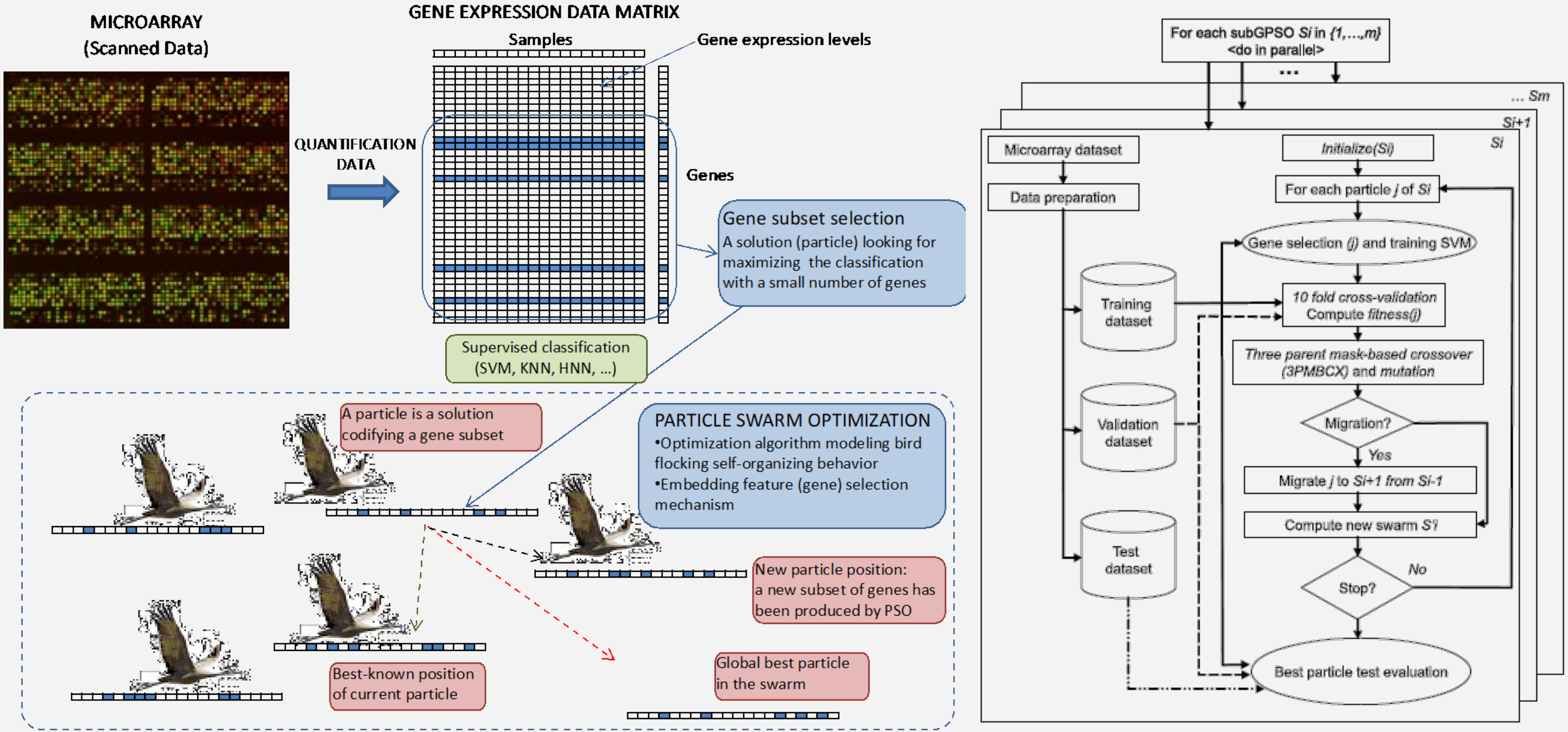
Tasks Scheduling in Operating Systems

Genomics (Fragment assembly, protein structure)

Games



Parallelism, Hybridization, and a real application



Real Use Cases

Data Science and Engineering (I)
Master's Degree in Computer Engineering

UNIVERSIDAD DE MÁLAGA

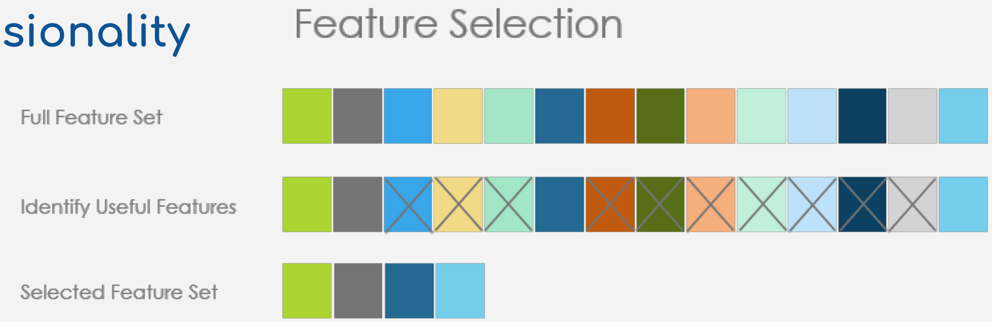
E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

Learn more in going for real applications

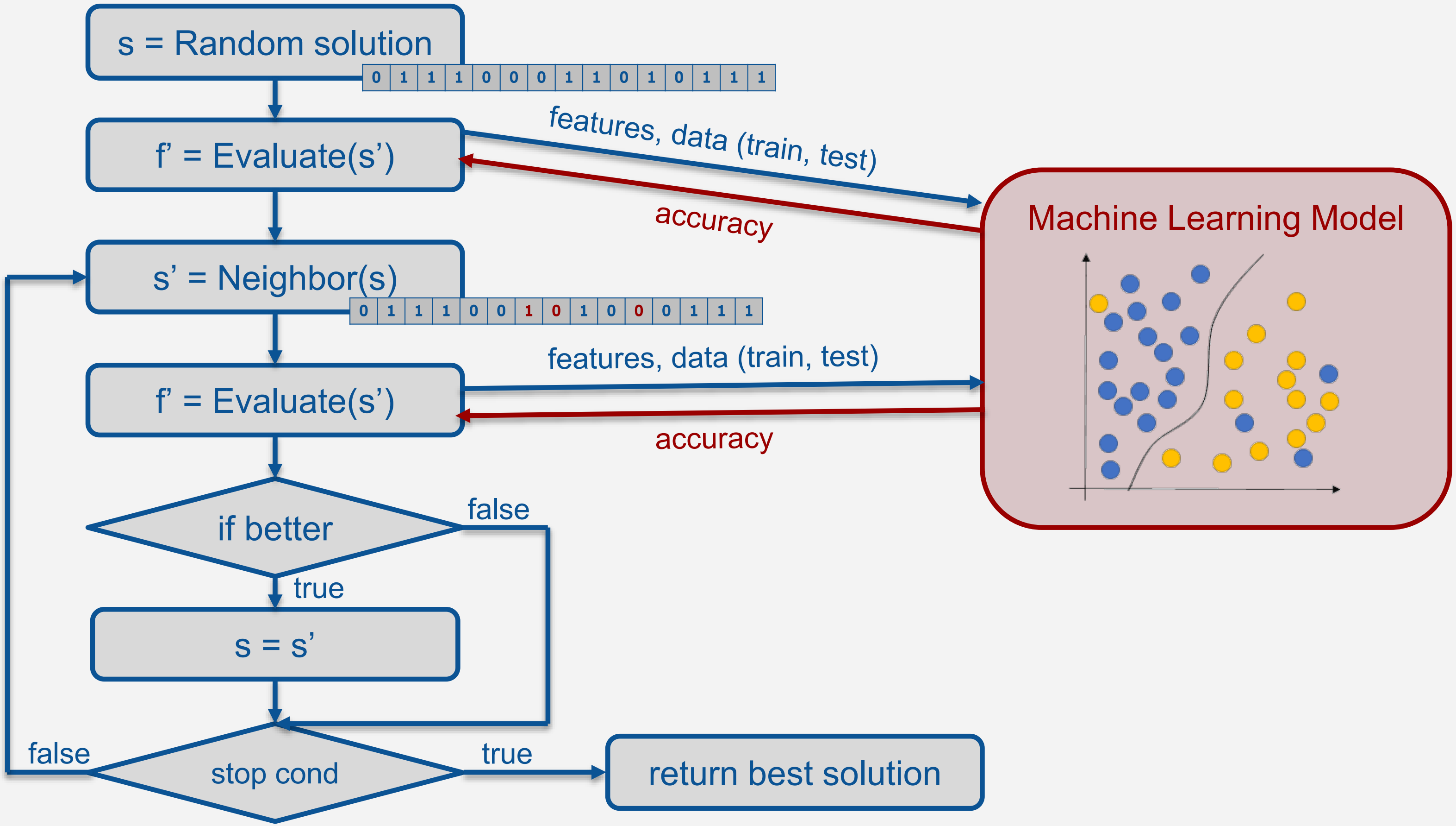
1. Feature selection
2. Neuroevolution
3. Surrogate models
4. Real applications

Feature selection

- The computational ability of machine learning models **depends** a lot on the **feature set**.
- Retaining the significant features vastly **improves** the learning **time**, and also improves **accuracy**.
- In **feature selection**, we find the **optimal feature subset** that contributes most to our predicted variable.
- **Advantages**:
  - **Improve generalization** of models by reducing overfitting of data.
  - **Remove** unnecessary/redundant data.
  - Curtail the **Curse Of Dimensionality**
  - **Optimize** training time



Feature selection as optimization problem



Feature selection as optimization problem

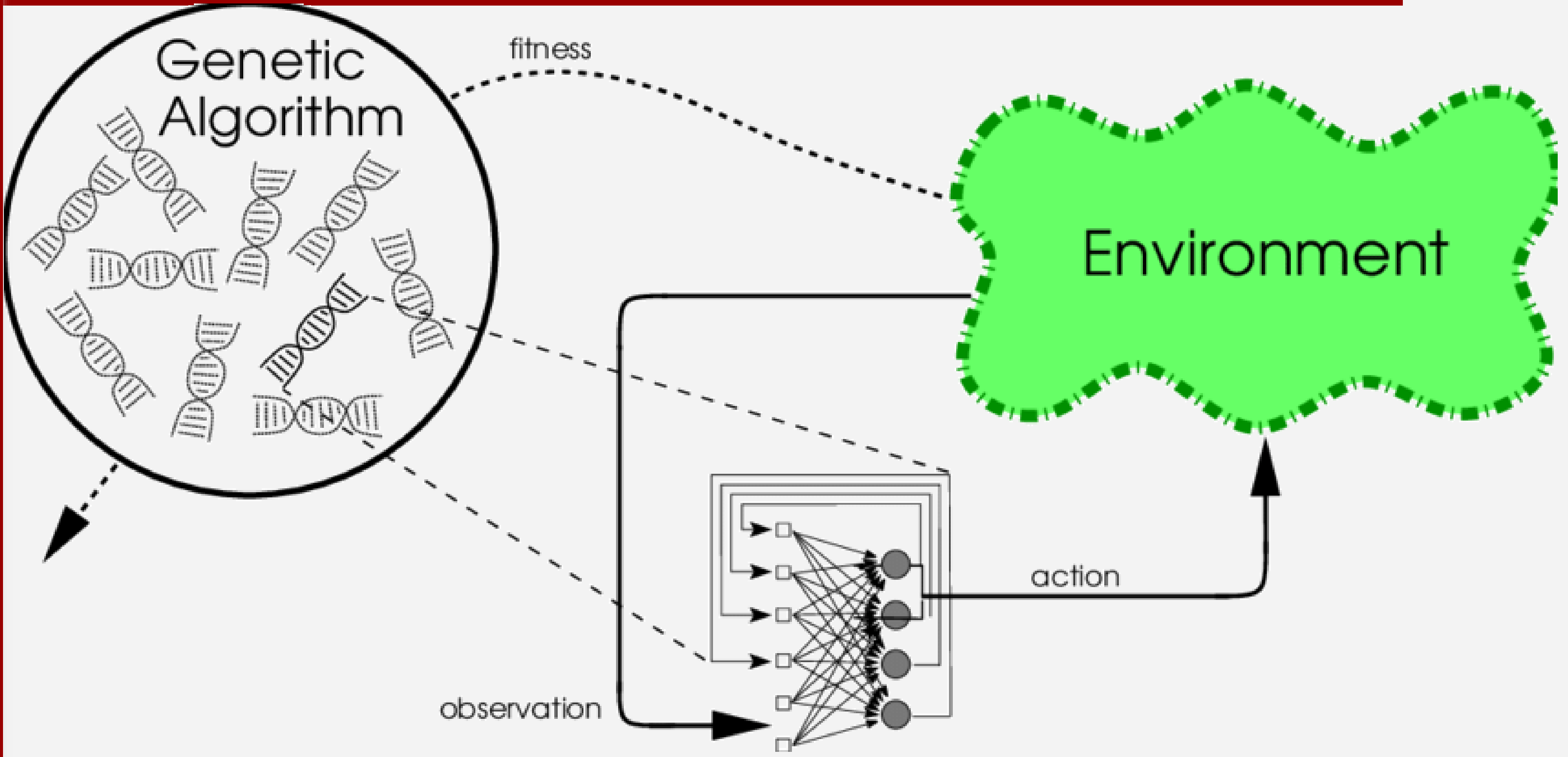
- **Solution encoding:**
  - The solution can be implemented as a bit string.
  - The solution's length is taken as the number of features in the dataset.
  - 0/1 indicates the presence/absence of the ith feature in the solution.

0	1	1	1	0	0	0	1	1	0	1	0	1	1	1

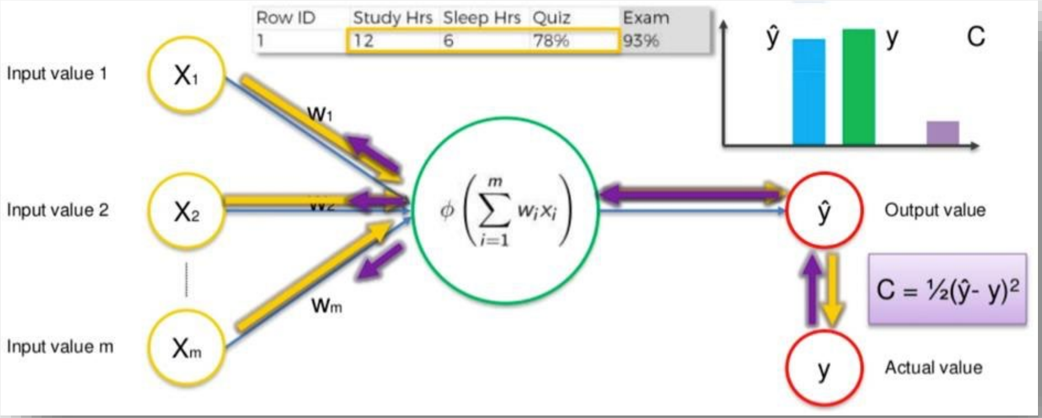
- **Fitness function:**
  - Number of features selected (nof)
  - Model accuracy (acc)
- **Approaches:**
  - Multiobjetive
  - Aggregate function:

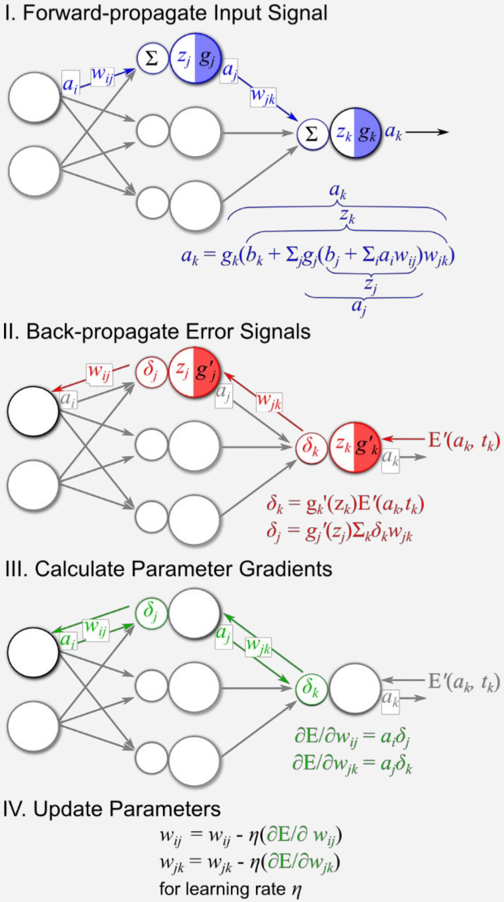
$$max\ f = \alpha \cdot \frac{MaxFeatures - nof}{MaxFeatures} + (1 - \alpha) \cdot acc$$

Neuroevolution



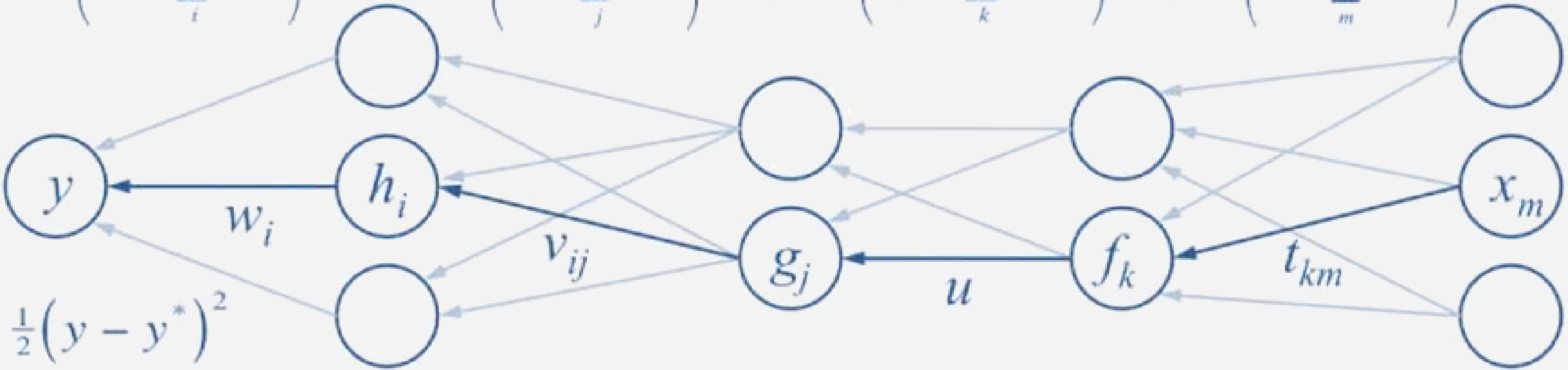
Forward propagation of inputs





Backpropagation of errors

$$y = \sigma\left(w_0 + \sum_i w_i h_i\right) \qquad h_i = \sigma\left(v_{i0} + \sum_j v_{ij} g_j\right) \qquad g_j = \sigma\left(u_{j0} + \sum_k u_{jk} f_k\right) \qquad f_k = \sigma\left(t_{k0} + \sum_m t_{km} x_m\right)$$



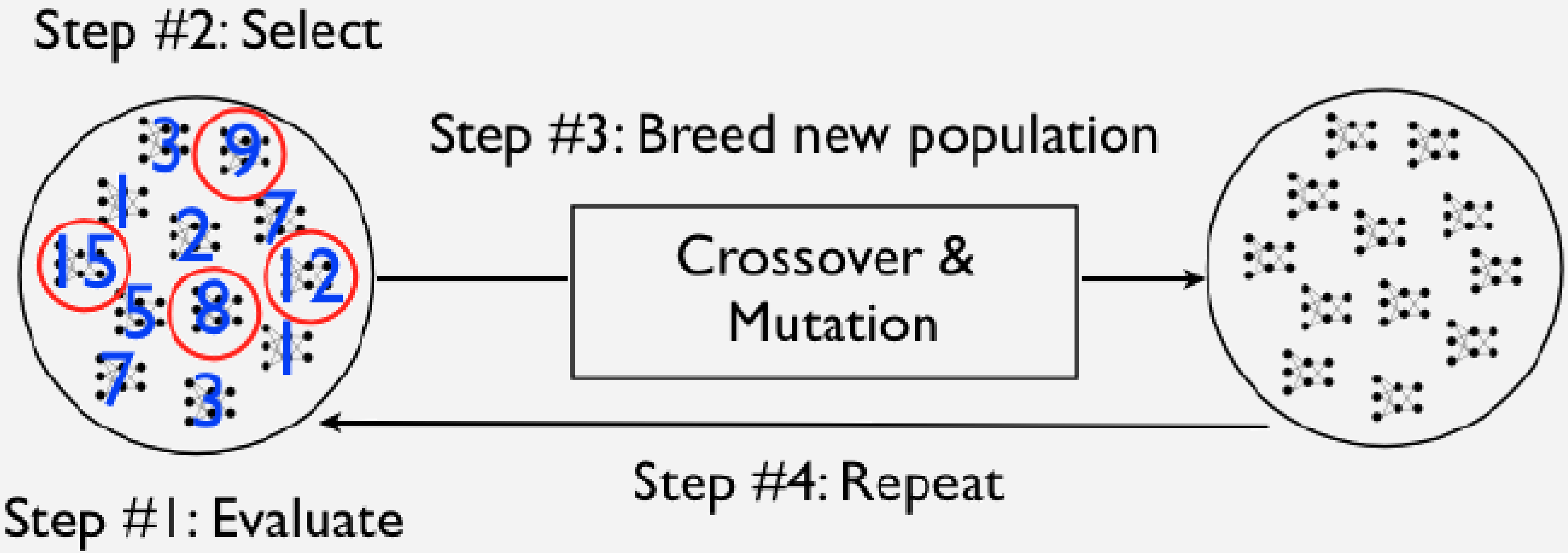
$$E = \frac{1}{2}\left(y - y^*\right)^2$$

$$\frac{\partial E}{\partial h_i} = \left(y - y^*\right) y(1-y) w_i$$

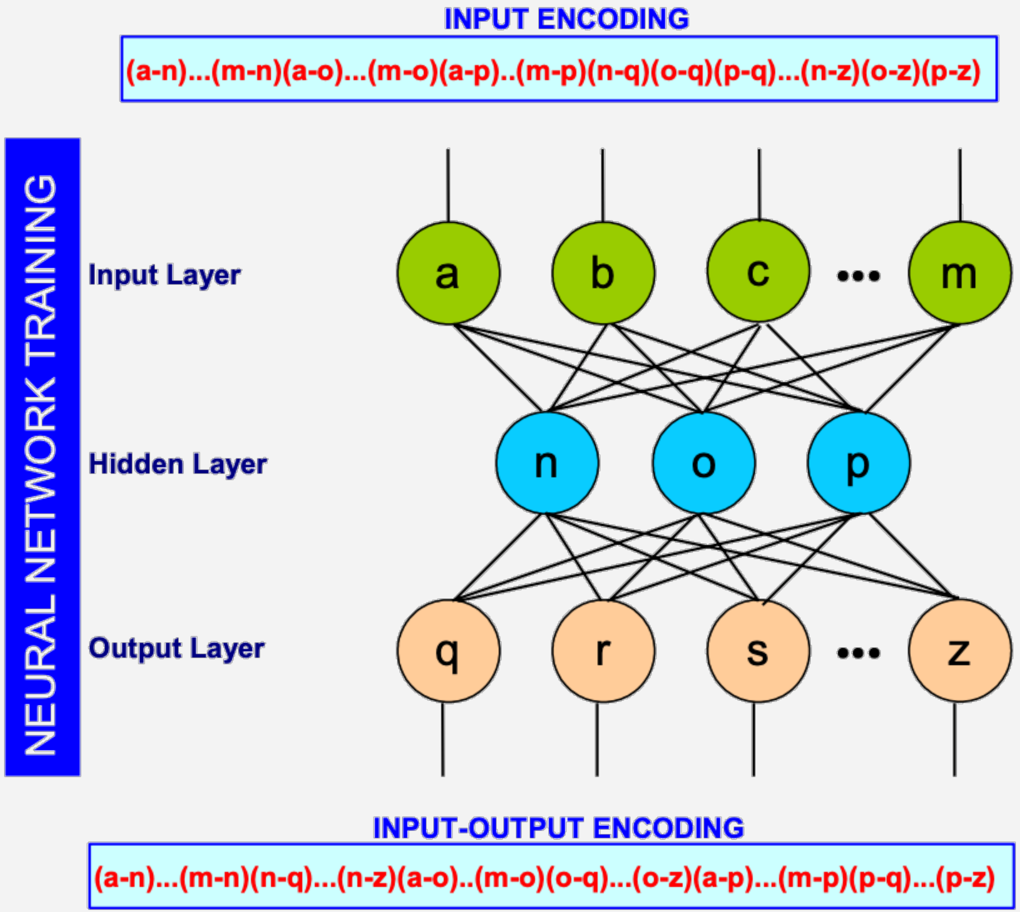
$$\frac{\partial E}{\partial g_j} = \left(y - y^*\right) y(1-y) \sum_i w_i h_i (1 - h_i) v_{ij}$$

- Prone to local optima
- Not appropriate for complex NN
- Oscillations
- Depends on structural functions
- Need unfolding in deep learning

Neuroevolution

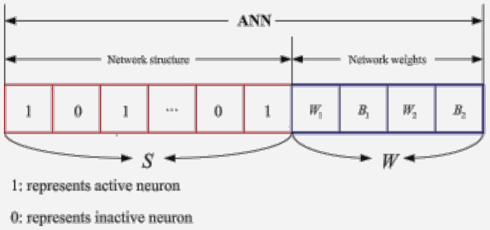


Encoding weights vs. backpropagation

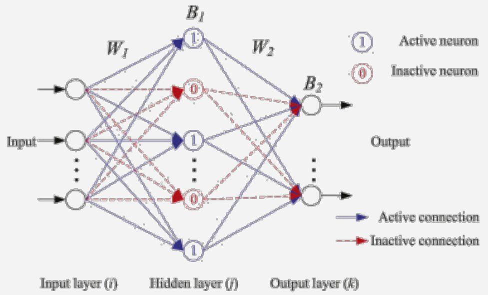


- Solution encoding:
  - List of float numbers
  - Mapping between poisition in list and weight in NN
- Fitness function:
  - Model accuracy

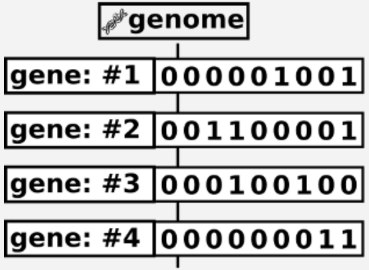
Evolving NN structures!!!

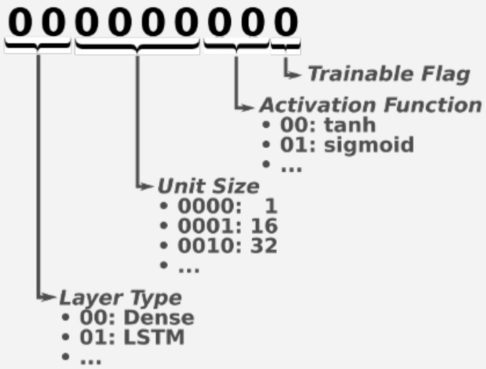


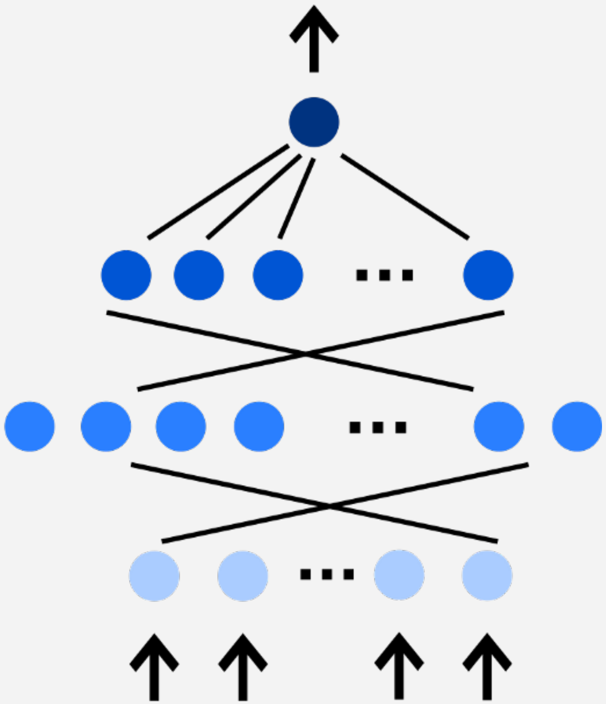
(i) Mixed-coding scheme



(ii) The configurations of three-layer feedforward ANN







Genotype

Translation Faculty

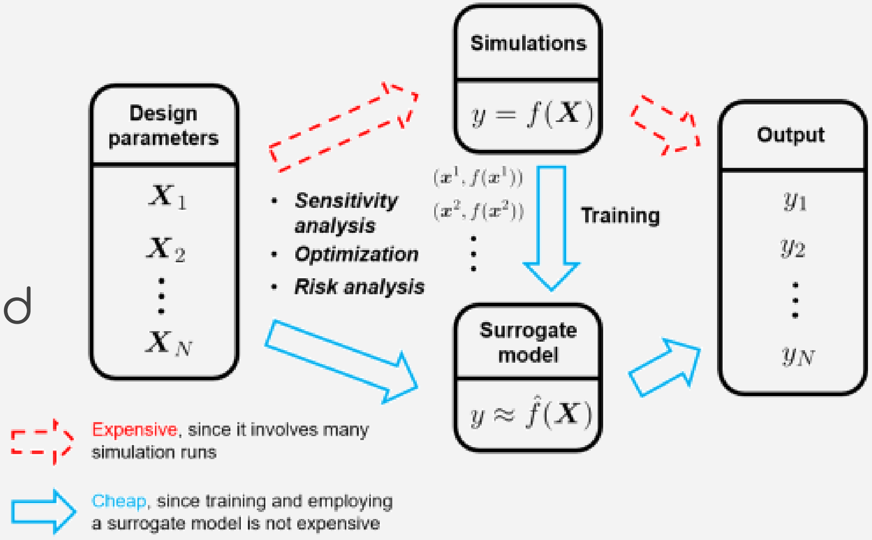
Phenotype

Surrogate models

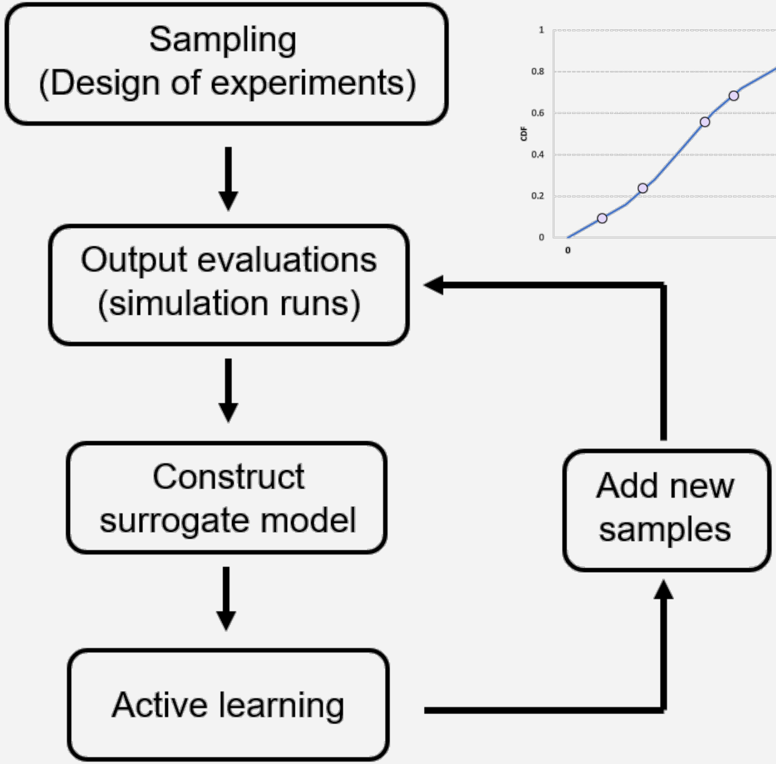
- A lot of engineering **problems** require experiments and/or **simulations** to evaluate design objective and constraint functions as a function of design variables.
- A single **simulation** can take many minutes, hours, or even days to complete, thus rendering them infeasible in practice.

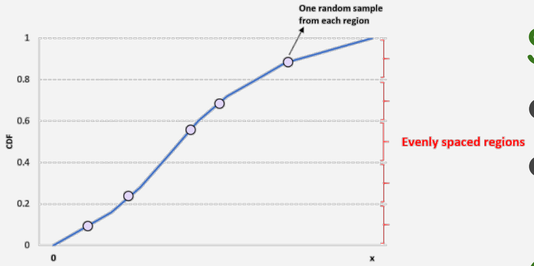
Surrogate models are a statistical model to accurately approximate the simulation output.

This **trained model** can be deployed to replace the original computer simulation.



Surrogate models





Sampling:
- Random
- Latin hypercube

Construct model:
- Model selection
- Tuning parameters
- Feature selection?

Utilization:
- Only surrogate model
- Mixed model:
surrogate + simulation
- New samples?

Real Applications: An example

Traffic Control Signals

- **Introduction**
  - Problem
  - Approaches
  - Proposal
- **Traffic Control Signals**
  - Concepts
  - Current systems
  - Proposed automatic system
- **Solving**
  - System modeling
  - Realistic data
  - Surrogate model

Problem

- City evolution:
  - Nowadays, cities are growing in the number of inhabitants, many of whom are arriving at the city for the first time
  - By 2050 the human population will reach 9 billion with 75% of the world's inhabitants living in towns and cities

- As consequence, the number of vehicles in streets is continuously increasing, affecting all aspects of daily life:
  - Traffic jams
  - Pollution
  - Security
  - Stress
  - Economic losses
  - ...

Potential Solutions

- "Classic" solutions:
  - Infrastructures
  - Promote the use of public transportation or green vehicles (bikes)
  - Promote the use of car-sharing (VAO lanes)
  - Limiting car access to city centers
- "Intelligent" solutions:
  - Provide real-time and accurate data to citizens to make informed decisions (traffic intensity, free park slots, ...)
  - Automatic assistance tools: adaptative and/or customized routes
  - Better tuning of existing elements: routes and frequencies of public transportation, traffic light timing...

Proposal

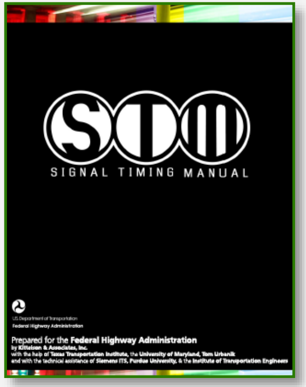
Automatic Traffic Control Signals

- Reduce the traffic jams
- Minimize the waiting times in red lights
- Faster routes
- Reduce the gas emissions

Traffic Control Signals

- First, we need to **study** the elements, constraints, and regulations in **the problem domain**

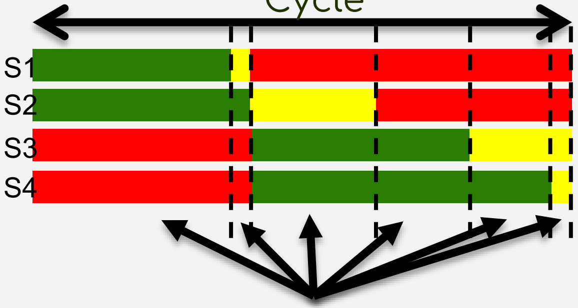
- Multiple sources of information:
    - **International regulations**. (I.e., U.S. Transport Department):
        - Manual on Uniform Traffic Control Devices (862 pages)
        - Traffic Signal Timing Manual (274 pages)
    - **National regulations**. (I.e., DGT):
        - Regulación semafórica (32 pages)
        - Cruces semafóricos y sincronismo (32 pages)
    - **Specialized personnel** (city traffic managers)
    - **Scientific literature**



- Information filtering

Traffic Control Signals

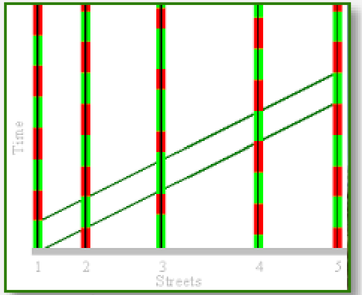
- **Important concepts:**
  - Intersections
  - Cycle
  - Phases
  - Traffic light schedule or plan



Intersection

Traffic Control Signals

- More information:
  - The duration of phases and cycles can be modified
  - The phases CANNOT be modified
  - Recommended duration of a cycle: 60-120 seconds
  - Yellow phases (before red light): 4 seconds
  - Minimum duration of some phases (i.e., red phases at crosswalks should allow to safely cross the road => minimum duration = 1 m/s * 4 m/lane * #lanes)
  - Promote green waves in important avenues
  - Traffic-dependent planes (time, weekday, season,...)
  - ...



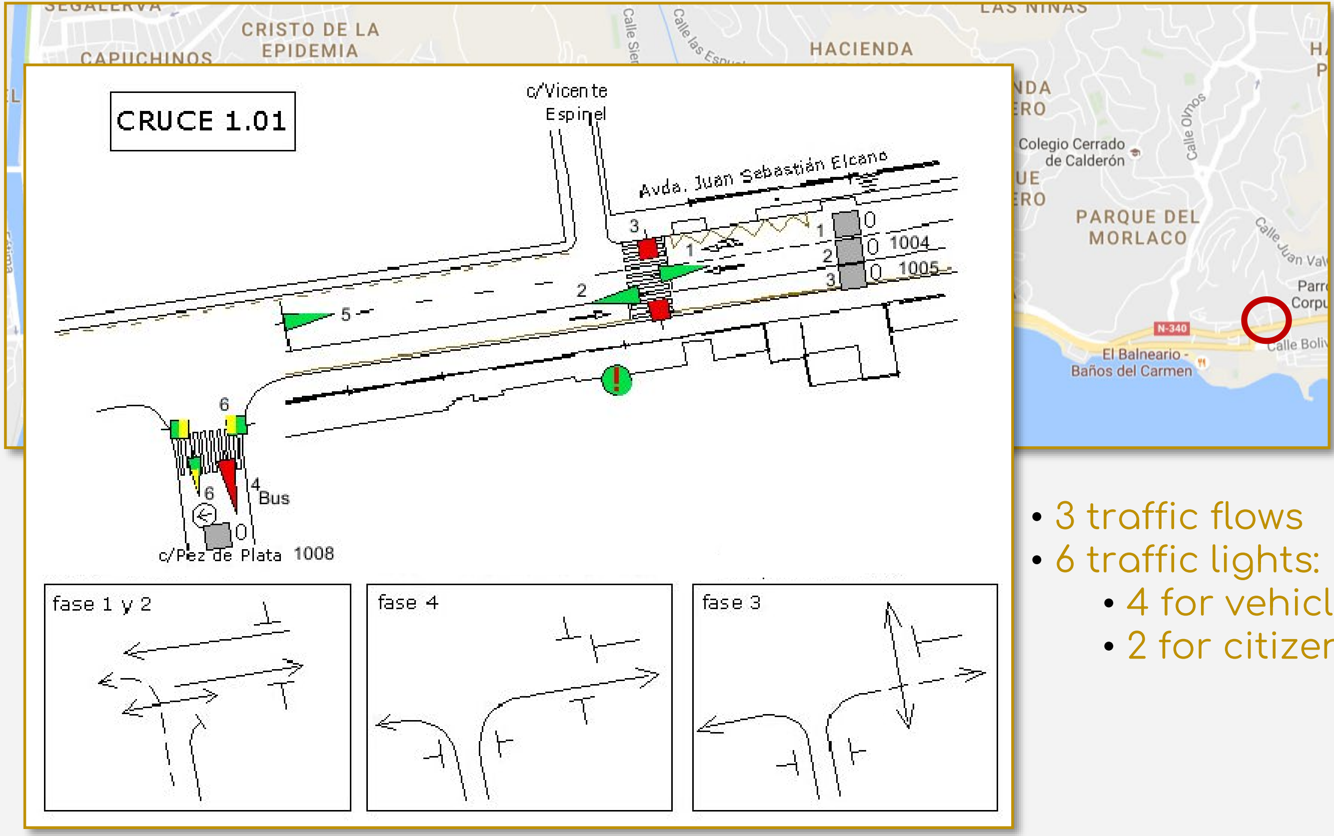
Green wave

How are traffic lights configured?

- Location/type:
  - Some locations are mandatory according to regulations
  - Other locations are recommended but not mandatory

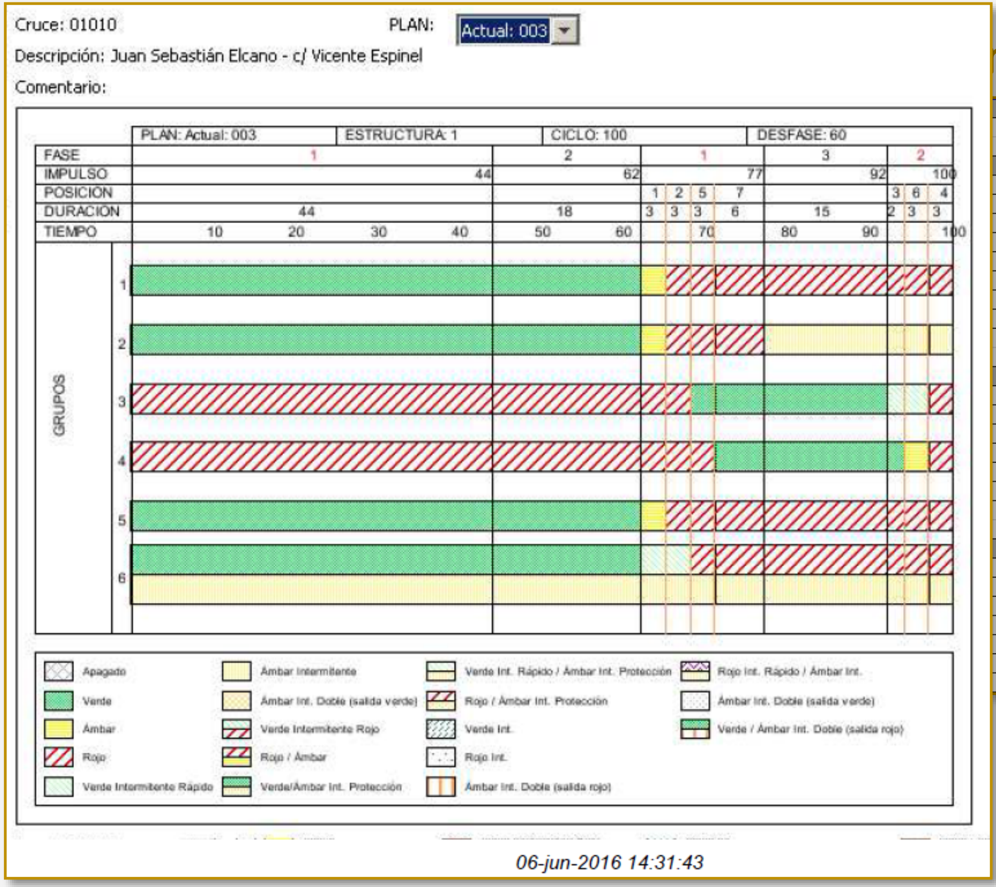
- Phases of traffic lights:
  - Regulations establish a procedure to set the phases
- Duration of phases/cycles:
  - It is defined by city traffic managers according to some constraints
  - Usually, it is manually done in each intersection
  - Based on experience and accumulated knowledge
  - There are dynamic systems (they react to current traffic). Problem: Quick changes that don't improve traffic

How are traffic lights configured?



- 3 traffic flows
- 6 traffic lights:
    - 4 for vehicles (1 for bus)
    - 2 for citizens

How are traffic lights configured?



- ...ns that are chosen

SELECCIÓN HORARIA - VERANO 2015

SUB 1 LABORABLE

Hora	Plan	Ciclo	Sentido
0:00	12	80	Oeste
2:00	13	70	Oeste
5:00	12	80	Oeste
6:30	20	115	Oeste-Simultáneo (118 a 115)
11:00	5	110	Oeste
16:00	2	115	Oeste
20:30	5	110	Oeste
22:30	7	100	Oeste

SUB 1 SABADO

Hora	Plan	Ciclo	Sentido
0:00	7	100	Oeste
3:00	12	80	Oeste
10:00	5	110	Oeste
14:00	7	100	Oeste
17:30	5	110	Oeste
22:30	7	100	Oeste

SUB 1 DOMINGO

Hora	Plan	Ciclo	Sentido
0:00	7	100	Oeste
3:00	12	80	Oeste
10:00	11	90	Oeste
11:30	10	95	Oeste
16:00	7	100	Oeste
17:00	20	115	Oeste
22:30	7	100	Oeste

Proposed system

- Automatically generated cycle and phase times:
  - Those times must respect the constraints
- Simultaneously consider all traffic lights in the city or the area defined by the traffic control center
- Obtain different plans (offline) according to traffic intensity
- The final goal is to obtain more fluid traffic that reduces the pollution

Modeling the problem

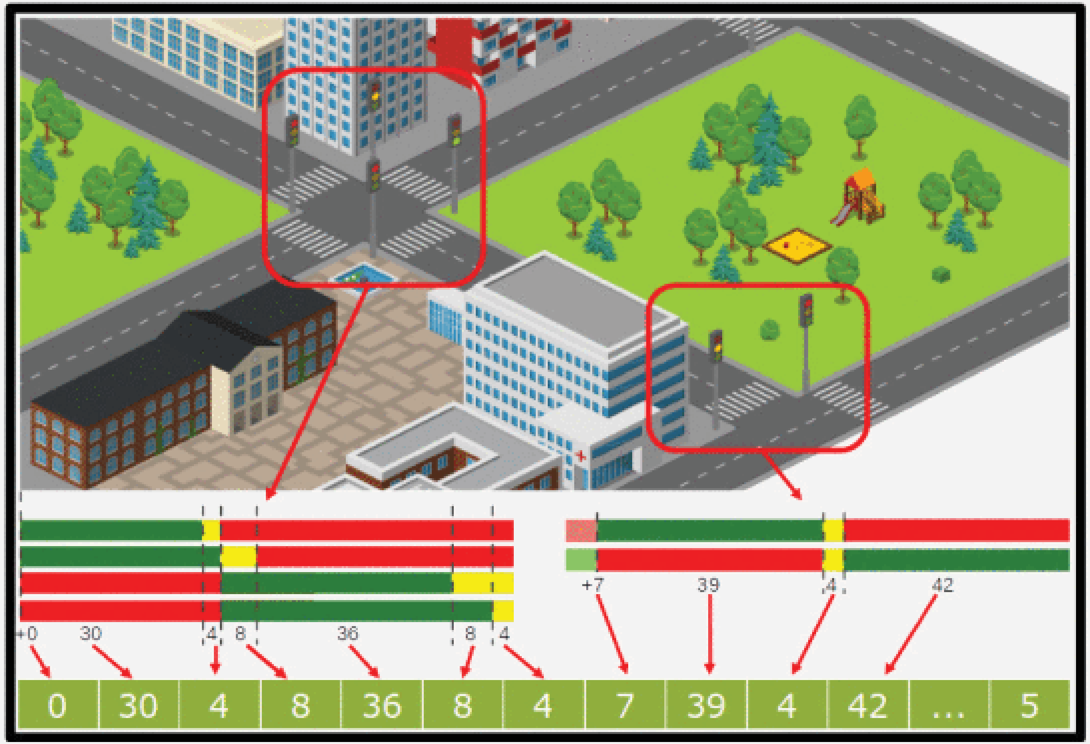
- Given the following **input data**:
  - **Intersections** to improve (location, phases)
  - And the **traffic flows**

- **Objective**: to find the configuration (duration for the phases) that outperforms the rest of the existing configurations

- **Questions**:
  - What is computationally a solution (**representation**)?
  - When is a configuration **better** than another one (from a numerical point of view)?
  - Where do we get the **information** from?
  - How do we **find the best**?

Solution encoding

- List of numbers (duration of phases):



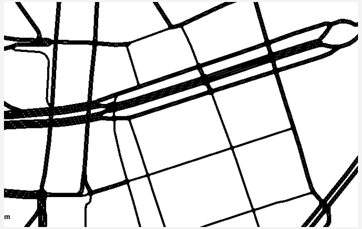
Solution encoding

- Restrictions not met by default:
  - Yellow phases to 4
  - Cycles > 60 and < 120
  - Green Wave Promotion
- Other alternative representations:
  - Each intersection: cycle time + percentage that each phase occupies
  - Reduce the number of traffic lights:
    - Cluster the intersections into groups
    - Only one in the group is optimized
    - The rest are small variants of the optimized
  - Others?

Solution fitness

- To compare solutions, we need quantitative values (fitness)
- The traffic system is very complex
    - No (realistic) mathematical models
    - Utilization of simulators
- SUMO (Simulator of Urban Mobility)
    - Input: Roadmap, traffic flows, traffic light plans
    - Output: statistics of the simulation

20	8	47	23	30	10	60	4	...	29



SUMO

Vehicles reaching destination: 30
Average trip time: 120
CO2: 543452
Average waiting time: 10
...

Solution fitness

- Statistics about simulation:
  - Number of vehicles that reach their destination during a given simulation time
  - Average trip time
  - Em
  - W

Trip duration

Waiting times

Vehicles not reaching destination

$$f_{obj} = \frac{T_{trip} + Tsw + VNRTsim}{V_R^2 + P}$$

Vehicles reaching destination

Green phases duration

- Fitnes
  - O
  - C
  - Multi-objective approach
  - Derivate values (green waves?)

- Additional challenge:
  - SUMO only calculates statistics for vehicles that reach their destination

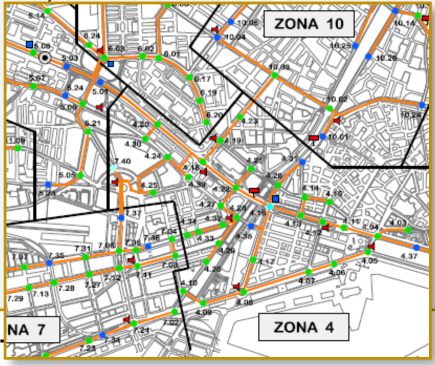
Realistic data

- Input for our system:
  - City map
  - Location of traffic lights
  - Vehicle flow according to several factors (time, weekday, or season)
  - Constraints in phase duration

- Source of the data:
  - **Maps**: OpenStreetMap
  - **Traffic lights**: traffic control center of the city
  - **Routes**: Mobility department

Realistic data

- Challenges: Maps and traffic lights
  - Incomplete maps: missing road, road directions, traffic lights, ...
  - Errors in conversion (OSM => SUMO)
  - Manual correction of the maps. Labour intensive process



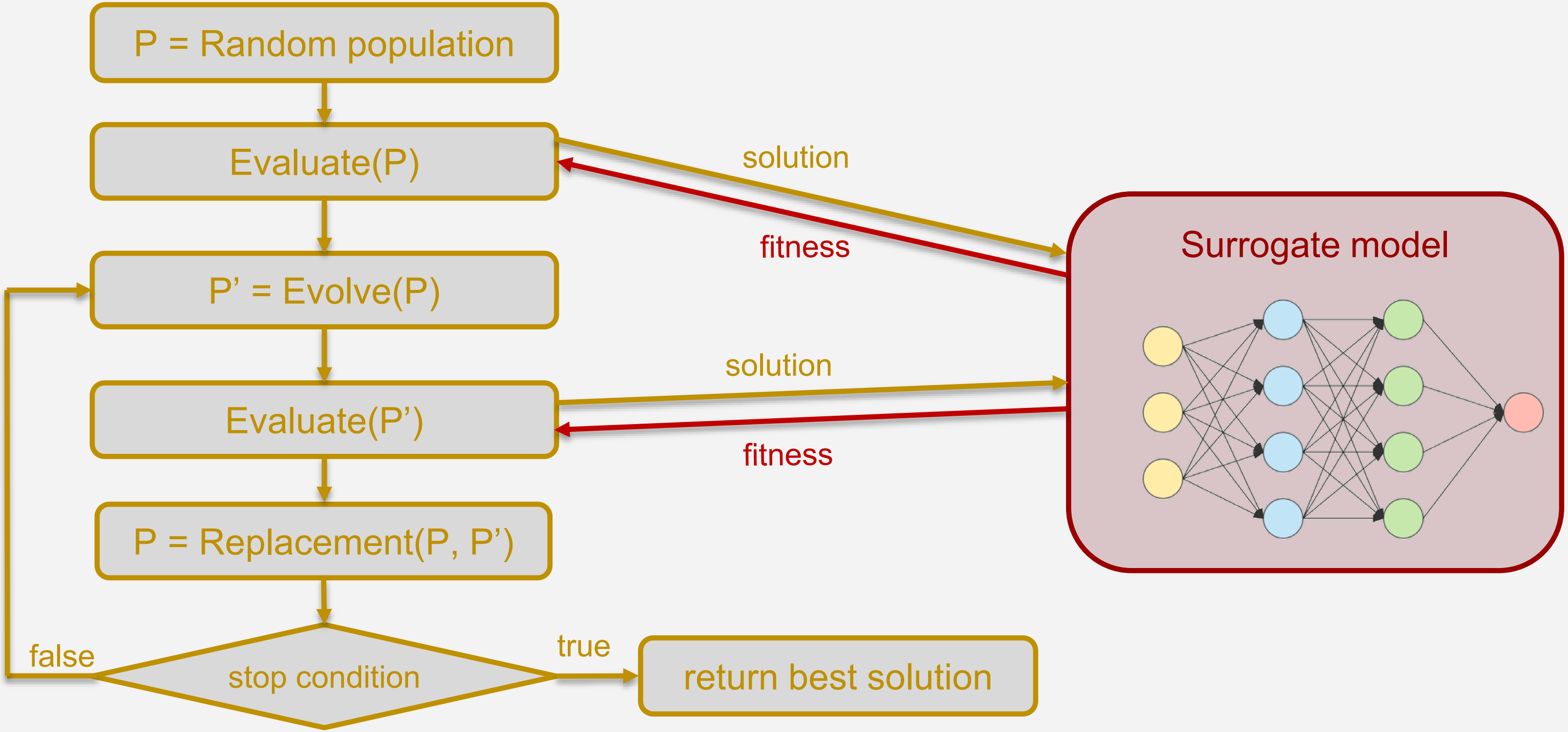
- Challenges: Routes
  - Not enough details
  - Non-automatable format
  - Conversion of traffic intensity to traffic flows

Intensidad de vehículos Mayo-

PM	Ubicación	I.M.D.L	I.M.D.S.	I.M.D.D.	I.M.H.P.L.	H.P.L.M.	I.M.H.P.S.	H.P.S.M.	I.M.H.P.D.	H.P.D.M.
1	Avda. Juan Sebastián Elcano - Este	2.004	1.636	1.388	169	12:00	149	12:00	138	13:00
2	Avda. Juan Sebastián Elcano - Oeste	20.383	17.890	15.186	1.465	8:00	1.278	21:00	1.236	21:00
3	Bolivia - Este	16.775	15.174	12.698	1.413	14:00	1.250	14:00	1.091	13:00
4	P.M. Pablo Ruiz Picasso - Este	19.376	15.574	12.631	1.908	14:00	1.337	14:00	1.044	13:00
5	P.M. Pablo Ruiz Picasso - Oeste	28.829	23.675	20.368	2.192	8:00	1.632	21:00	1.582	21:00
6	Pso. Reding - Este	8.528	6.767	5.296	687	14:00	554	14:00	418	13:00
7	Pso. Reding - Oeste	6.987	5.930	4.655	546	9:00	445	12:00	342	21:00
8	Victoria - Sur *	5.874	5.255	4.285	419	8:00	443	22:00	319	22:00
9	Victoria - Norte *	6.474	5.888	4.923	483	14:00	454	22:00	360	22:00
10	Túnel Alcazaba - Este	15.870	14.744	12.561	973	8:00	849	14:00	701	1:00

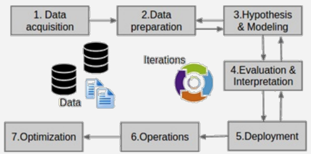
Algorithms

- GA + ANN as surrogate model



Summary

Management of DS proyects



Challenges in real-world problems

- Scalability
- Dynamism
- Uncertainty
- Robustness
- Multiple Objectives
- Contraints

Approaches

- Efficiency: Parallelism
- Efficacy: Hybridization
  - Surrogate models
  - Neuro-evolution
  - Feature Selection

Real-world Application: TLSP

+

Surrogate models

Neuro-evolution

Feature Selection