Metaheuristic Algorithms

- What is this course about?
- Difficult optimization problems
- Metaheuristics taxonomy
- Main topics

What is course about ?

- As almost all courses in Computer Science it is about problem solving
- Its main aim is to present techniques to solve hard problems
- There are problems which are hard:
  - for computers but rather easy for humans (e.g. character recognition, face recognition, speech recognition etc) – ill posed problem
  - both for humans and computers (e.g. large combinatorial optimization problems, multimodal / multiobjective optimization problems etc) – computationally hard problems

Computationally hard problems

- Problems characterized by a large space of solutions for which there are no exact methods of polynomial complexity (so-called NP hard problems) – they are characterized by a huge size search space (which cannot be exhaustively searched)

Examples:

- Satisfiability problem (SAT): find the values of boolean variables for which a logical formula is true. For n variables the search space has the size 2^n

- Travelling Salesman Problem (TSP): find a minimal cost tour which visits n towns. The search size is (n-1)! (in the symmetric case, it is (n-1)!/2)

Ill-posed problems

- The particularity of problems which are easy for humans but hard for computers is that they are ill-posed, i.e. there is difficult to construct an abstract model which reflects all particularities of the problem

- Let us consider the following two problems:
  - Classify the employees of a company in two categories: first category will contain all of those who have an income larger than the average salary per company and the second category will contain the other employees
  - Classify the employees of a company in two categories: first category will contain all those which are good candidates for a bank loan and the second category will contain the other employees

Ill-posed problems

- In the case of the first problem there is easy to construct a rule-based classifier:

  IF income > average THEN Class 1
                                          ELSE Class 2

- In the case of the second problem it is not so easy to construct a classifier because there are a lot of other interrelated elements (health status, family, career evolution etc) to be taken into account in order to decide if a given employee is reliable for a bank loan. A bank expert relies on his experience (previous success and failure cases) when he makes a decision

Well-posed vs Ill-posed Problems

- Differences between well-posed and ill-posed problems:

Well-posed problems:

- There is an abstract model which describes the problem
- Consequently, there is a solving method, i.e. an algorithm

Ill-posed problems:

- They cannot be easily formalized
- There are only some examples for which the results is known
- The data about the problem could be incomplete or inconsistent
- Thus, traditional methods cannot be applied

Ill-posed problems

The methods appropriate for ill-posed problems should be characterized by:

- Ability to extract models from examples (learning)
- Ability to deal with dynamic environments (adaptability)
- Ability to deal with noisy, incomplete or inconsistent data (robustness)
- Ability to provide the answer in a reasonable amount of time (efficiency)

Optimization challenges

Search space

- Large size
- Characterized by complex constraints

Objective function

- Many local optima
- „black-box” function
- Noisy function
- Multiple (conflictual) objective criteria

Remark:

- Traditional techniques (e.g. mathematical programming) are either inapplicable or inefficient

Classes of optimization problems

- Discrete search space → combinatorial optimization
    - Routing (e.g. vehicle routing)
    - Planning (e.g. Timetabling, task scheduling)
    - Allocation (e.g. resource allocation)
    - Selection (e.g. feature selection)
- Continuous search space → continuous optimization
    - Parameter estimation
    - Finding minimal energy configurations
    - Training adaptive systems

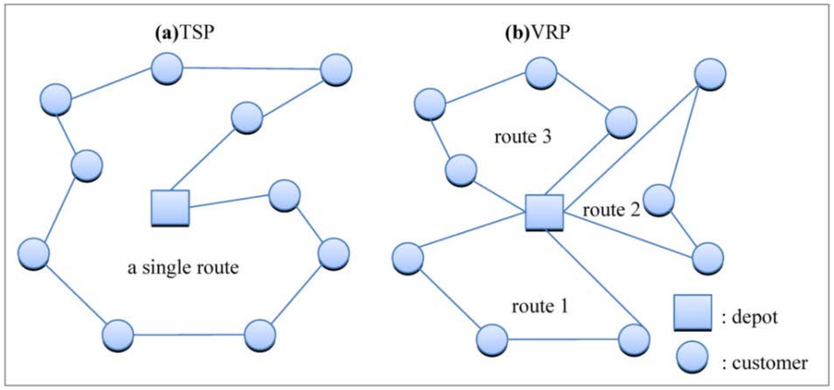
Routing

Vehicle Routing Problem:

- Find a minimal cost route which visit a set of locations and satisfy some constraints

Particular case: travelling salesman problem

TSP= find a hamiltonian circuit of a minimal cost in a complete graph



Solution representation:

1. Binary allocation matrix (nxn):

A_{ij} = 1 if node j is visited at step i
 = 0 otherwise

Constraints:

- Each row contains exactly one 1
- Each column contains exactly one 1

Search space size: 2^{n*n}

2. Permutation:

p_i = index of node which is visited at step i

Constraints:

- The elements of p are distinct

Search space size: n! (for symmetric problems (n-1)!/2)

Planning

Problem: Let us consider

- a set of events (e.g. lectures, exams etc.),
- a set of rooms
- a set of time slots

Construct a schedule such that each event is assigned to a room and a time slot such that different types of constraints are satisfied.

The constraints can be:

- hard (mandatory)
- soft (can be violated)

	S1	S2	S3
T1	E1	E3	E9
T2	E4		E8
T3	E6	E5	
T4	E2		E7

Search space size: $(k+1)^{mn}$

k events

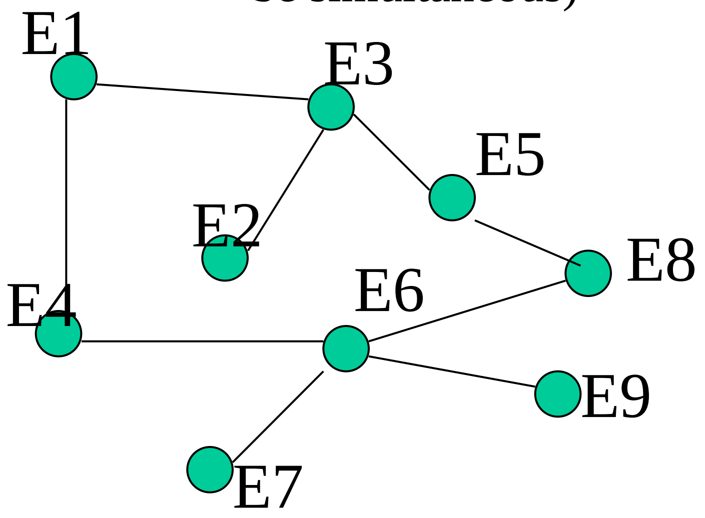
m time slots

n rooms

Planning

- Hard constraints (the solution is feasible only if they are satisfied):
  - Each event is scheduled exactly once
  - Only one event is scheduled in a room at a given time moment
  - The room satisfies the requirements of the event
  - There are simultaneous events at which same persons should attend

Conflict graph (two connected nodes correspond to events which cannot be simultaneous)



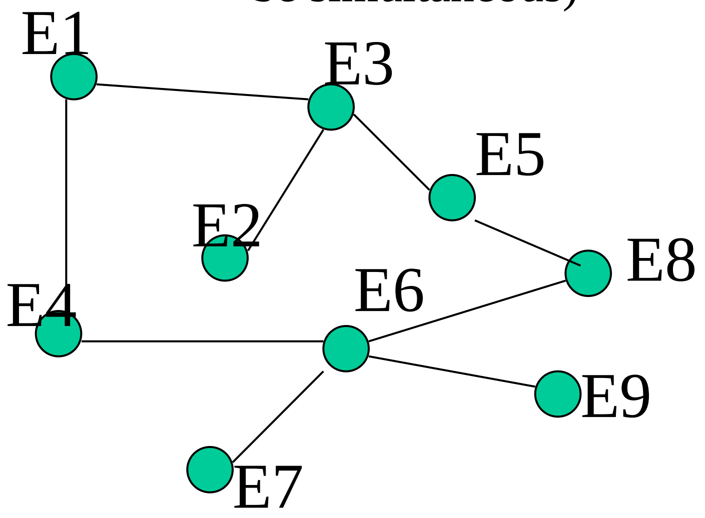
	S1	S2	S3
T1	E1	E3	E9
T2	E4		E8
T3	E6	E5	
T4	E2		E7

Planning

- Soft constraints:
  - There are no more than k successive events which should be attended by the same participant
  - There are no participants for which there is only one event/day

Idea: the soft constraints can be transformed in optimization criteria (e.g. The number of participants for which some of the constraints are violated is as small as possible)

Conflict graph (two connected nodes correspond to events which cannot be simultaneous)



	S1	S2	S3
T1	E1	E3	E9
T2	E4		E8
T3	E6	E5	
T4	E2		E7

Resource allocation

Problem: cloud resource allocation

Let us consider:

- A set of tasks characterized by some requirements
- A set of virtual machines (VMs) having some characteristics

The aim is to assign the tasks to VMs such that:

- The tasks requirements are satisfied
- The used virtual machines (or the global cost or the consumed energy) is as small as possible

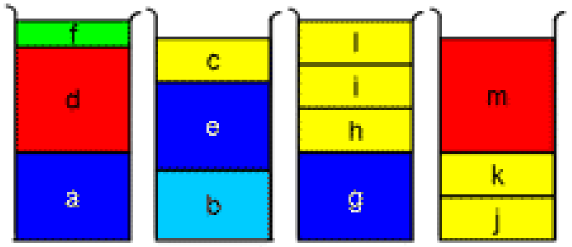
Particular case: bin packing

Solution representation:

(n tasks, m VMs)

V_i = index of the VM on which is placed the task i

Search space size: m^n



Selection

Feature/attribute selection

Let us consider:

- a dataset characterized by a large number of attributes

We are looking for a subset of features:

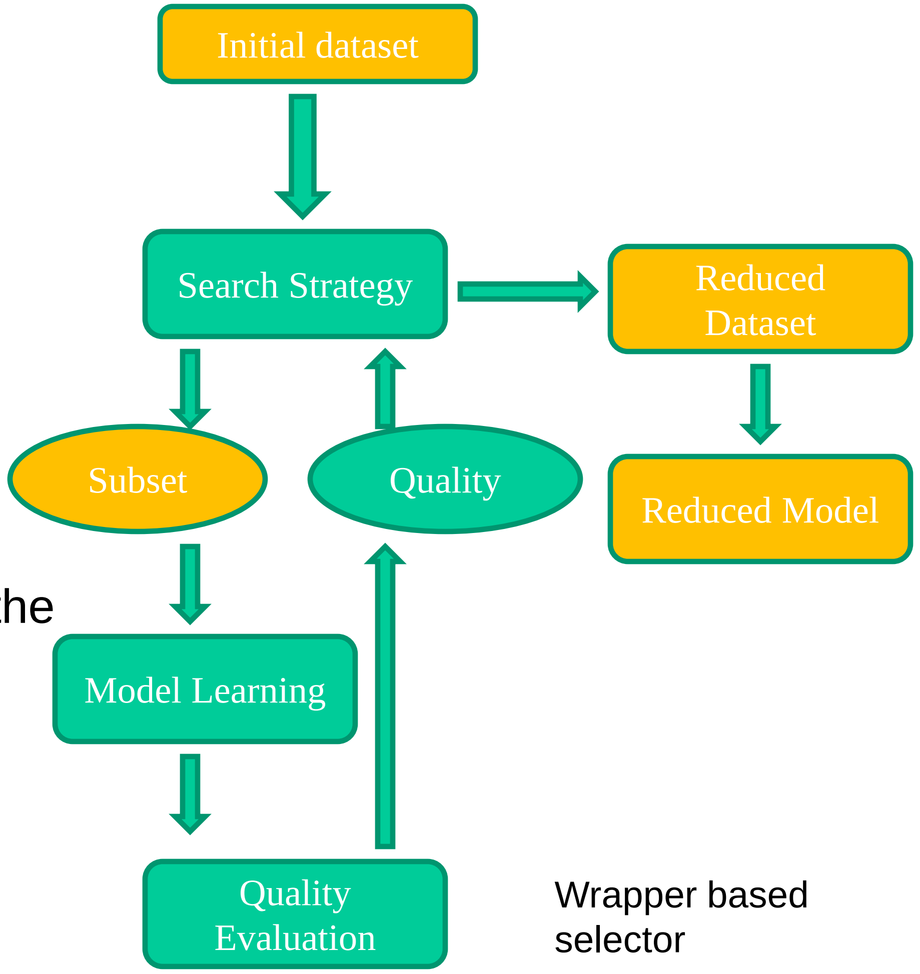
- Which maximizes the classification accuracy
- Minimizes the cost of data processing (and/or the size of the model extracted from data)

Solution: binary vector

S_i=1 if attribute i is selected

=0 if attribute i is not selected

Search space size: 2^n (n=initial number of attributes)



Wrapper based selector

Parameter estimation

Image registration

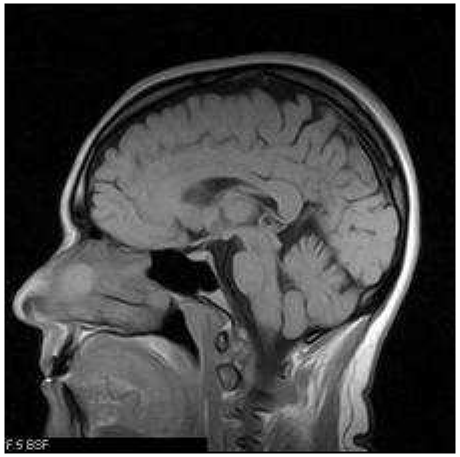
Let us consider two images of the same object which should be „aligned” = we search for an image transformation which maximizes the similarities between the corresponding pixels of transformed images

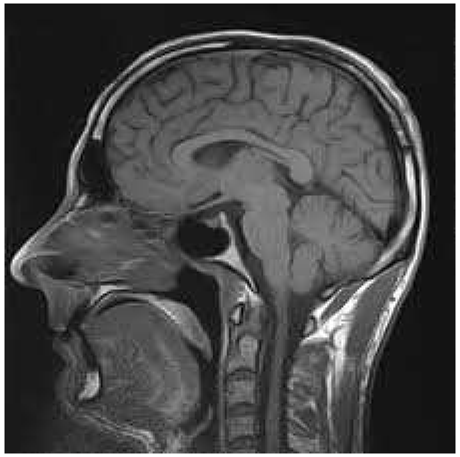
Solution representation:

Vector of real values corresponding to the parameters of the transformation

Search space:

$[a_1,b_1]x[a_2,b_2]x...x[a_n,b_n]$





Finding a minimal energy configuration

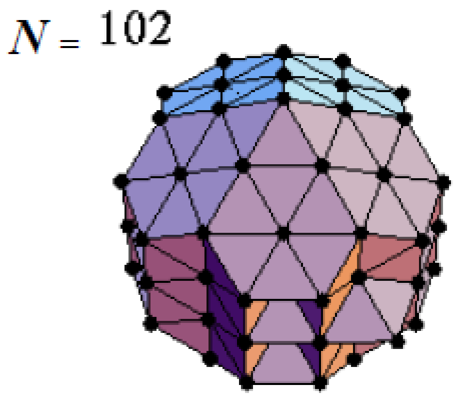
Problem: Find the geometric coordinates of a set of atoms which minimizes the internal energy of the system (there are different forms of the internal energy which are determined by the used potential function, e.g. Lennard-Jones)

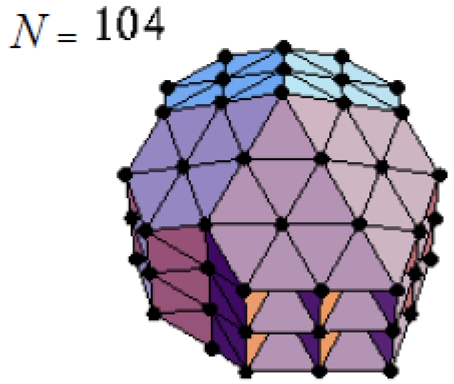
Solution representation: vector of real values containing triplets of atoms coordinates

Search space: $[a_1,b_1]x[a_2,b_2]x...x[a_n,b_n]$

(for instance: [0,4]x[0,4]x[-4,4]x...x[-4,4])

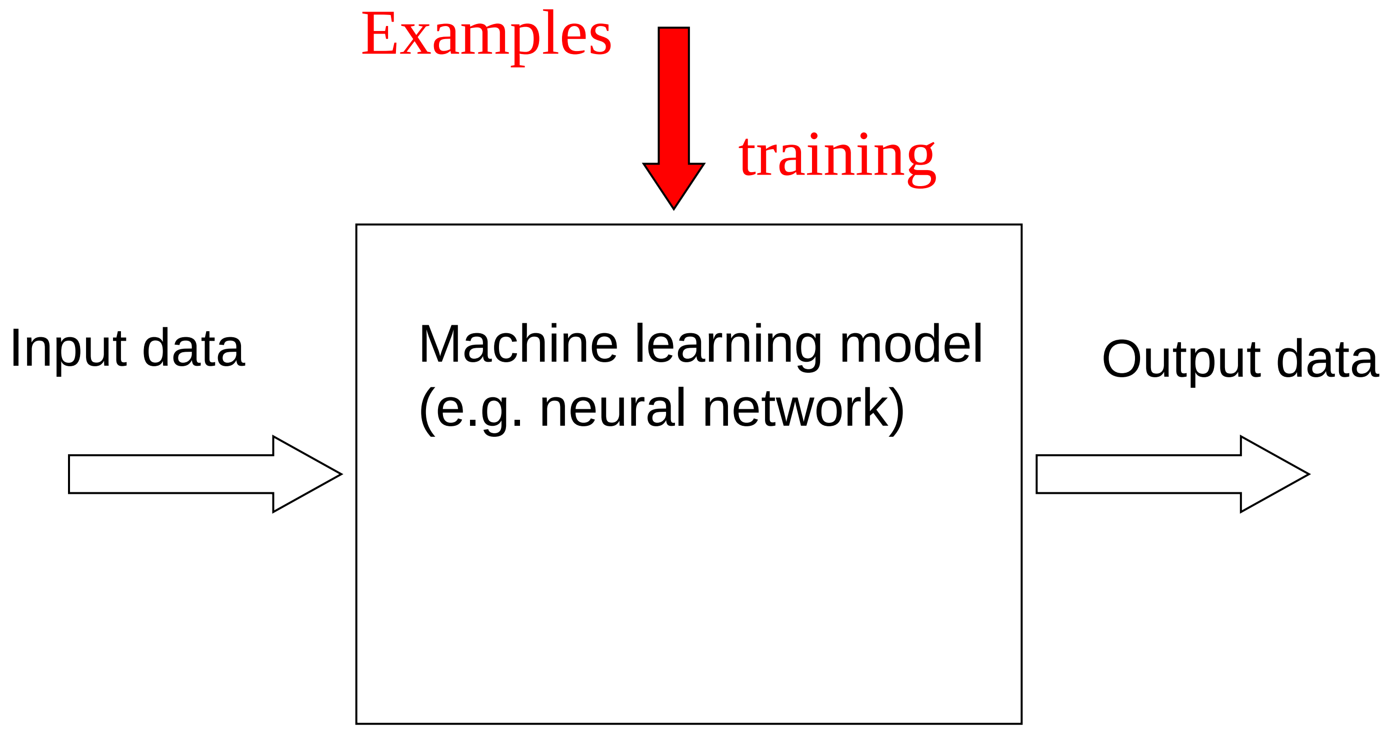
Challenge: it is difficult to find the global optimum (particularly for large values of the number of atoms)





Training adaptive systems

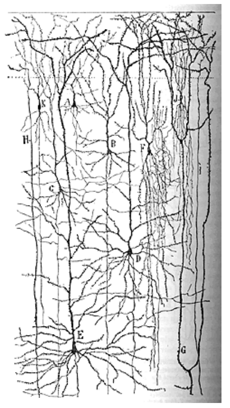
Machine learning approach

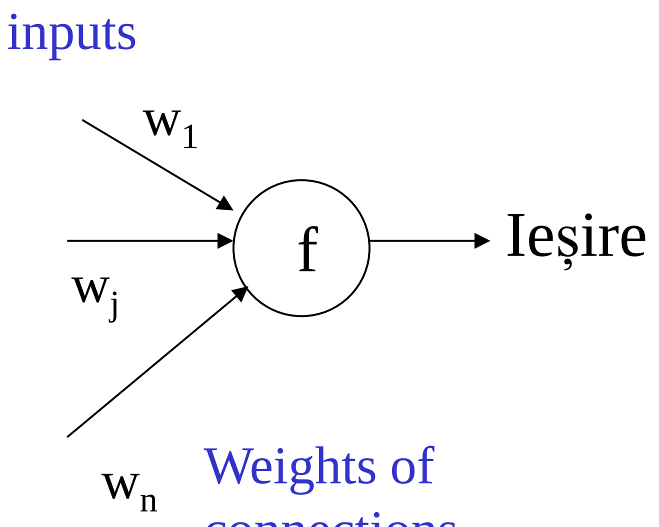


Training = estimate the adaptive parameters of the model

Artificial neural networks

ANN= set of interconnected artificial neurons (functional units characterized by adaptive parameters – the weights)





Weights of connections

$$y = f\left(\sum_{j=1}^{n} w_j x_j - w_0\right)$$

$$u = \sum_{j=1}^{n} w_j x_j - w_0$$

Aggregation function

$$f(u) = \mathrm{sgn}(u) \qquad f(u) = \tanh(u)$$

$$f(u) = H(u) \qquad f(u) = \frac{1}{1+\exp(-u)}$$

Activation function (computation of the output signal)

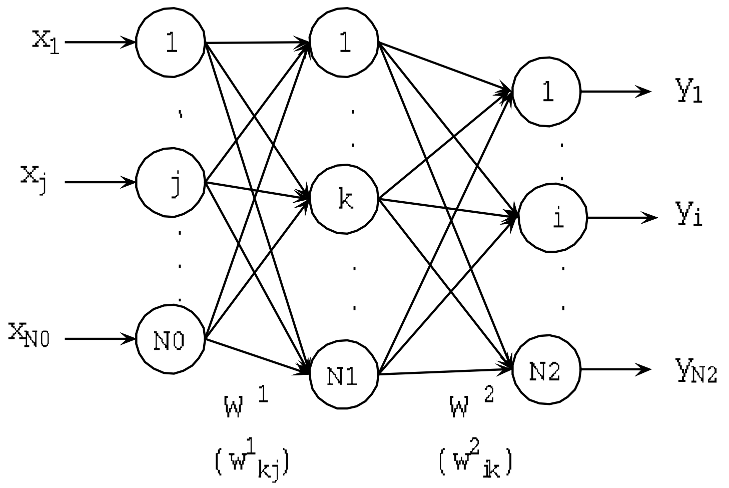
Neural network training

Structure of an artificial neural network:

- Architecture

- Functioning

- Supervised training: find the weights which minimizes an error function (difference the expected answer and that produced by the network)



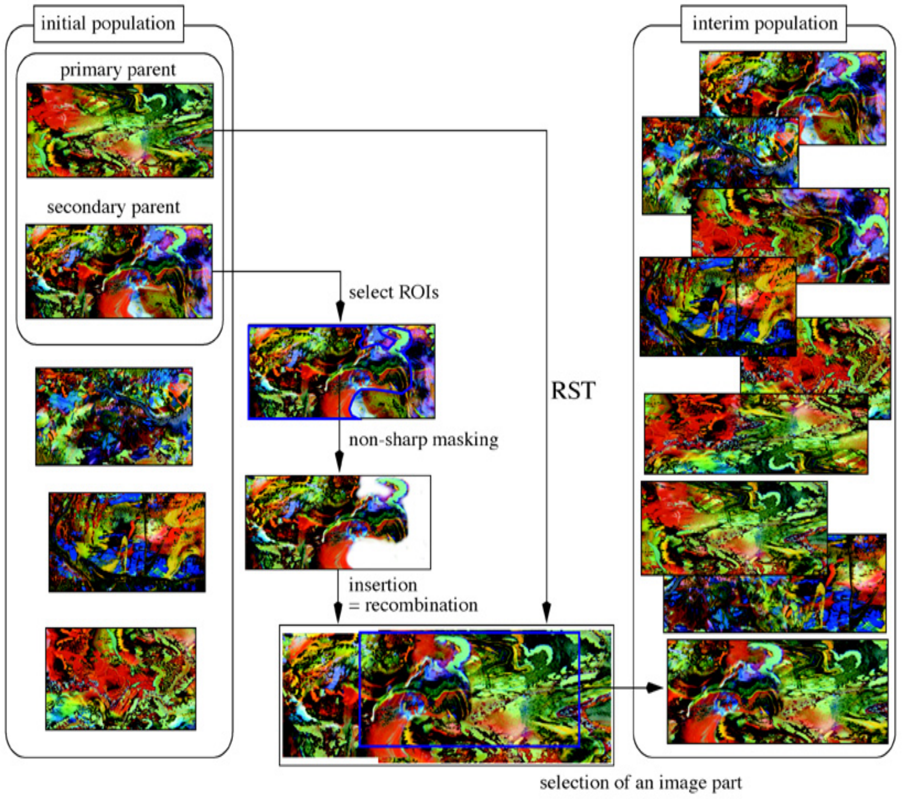
$$y_i = f\left(\sum_{k=0}^{N1} w_{ik}^2 \, f\left(\sum_{j=0}^{N0} w_{kj}^1 x_j\right)\right), \quad i = \overline{1, N2}$$

Feedforward neural network (used for classification/association problems)

Interactive optimization

- Design of various artistic structures (images, sounds) which optimizes a subjective criterion provided by a user

- DarwinTunes http://darwintunes.org/



[http://www.evogenio.com/de/GBEvoArt/EvoArt1.html]

Metaheuristic Algorithms

The metaheuristics are general search techniques which can be used to solve hard optimization problems. Their main characteristics are:

- They allow to identify some sub-optimal solutions
- They use a reasonable amount of computational resources

Particularities: they mainly rely on some stochastic operators

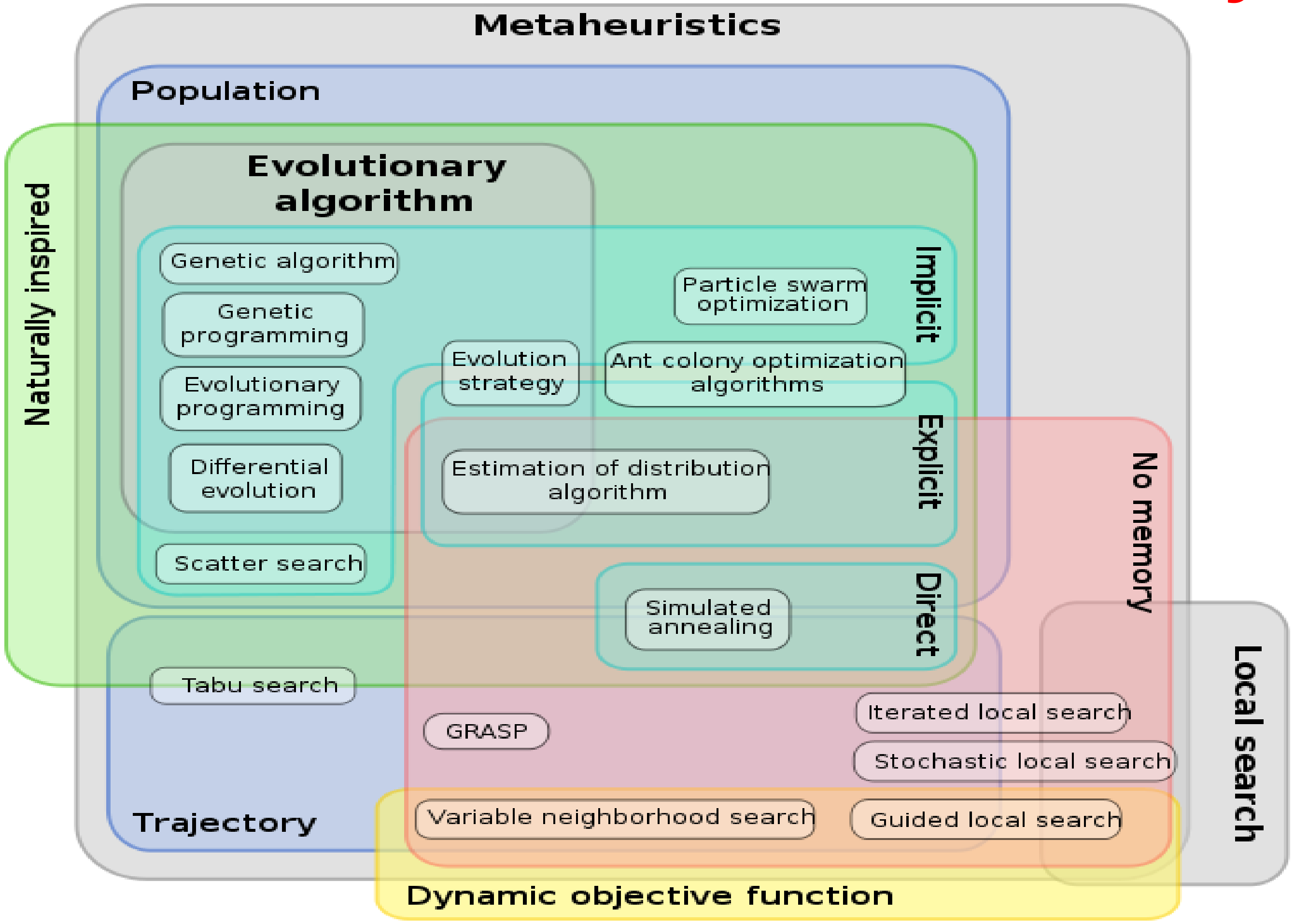
Advantages:

- They require only the values of the objective function (they do not have to be smooth or continuous)
- They are global optimization methods

Disadvantages:

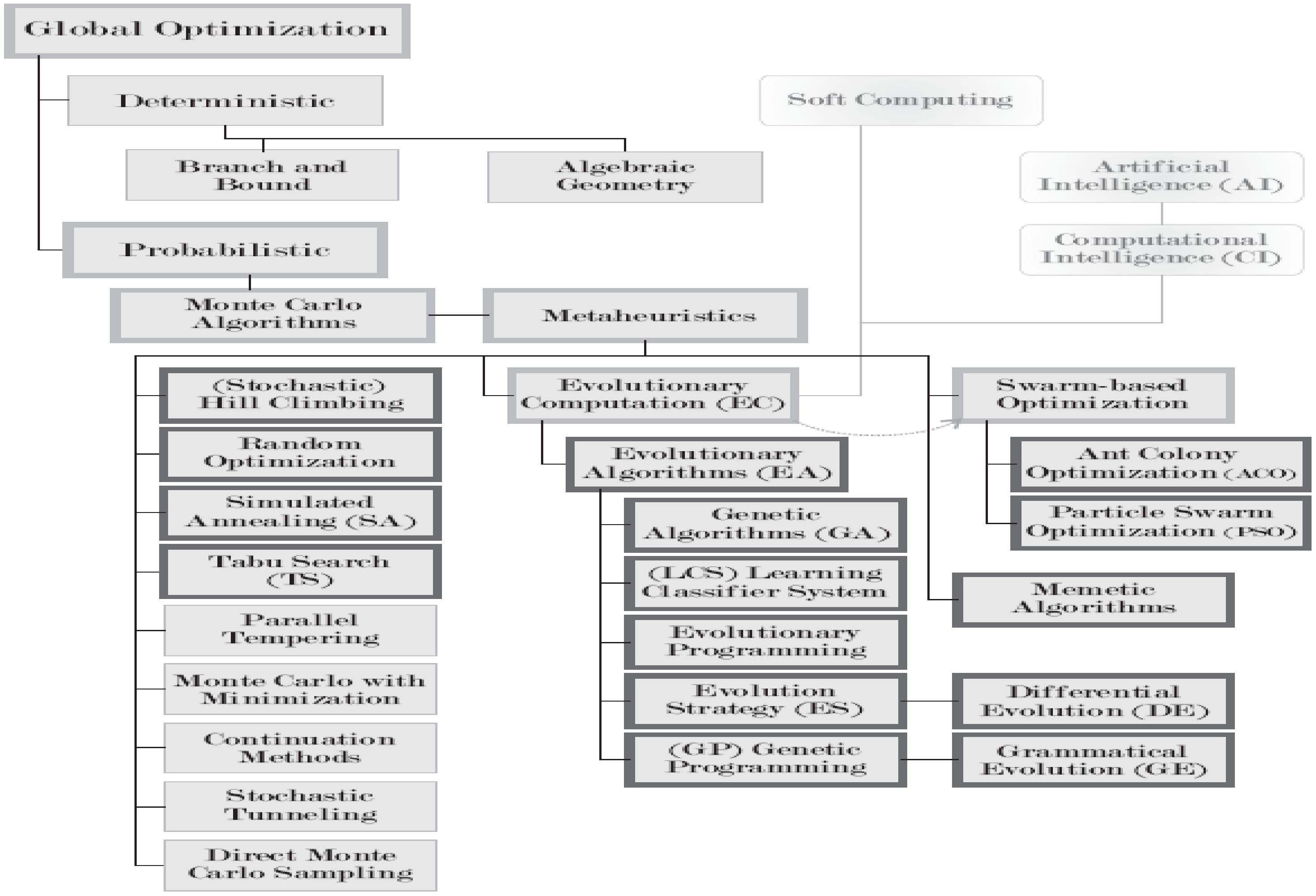
- There are no theoretical results concerning the convergence and the quality of the approximation

Metaheuristics Taxomomy



Metaheuristics Taxonomy [Wikipedia, nojhan.free.fr]

Metaheuristics Taxomomy



Structure of a generic metaheuristic

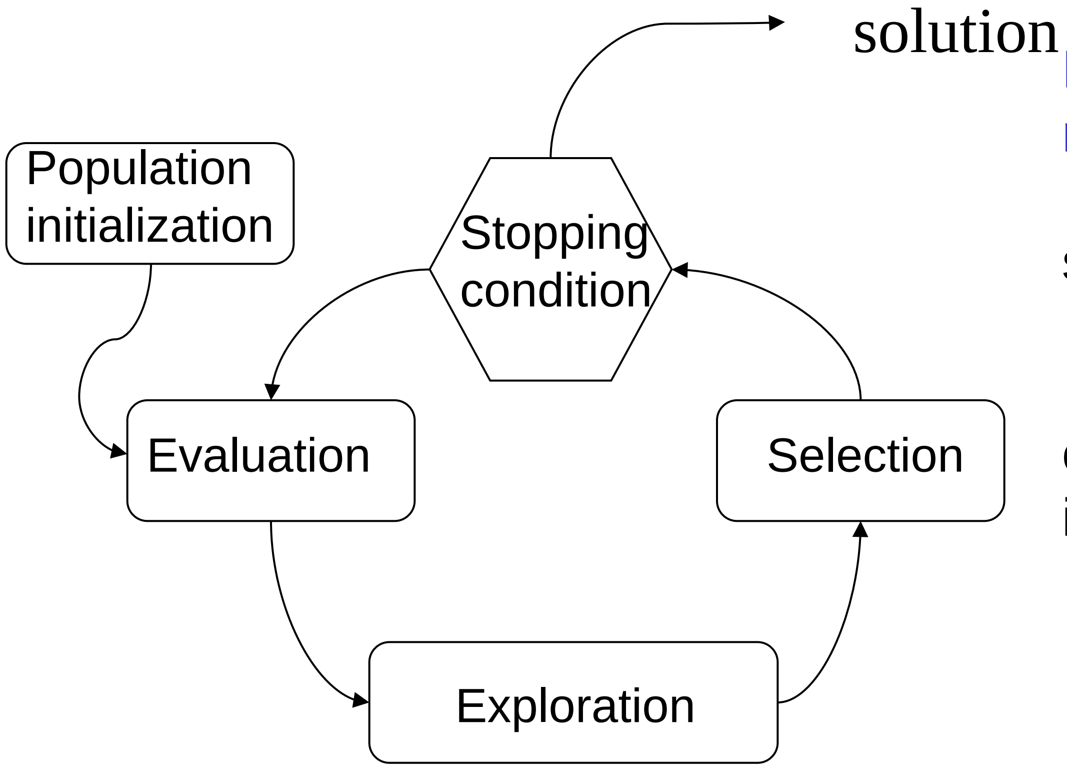
```
S=initial candidate solution
Repeat
    R=modify(S)
    if quality(R)>quality(S) then R=S
Until <stopping condition>
```

Components:

- Initialization
- Modification (ensures the solution space exploration)
- Selection (ensures the exploitation of good quality solutions)

Key element: ensure the balance between exploration and exploitation

Structure of an evolutionary algorithm



Population based metaheuristics= Iterative process based on successive application of
- exploration
- exploitation

operators on a randomly initialized population

Topics

- Metaeuristics which use a candidate solution (trajectory-based)
  - Hill climbing
  - Iterated Local Search
  - Simulated Annealing
  - Tabu Search
  - Greedy Randomized Adaptive Search
  - Variable Neighborhood Search
- Metaeuristics which use a population of solutions (population-based)
  - Evolutionary algorithms:
    - ES – evolution strategies,
    - EP- evolutionary programming
    - GA – genetic algorithms
    - GP – genetic programming
  - Differential Evolution
  - Swarm intelligence algorithms:
    - PSO - Particle Swarm Optimization,
    - ACO - Ant Colony Optimization
    - ABC - Artificial Bee Colony etc
  - Algorithms based on estimation of the probability distribution:
    - PBIL – Population Based Incremental Learning
    - EDA – Estimation of Distribution Algorithms

Structura curs

- Scalable metaheuristic algorithms
  - Cooperative coevolution
  - Parallelization models:
    - Master-slave
    - Island model
    - Cellular model
- Algorithms for some classes of optimization problems
  - Constrained optimization
  - multi-objective optimization
  - Multimodal optimization
  - Dynamic optimization
- Applications
  - Evolutionary design of neural networks
  - Data mining: feature selection, rules mining, clustering
  - Planning: routing, scheduling and resource allocation

References

1. Sean Luke, 2013, Essentials of Metaheuristics, Lulu, second edition, accesibilă la http://cs.gmu.edu/~sean/book/metaheuristics/
2. Z. Michalewicz, D. Fogel: How to Solve It. Modern Heuristics. Springer, 1999

Lab

Lab 1: Introduction in SciLab and simple optimization problems

Lab 2: Random values in SciLab. Simulated Annealing, Tabu Search etc. Combinatorial optimization (TSP)

Lab 3: Evolutionary Algorithms. Continuous optimization

Lab 4: Swarm intelligence algorithms (PSO, ACO)

Lab 5: Multi-objective optimization - MOEA

Lab 6: Evolutionary design of neural networks

Lab 7: Data mining and planning applications

Tools:

SciLab, R

Evaluation

Course materials: http://www.info.uvt.ro/~dzaharie/ma2015

Evaluation:

Final project: 60-80%

Written test: 20%

Homework: 20%