Evolutionary Computation

2023/24

Master Artificial Intelligence

Arno Formella

Departamento de Informática
Escola Superior de Enxeñaría Informática
Universidade de Vigo

23/24

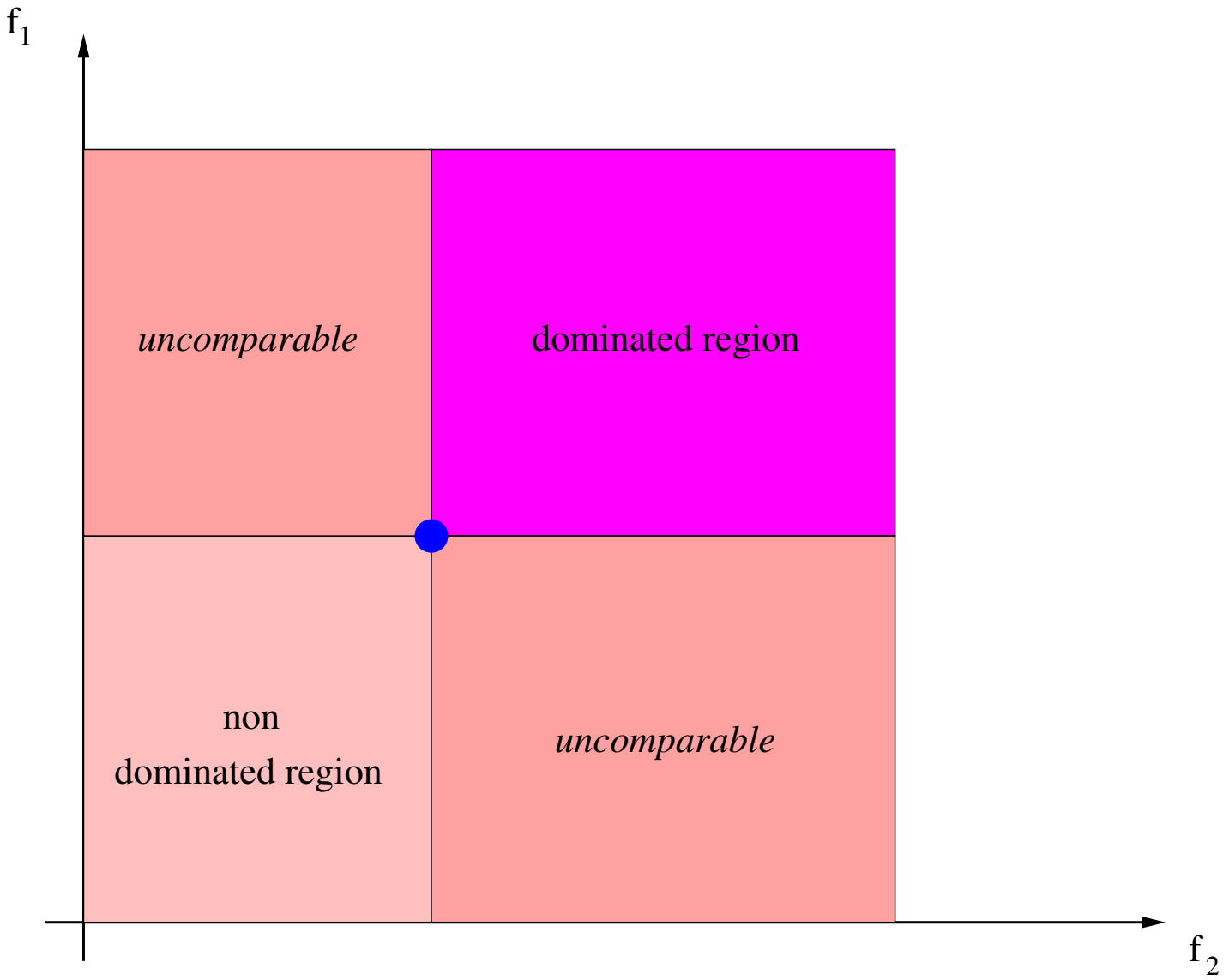
multi-objective optimization

- Given a search space $\mathbb{X}$ (called as well *search domain* or *problem space*) and
- a set of k functions f_i (bounded from below) from the search space to the real numbers (or at least a totally ordered set), e.g. $f_i : \mathbb{X} \longrightarrow \mathbb{R}$,
- find an element $x^\star \in \mathbb{X}$ such that $f_i(x^\star) \leq f_i(x)$ for all $x \in \mathbb{X}$ and all k functions f_i.
- Maybe there is no point in $\mathbb{X}$ that minimizes all the k functions simultaneously, then we look for Pareto-optimal solutions, i.e., solutions that cannot be improved without worsening at least one of the other objective functions.

Pareto front

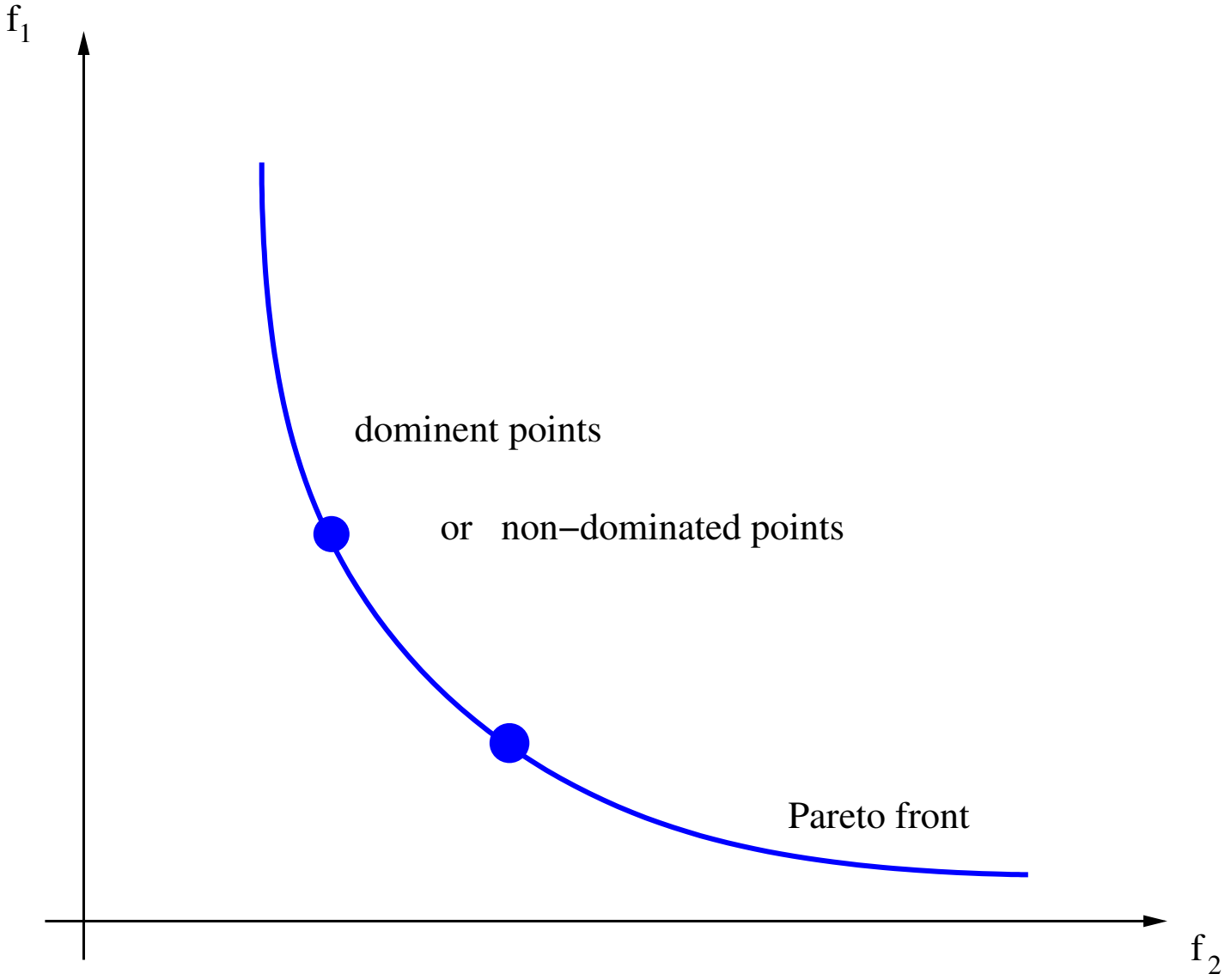
- Remind we have more than one independent objective function
- Pareto optimal (global): every other component for all other solutions is worse (or equal)
  (other names are: efficient points, dominant points, non-interior points)
- Pareto optimal (local): every other component for all other solutions in a local neighborhood is worse (or equal)
- Hence, the Pareto frontier describes the trade-off between the different objectives.

Dominant points and regions



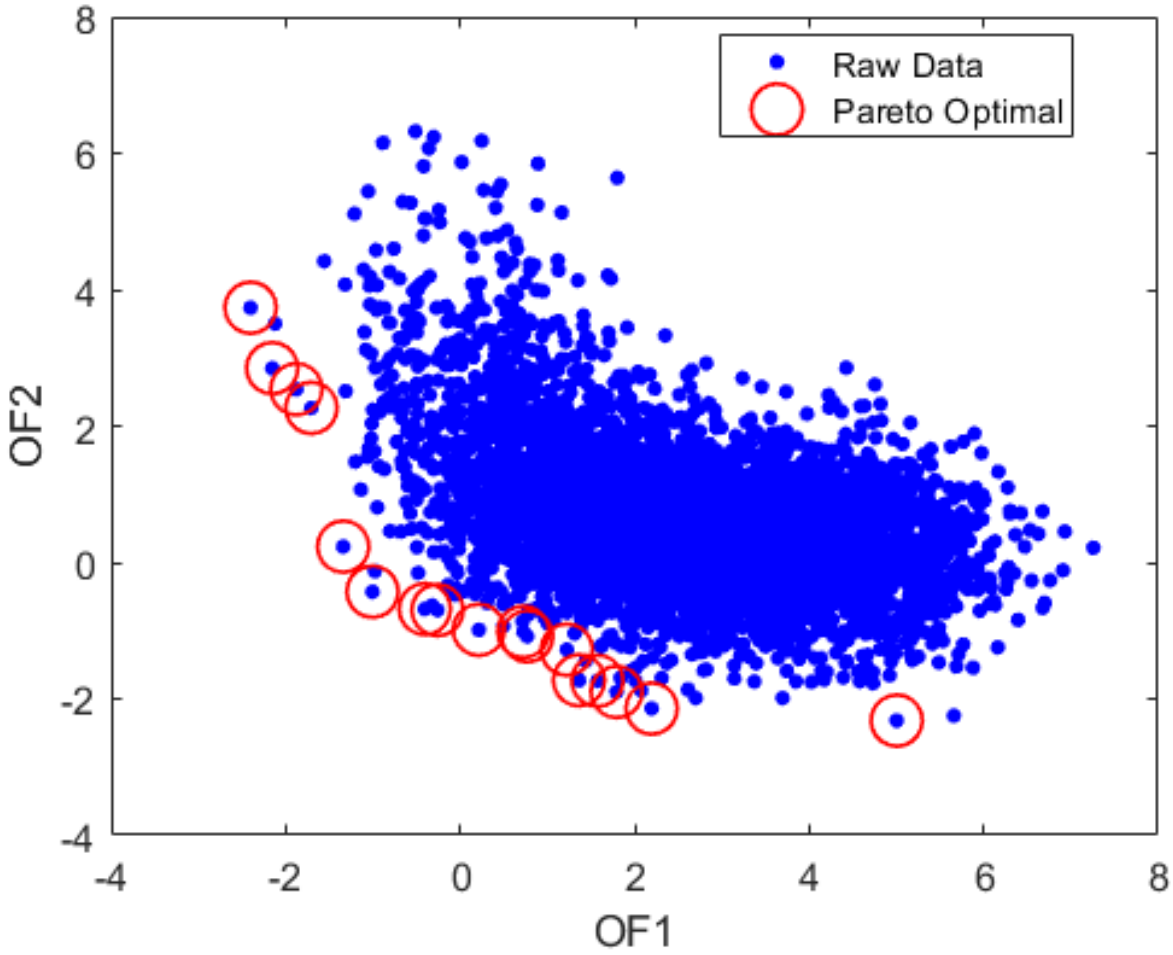
Pareto front

Example of a Pareto front with two marked non-dominated points:



Pareto front

Example of a Pareto front for two objectives (on measured data):



Multi–objective optimization

How can we find a *good* solution to the optimization problem with more than one objective function?

- convex combination of the objectives
- homotopic techniques, i.e., compute the entire Pareto frontier, for instance with a population based algorithm, and select later... (to obtain the Pareto frontier one might explore the coefficient space of the convex combination)
- goal programming, i.e., fixed values for all objectives and minimize the distance of all objectives to the predefined goals (according to some convenient distance metric)

Multi–objective optimization

- priority optimization, i.e., fix thresholds for all but one objective function beforehand and optimize above the threshold according to the most important one
- priorization (multi-level) programming, i.e., optimize according to a predefined ordering of the objective functions.
- fixed trade-off, i.e., find the point in the Pareto front that is tangent to a certain hyperplane (especially usefull when Pareto front is convex and low dimensional).

Optimization with restrictions (or constraints)

In many application there are restrictions (or constraints) that limit the optimization process:

- find an element $x^\star \in \mathbb{X}$ such that $f_i(x^\star) \leq f_i(x)$ for all $x \in \mathbb{X}$ and all k functions f_i and
- a certain number L of inequality constraints $g_j(x) \geq 0$ (for all $j \in [1 : L]$) are fulfilled, and
- a certain number E of equality constraints $h_n(x) = 0$ (for all $n \in [1 : N]$) are fulfilled.

Simple constraints are for instance so-called box-constraints, i.e., the search space is confined in each dimension by an interval.

Such box constraints are often handled separately in the optimization packages.

Optimization with restrictions (or constraints)

- The inequality and equality constraints again might be linear or non-linear functions.
- Due to the restrictions there might arise points (elements of the search space) during the optimization process which are unfeasible, i.e., no valid objective function values can be computed (or even the objective function cannot be computed at all).
- Sometimes even trying to find some feasible solution is already a very complex task (for instance: for TSP with time windows the problem is already NP-hard).

Optimization with restrictions (or constraints)

To tackle constraints there are two main classical approaches:

- use of penalties, i.e., assign a *sufficiently large* values to the objective function(s)
- use of interior methods, i.e., make sure not to leave the feasible region
  (main idea: use the fulfillment of the constraints as additional objective function building a so-called *barrier function* with an additional parameter μ that is continuously shrinked to reach zero.)

Multi–objective optimization with evolutionary methods

- Evolutionary methods can approximate the Pareto frontier in parallel (with the help of the diversity among the individuals).
- For instance particle swarm systems varying the weights of a convex combination periodically during the iterations.
- For instance a genetic algorithm can hold a population that tries to converge (in some sense uniformly) towards the Pareto front.

multi-objective evolutionary algorithms (MOEA)

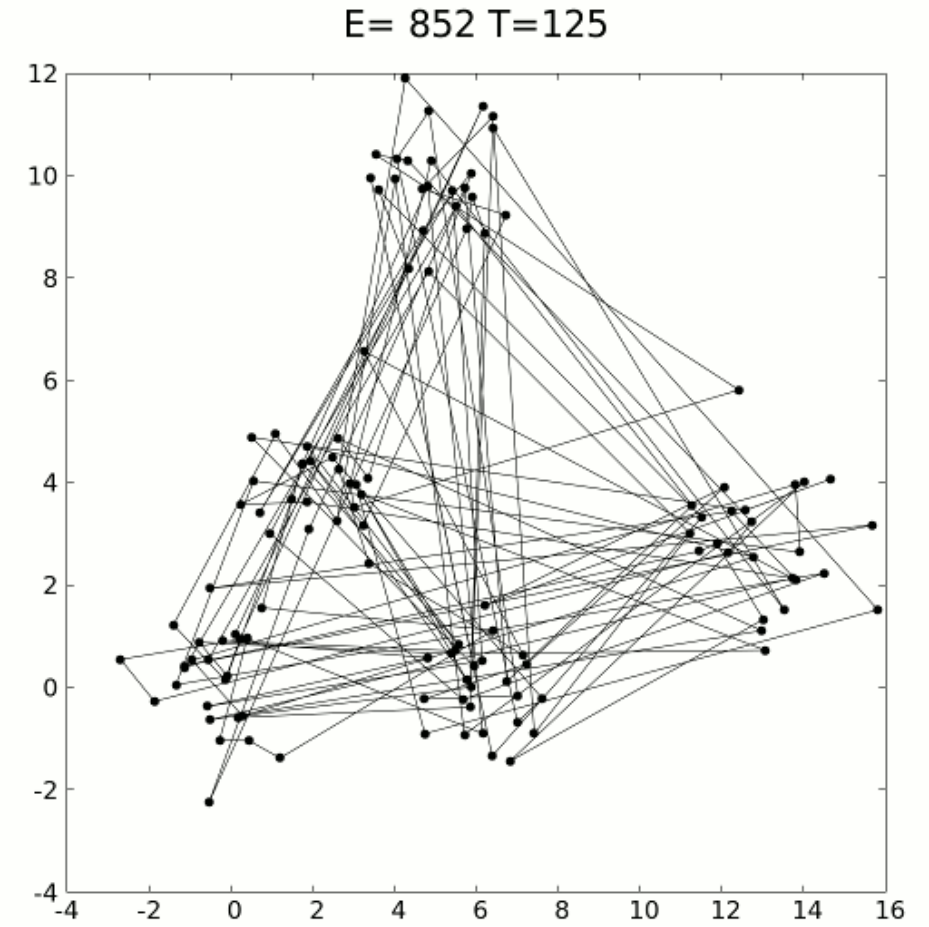
- VEGA, vector evaluated genetic algorithm
  Idea: in the selection process parts of the mating parents are selected according to each objective function
- MSGA, multi-sexual genetic algorithm
  Idea: individuals are marked as belonging to a certain objective function, ranking is used to select parents, only differently marked individuals are used to generate children
- NSGA, non-dominant sorting genetic algorithm
  Idea: sort individuals according to their dominance, and design the selection according the dominance classes (i.e., work with several frontiers, intending to converge eventually to the Pareto front.

multi-objective evolutionary algorithms (MOEA)

- NSGA-II, including elitism, dominant individuals are preserved in population, clustering is avoided
- SPEA, strength Pareto Evolutionary algorithm
  Idea: maintain a fixed set of best individuals while guaranteeing that they are spread over the Pareto front without to much clustering

Simulated Annealing

The name and idea stems from the slow and repeated heating and cooling process of certain materials (usually alloys of different metals with addings, e.g., iron+carbon) until certain properties are achieved.



https://en.wikipedia.org/wiki/Simulated_annealing

Simulated Annealing

- Explores a neighbor in the search space, whenever there is an improvement in the objective function at the neighbor or it is not worse than a temperature dependent threshold with some probability.
- Let Δf be the difference between current solution f and neighbor solution f_n.
- Let T be a temperature.
- Then the neighbor is accepted whenever either $\Delta f < 0$
  (we are going downhill)
  or if $e^{-\Delta f/T} > r$, for r being a random value in $[0 : 1]$
  (we are going uphill).
- With increasing iteration rounds, i.e., during a certain number of iterations the temperature is held constant, the temperature is reduced, hence, the threshold converges towards zero.

Simulated Annealing cooling schemes

- linear cooling: $T = T_0 - \eta k$
  with T_0 an initial temperature, k the iteration number and η some free parameter, being constant during optimization.
  (To avoid negative temperature: $T = \max(T_0 - \eta k, T_{\min})$)
- exponential cooling: $T = aT$
  with $a \in [0.8, 1)$ typically; slower cooling the a close to 1.
- inverse cooling: $T = T/(1 + \beta T)$
  with β being a small constant (e.g. $\beta = 0.001$).
- logarithmic cooling: $T = T_0/\log k$
  with c a suitable constant.
  (Used as well a generalization: $T = T_0/\log(k + d)$).
  Not really practical in applications but was used to prove convergence to global minimum under certain conditions.
- inverse linear cooling: $T = T_0/k$.
  Not really practical in applications but was used to prove convergence to global minimum under certain conditions.