MULTI-OBJECTIVE OPTIMIZATION

Luca Manzoni

SINGLE VS MULTI-OBJECTIVE

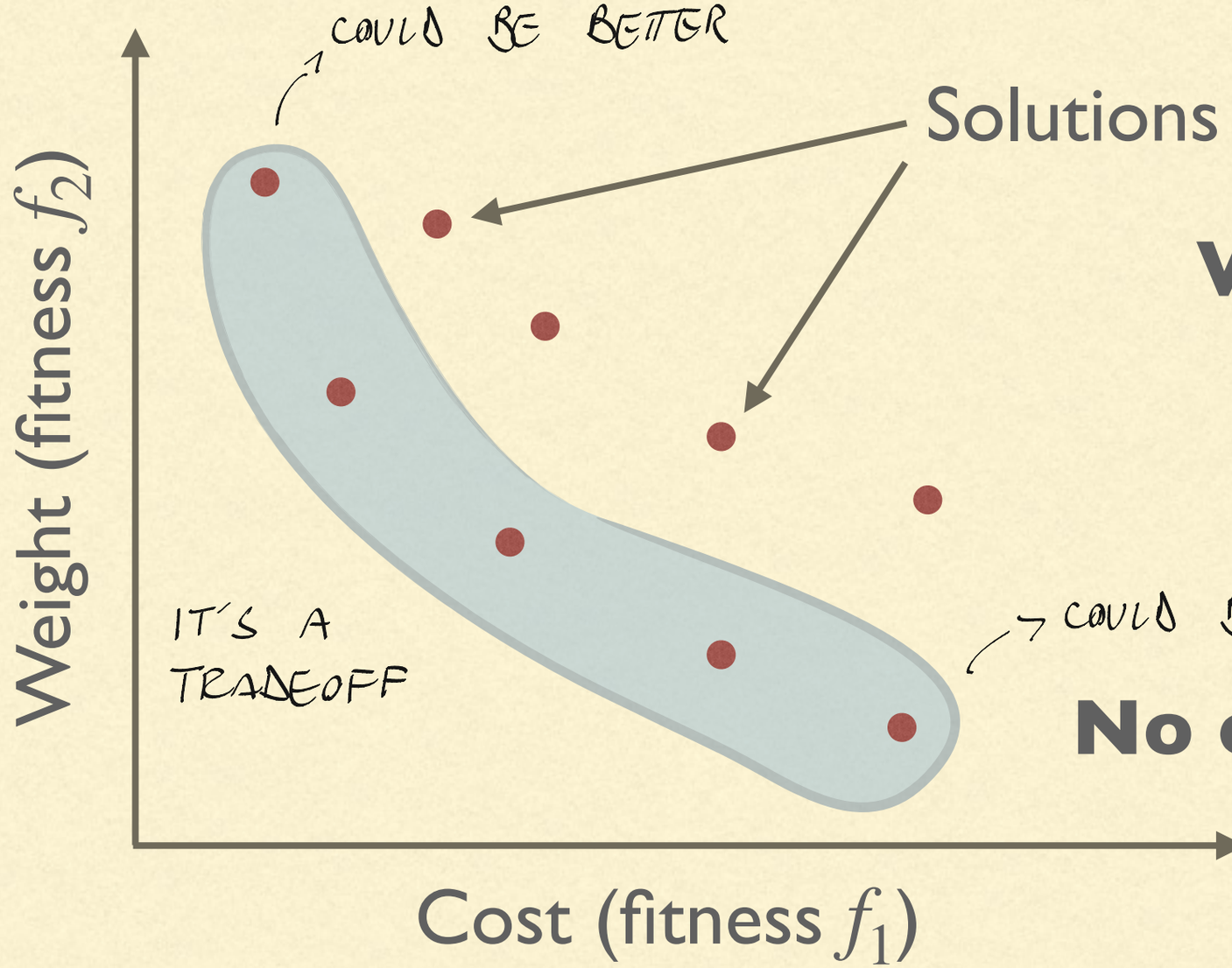
- Classic global optimization: single objective/criterion:
  - Minimize or maximize a given fitness function $f : X \to \mathbb{R}$
    - GA, DE, PSO, etc. can solve this class of problems
- Sometimes, problems have **multiple objectives**
  - Multiple fitness functions $f_1, f_2, \ldots, f_k$ to be simultaneously optimized
  - Why can't we simply combine all fitness functions in a single one?

SINGLE VS MULTI-OBJECTIVE

- These functions can be conflicting:
  - "Optimize costs" vs "Optimize weight/performance/etc"
  - "Fastest route" vs "Cheapest route"
- A trivial combination of the fitness values does not work since it generally leads to the overfitting of a subset of the criteria
- there might be infinite optimal (and perfectly valid) solutions: the **dominating solutions**

DOMINATING SOLUTIONS

A SOLUTION IS SAID TO BE DOMINATED IF EXIST A BETTER SOLUTION

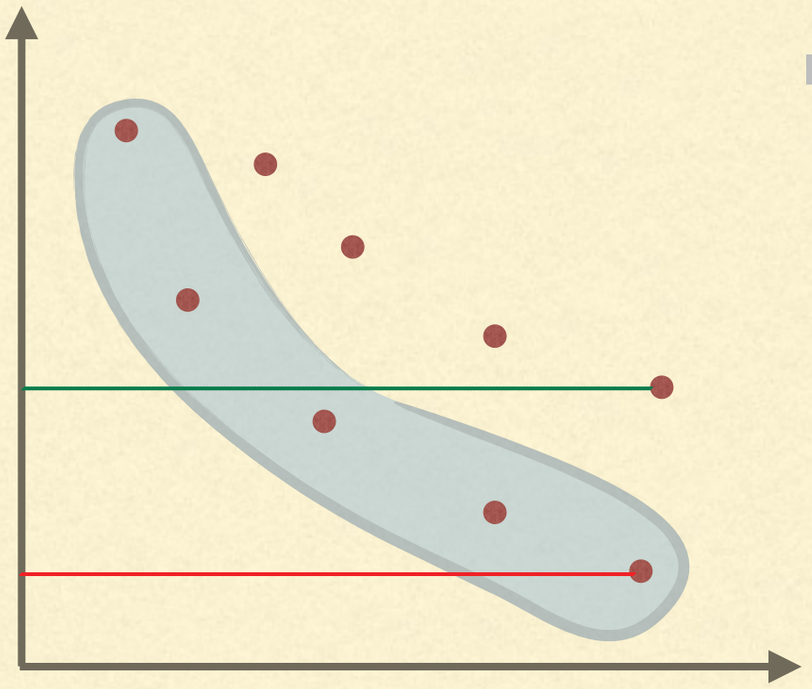


Which solution is better?

THE USER DECIDE

No clear winner

DOMINATING SOLUTIONS



- A solution x is said to **dominate** a solution y when:
  - $f_i(x) \leq f_i(y)$ for all $i \in \{1,\ldots,k\}$
  - $f_i(x) < f_i(y)$ for at least one $i \in \{1,\ldots,k\}$
  - Dominance is denoted by $x \prec y$

The set of non-dominated solutions is called the **Pareto** front

PARETO FRONT AND OPTIMIZATION

- In multi-objective optimization we are not interested in one solution
- We look at the Pareto front of the solutions
- The Pareto front may contain infinite elements…
- …but, as usual a good approximation (a "representative" set of solutions on the Pareto front) is sufficient

LACK OF DIVERSITY
BEFORE: CONVERGENCY TO SUB-OPTIMAL SOLUTION
NOW: EVEN WORSE, YOU ARE NOT PROVIDING A GOOD APROXIMATION OF SOLUTIONS FRONT

PARETO FRONT AND OPTIMIZATION

- We can use evolutionary computations or swarm intelligence methods:
  - The population converges to the Pareto front (i.e., all or most of the individuals are the "best" solution)
  - Loss of diversity is a issue: the population might converge to a single point!

    INSTEAD, WE WANT A SET OF OPTIMAL SOLUTION
  - Any multi-objective algorithm must prevent convergence to a single point and must keep the solutions "spread out" on the Pareto front

WRONG APPROACHES

↳ IF YOU TRY THEM, THEY WILL NOT WORK

- Wrong approach #1: **Objectives weighting**
  - Define $f(x) = \sum_{i=1}^{k} w_i f_i(x)$ where we select the weights $w_1, \ldots, w_k$ such that $0 \leq w_i \leq 1$ and $\sum_{i=1}^{k} w_i = 1$ → NOT NECESSARY

DIFFERENT WEIGHTS FOR DIFFERENT OBJECTIVE

- How can we select the weights?

- We are still converging to a particular point on the Pareto set

BECAUSE WE TRY TO OPTIMIZE THE SUM

WE CAN TRY TO KEEP TRACK, BUT THE EVOLUTION WILL STILL CONVERGE

WRONG APPROACHES

- Wrong approach #2: **Method of distance function**
  - Let $f(x) = \left(\sum_{i=1}^{k} |f_i(x) - y_i|^r \right)^{\frac{1}{r}}$ with $r > 0$ (usually $r = 2$) and $y = (y_1, \ldots, y_k)$ is the vector of optimal fitness values
  - The vector y might be unknown
  - The vector of optima strongly influence the convergence to a solution

WHY DOES THIS DOESN'T WORK? MAYBE YOU DON'T KNOW THE VECTOR

WRONG APPROACHES

- Wrong approach #3: **Vector Evaluated Genetic Algorithm**

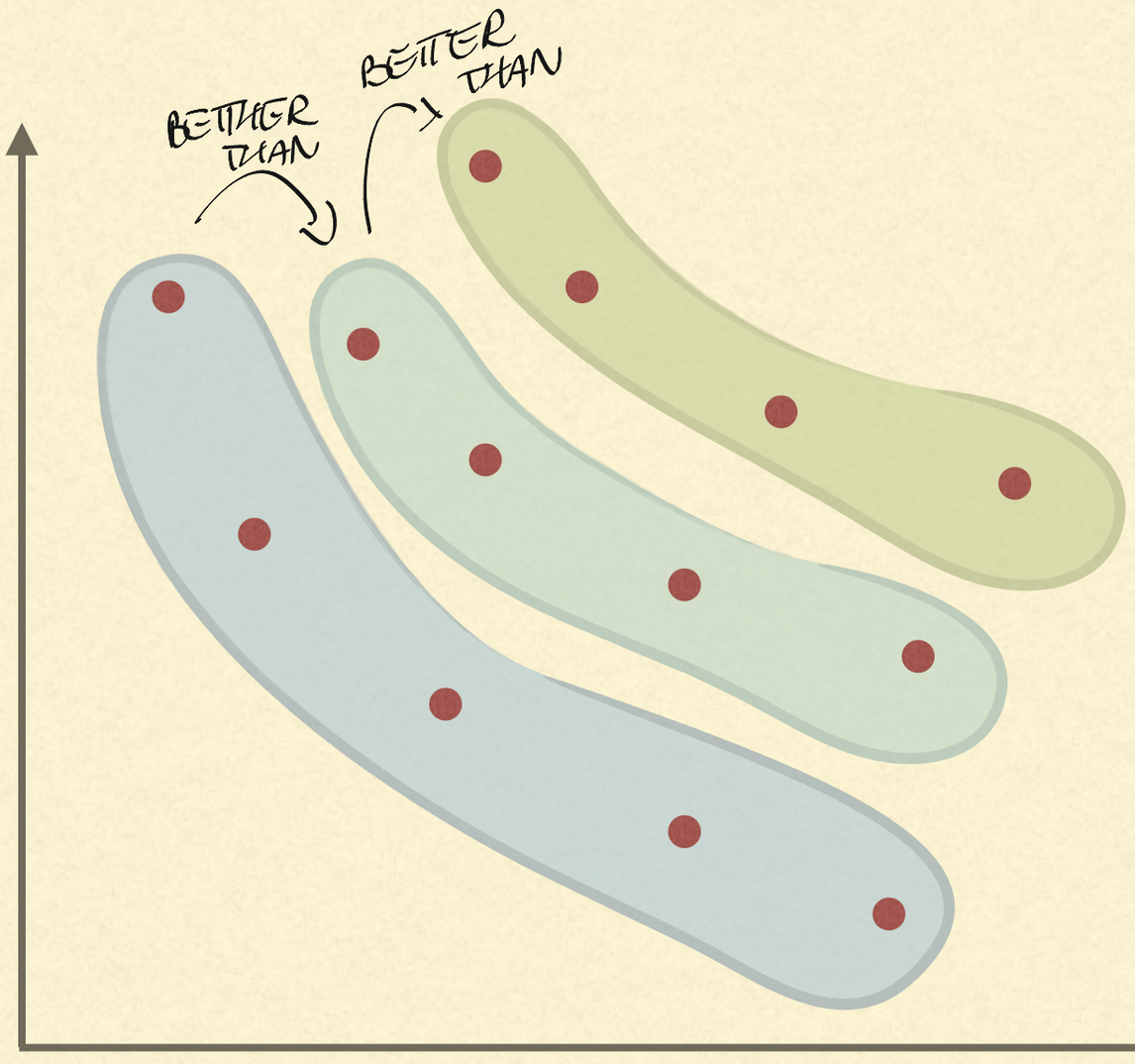
PREDECESSOR OF EVOLUTION ALGORITHM

  - Perform multiple rounds of selection, one for each objective
  - The new population is made of individuals that are selected using different objectives
  - There is a problem of speciation: the population divides into multiple sub-populations good for only one objective!

WHAT WAS THE PROBLEM?

- In all cases we used a single objective:
  - Either we created a single objective from multiple objective functions
  - Or we used only one for the selection process
- To actually return a Pareto set of solutions we need to keep track of multiple objectives in the design of the algorithm!

NON-DOMINATED SORTING

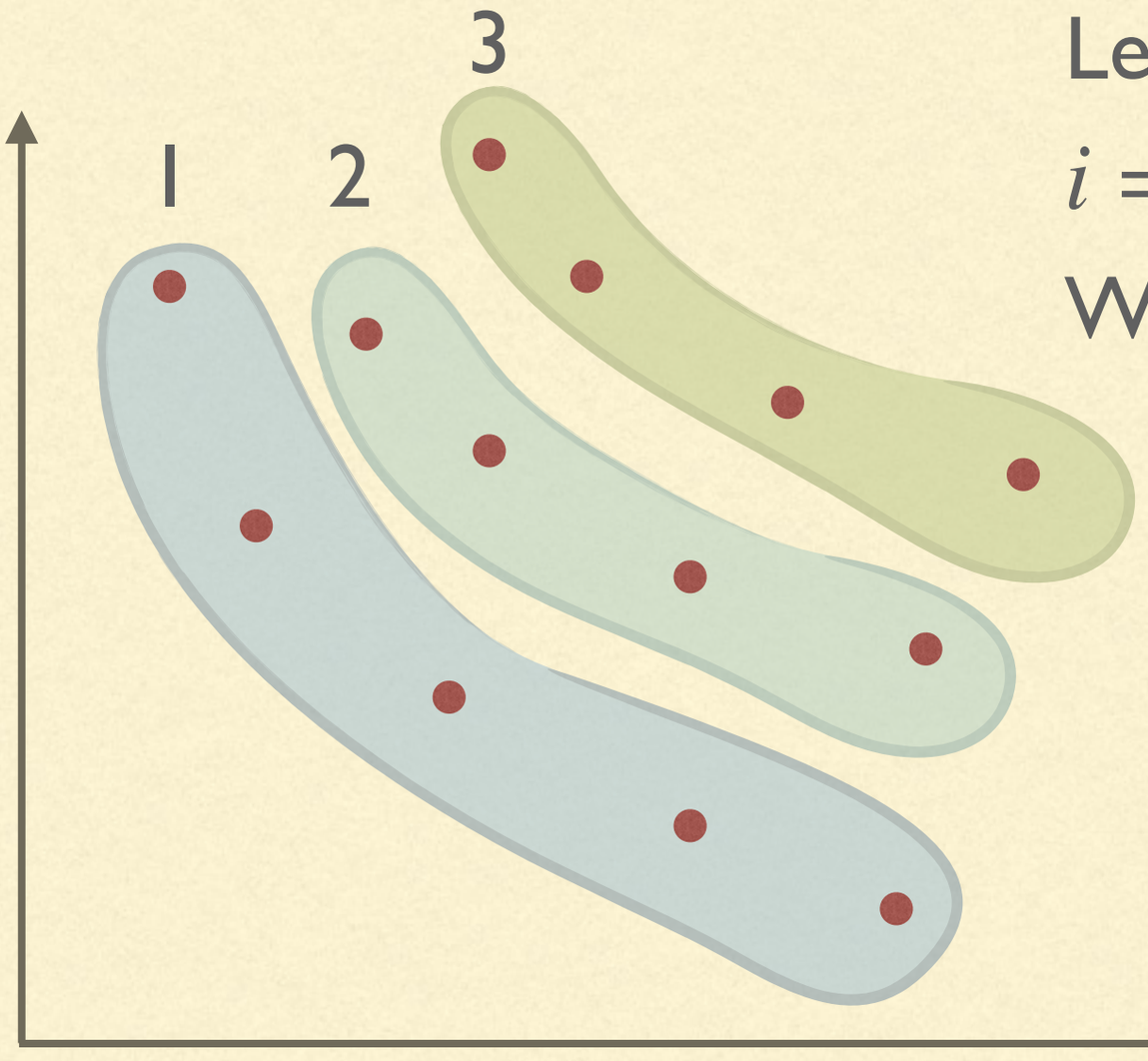


Pareto-dominance induces a partial ordering of solutions

All solutions in the Pareto front have the same rank

We can sort multiple fronts according to their "dominance" level

NON-DOMINATED SORTING



Let P be the current population

$i = 1$

While $P \neq \emptyset$

$O(n^3$ - BE REDUCE TO $O(n^2)$

Select $\mathcal{O}$ as the Pareto front

Assign rank i to all elements of $\mathcal{O}$

$P = P - \mathcal{O}$

$i = i + 1$

I STILL NEED TO FIGURE HOW TO UNDERSTAND IF TWO SOLUTION HAVE THE SAME RANK

NSGA

- Non-dominated Sorting GA (NSGA) works similar to GA except that the fitness is computed differently using non-dominated sorting and sharing

- A **dummy fitness**, proportional to front number, is assigned to the solutions

- Sharing is used to promote diverse solutions

NSGA: SHARING

- Let $d(x, y)$ be a phenotypic distance between x and y
- Then the sharing is defined by

PENALTY

$$Sh(x,y) = \begin{cases} 1 - \left(\dfrac{d(x,y)}{\sigma_{\text{share}}} \right)^2 & \text{if } d(x,y) \leq \sigma_{\text{share}} \\ 0 & \text{otherwise} \end{cases}$$

- σ_{share} is the maximum phenotypic distance allowed between any two individuals to become members of a niche

IT'S LIKE CLUSTERING POINTS ON A SPACE

NSGA-II

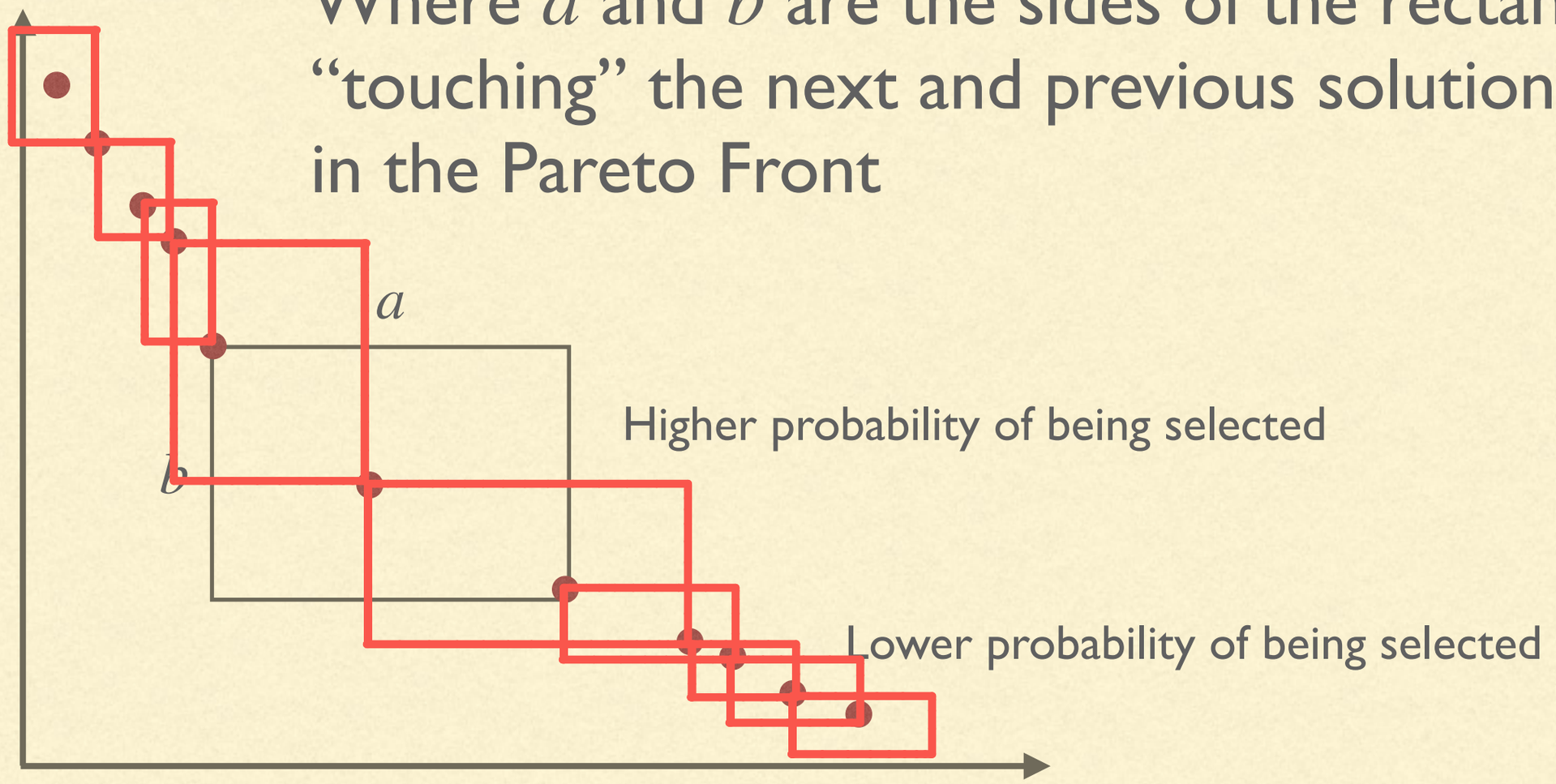
- Evolution of NSGA, it adds:
  - Elitism (KEEP THE RANK 1 SOLUTIONS)
  - Fast non-dominated sorting to rank solutions (to reduce the time needed to compute the Pareto fronts)
  - Crowding distance (CD) to maintain diversity in the population
    → I USE 2 METHODS
  - Tournament selection where the best rank wins (if two solutions have the same rank then CD is used)

CROWDING DISTANCE

- To prevent the loss of diversity we need a "measure" of diversity
- Diversity in objective space (not between genotypes)
- Crowding Distance (CD) is one example of this measure
- CD selection: less crowded solutions (belonging to the same front) have a higher possibility of being selected for the next generation

CROWDING DISTANCE

Crowding distance for x: $a + b$
Where a and b are the sides of the rectangle "touching" the next and previous solution in the Pareto Front



WE MAKE SURE TO SPREAD THE SOLUTIONS

CROWDING DISTANCE

- The effect of using the crowding distance is to spread out the solutions uniformly on the Pareto front

- So recap:
  - Better rank wins
  - If there is a tie for better rank then the solution in the less crowded part of the Pareto front wins

NSGA-III

- A variant of NSGA-II to work better on more than two objectives
- Uses a set of "reference points" to preserve diversity on the Pareto front
- Another variant is R-NSGA-III where the set of reference points is based on a set of user-defined "aspiration points"

MOPSO

- Multi-objective PSO, so swarm intelligence instead of evolutionary algorithms

- **Main idea**: change the particles' velocity update wrt classic PSO to consider an attraction "towards the Pareto front", i.e., towards the dominating particles found so far

- **Secondary idea**: maintain a list of the hyper-cubes that keep track of the dominating positions visited so far

MOPSO: REPOSITORY

- In MOPSO, a global repository of non-dominated particles REP is maintained
- When a new dominating solution is added to REP, the dominated particles are removed
- The repository has a limited size Q
- Then the number of particles in the repository is greater than Q, the particles belonging to most crowded hyper-volumes are removed until the repository contains exactly Q particles

MOPSO: UPDATE

- MOPSO's velocity update formula becomes
  $v_i(t+1) = w \cdot v_i(t) + c_{soc} \cdot r_1 \otimes (REP_h - x_i(t)) + c_{cog} \cdot r_2 \otimes (p_i - x_i(t))$

- REP_h is the h-th particle of the repository

- The index h is determined using a tournament selection over the particles in the repository

- Fitness sharing is used to reduce the fitness values of particles belonging to crowded hyper-cubes

- The personal best p_i now corresponds to the best position visited so far by the i-th particle wrt all objectives

MOPSO: HYPERPARAMETERS

- MOPSO has several hyper-parameters to be tuned
  - **The size of the repository**
  - The size of the tournament
  - **The size of hyper-cubes** (i.e., the number of divisions)
  - The size of the swarm
  - The inertia factor
  - The social and cognitive factors
  - The maximum (and minimum) velocity
  - The number of iterations

AVAILABLE LIBRARIES

- The main library (in Python) for working with multi-objective optimization is Platypus

- https://github.com/Project-Platypus/Platypus

```python
from platypus import NSGAII, Problem, Real

def schaffer(x):
    return [x[0]**2, (x[0]-2)**2]

problem = Problem(1, 2)
problem.types[:] = Real(-10, 10)
problem.function = schaffer

algorithm = NSGAII(problem)
algorithm.run(10000)
```