COMP7630 – Web Intelligence and its Applications

Evolutionary Algorithms

Valentino Santucci

(valentino.santucci@unistrapg.it)

Outline

- <u>What are Evolutionary Algorithms?</u>
- Ingredients of an EA
- A simple Genetic Algorithm
- Current trend in Evolutionary Computation
- Analysis of EAs

Definition of Evolutionary Algorithm

«In the subfield of Artificial Intelligence known as "Evolutionary Computation", an Evolutionary Algorithm (EA) is a generic population-based metaheuristic optimization algorithm. Candidate solutions to the optimization problem play the role of individuals in a population, and the fitness function determines the quality of the solutions. Evolution of the population then takes place after the repeated application of operators inspired by biological evolution, such as: mutation, recombination, and selection.»

(from Wikipedia)

Why Evolutionary Algorithms?

«Traditional algorithms» are:

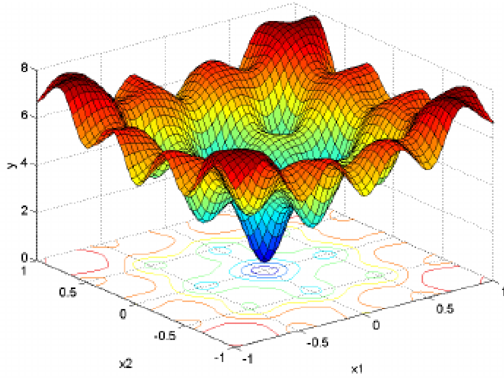
- tailored on a given problem ... but problems are innumerable
- exact ... but problems are NP-Hard
- have theoretical guarantees ... it is difficult to analyze EAs

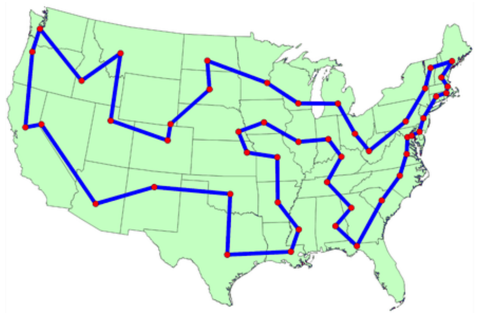
EAs are often easy to implement and provide «good enough» solutions in a reasonable amount of computational time.

EAs and the Black Box Model

- No knowledge about the problem is required
- EAs work on different problems (they are metaheuristics)
- «Generate and test» paradigm
- EAs navigate the space of solutions
- EAs are stochastic algorithms

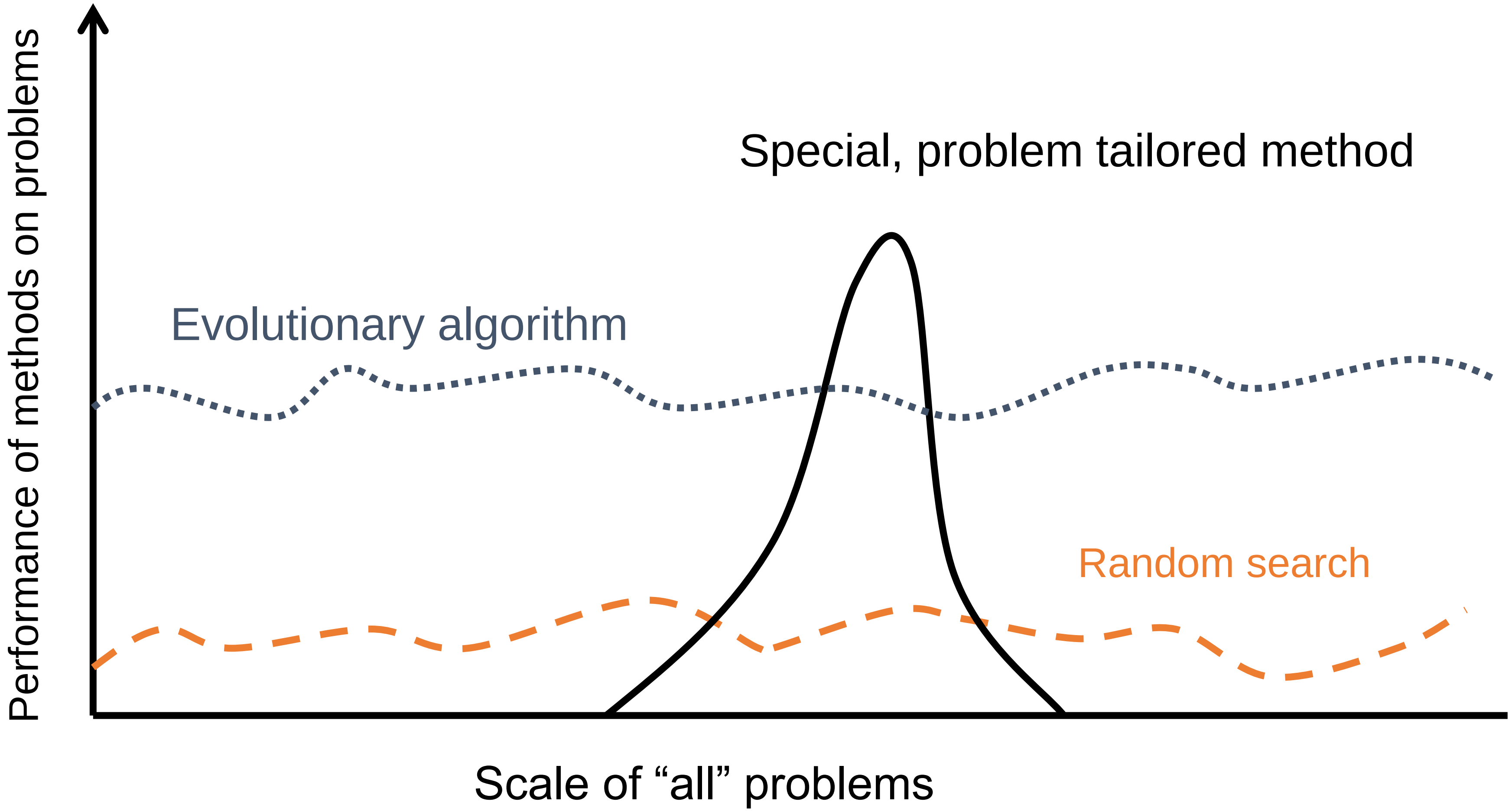
Applications of EAs





- Optimization of numerical continuous functions
  - ex: economic load dispatch problem, …
- Combinatorial Optimization Problems:
  - Traveling Salesman Problem, Flowshop Scheduling, Linear Ordering Problem, …
  - Knapsack, Number Partitioning, MAX-SAT, Subset Selection, …
  - Vehicle Routing
  - etc…
- Learning of Bayesian networks, neural networks, decision trees, etc…
- Choosing the hyperparameters of any Machine Learning algorithm
- Problems where the objective function is computed through numerical simulations
- Multi-objective optimization, multi-modal optimization, etc…

Goldberg (1989)



How many Evolutionary Algorithms?

- Based on Darwinian principles:
  - Genetic Algorithms
  - Differential Evolution
  - Evolution Strategies (es: CMA-ES = Covariance Matrix Adaptation Evolution Strategy)
  - Estimation of Distribution Algorithms
  - …
- Based on Swarm Intelligence principles:
  - Particle Swarm Optimization
  - Ant Colony Optimization
  - Bacterial Foraging Optimization
  - Artificial Bee Colony
  - …

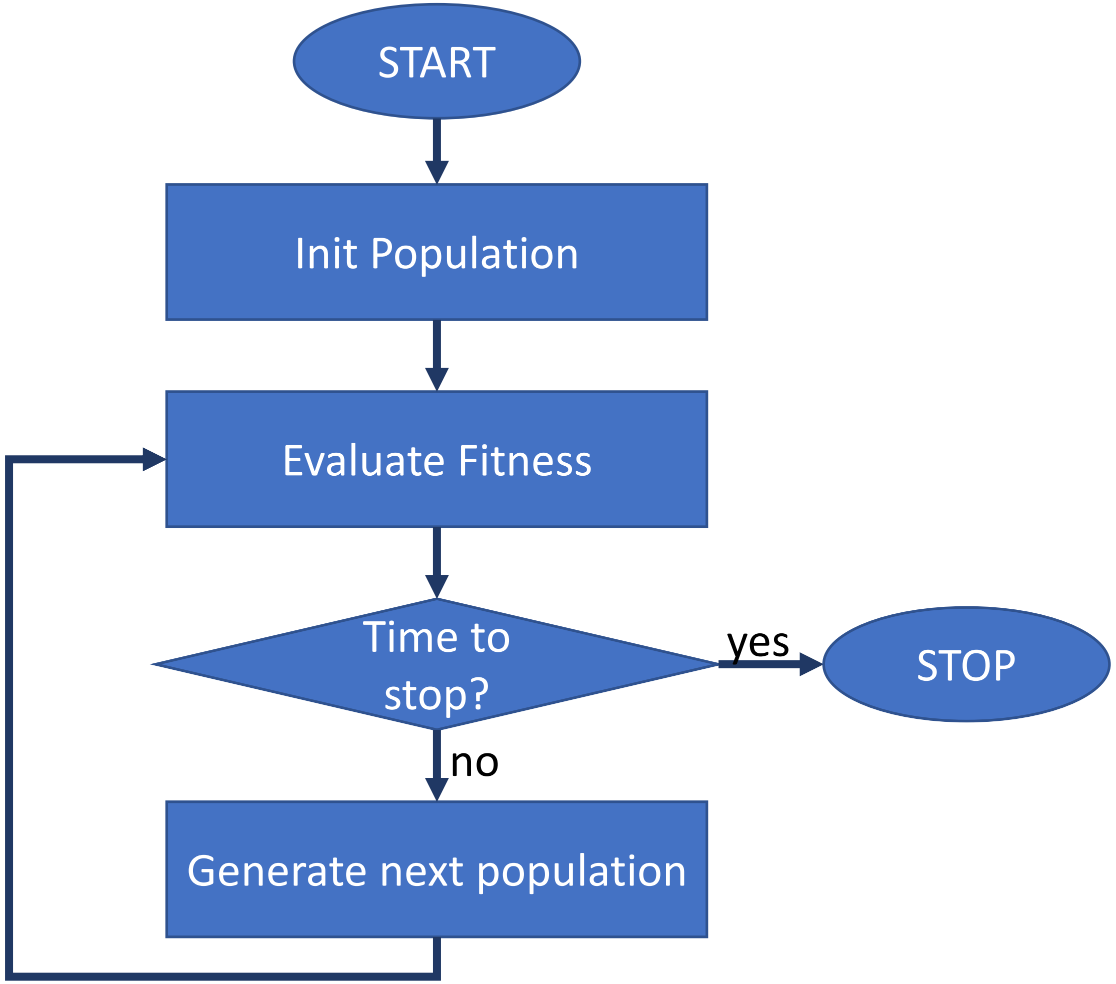
Outline

- What are Evolutionary Algorithms?
- <u>Ingredients of an EA</u>
- A simple Genetic Algorithm
- Current trend in Evolutionary Computation
- Analysis of EAs

Ingredients of an Evolutionary Algorithms

- Fitness function
- Representation of the solutions
- Variation and selection operators

- Metaphors:
  - Individual = Candidate solution
  - Population = Set of candidate solutions
  - Fitness = Quality of a solution
  - Generation = Iteration
  - Genotype = Encoding of a solution
  - Phenotype = Proper solution
  - Diversity … of the population
  - …



Representation

- Typical representations:
  - Vectors of real numbers for continuous problems
  - Bit-strings for binary problems
  - Permutations of integers per ordering, routing, scheduling problems
  - Strings over any finite alphabet
  - Syntactic trees for Genetic Programming
  - …
- Genotype vs Phenotype
  - EX: Graph = Phenotype, Adjacency matrix of the graph = Genotype
  - EX: bit-strings with Gray-Code encoding for representing real numbers
  - Relations between genotype and phenotype can be 1-to-1, 1-to-many, many-to-1
  - Encoding/decoding schemes may sometime be considered (EX: encode a permutation of the first *n* integers as a *n*-dimensional real vector, than decode the real vector to a permutation using "arg-sort")

Fitness Function

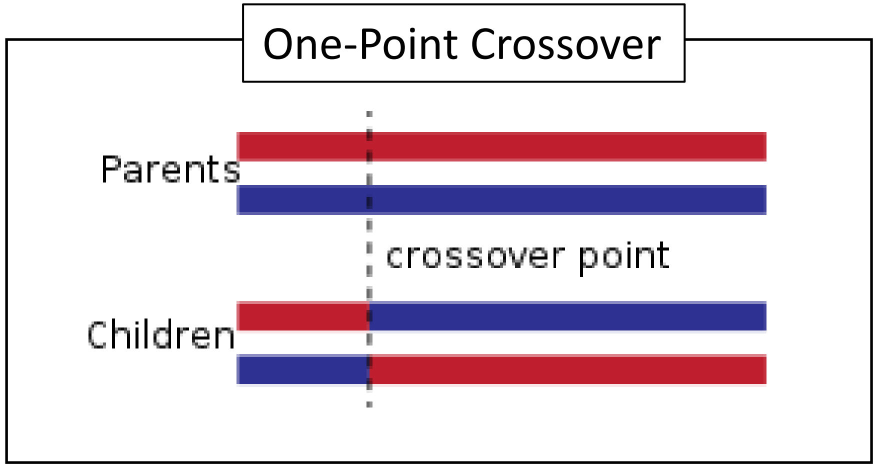
- Evaluate the quality of an individual

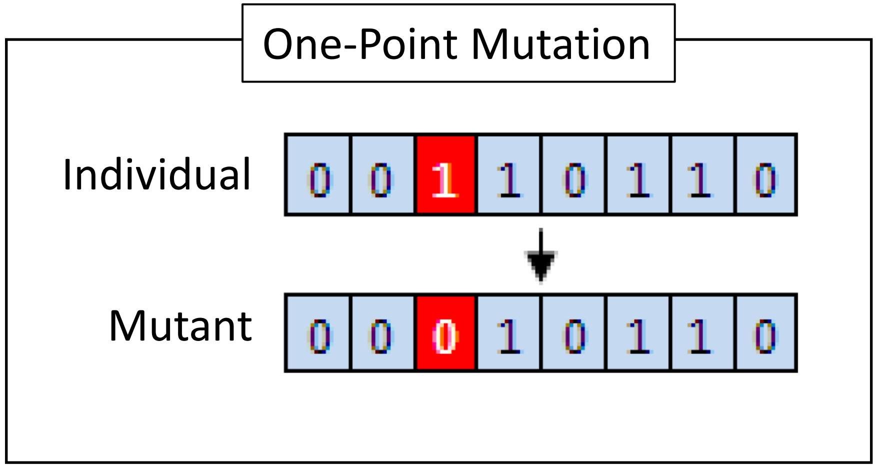
$$f : S \to \mathbb{R}$$

- Generally, it is the objective function of the given optimization problem. Sometimes it is modified via a monotonic transformation.
- When the problem has constraints, the fitness function may include penalty terms (i.e., penalize those solutions which violates the constraints in such a way that the evolution pushes back the solutions in the feasible region of the search space)
- The «fitness sharing» scheme is sometime used in order to maintain a certain level of population diversity throughout the evolution.

Variation Operators

- They generate one or more offspring individuals from one or more inputted individuals
  - EX: crossover, mutation
- Strictly tight to the chosen representation
- Any EA generally uses more than one variation operator





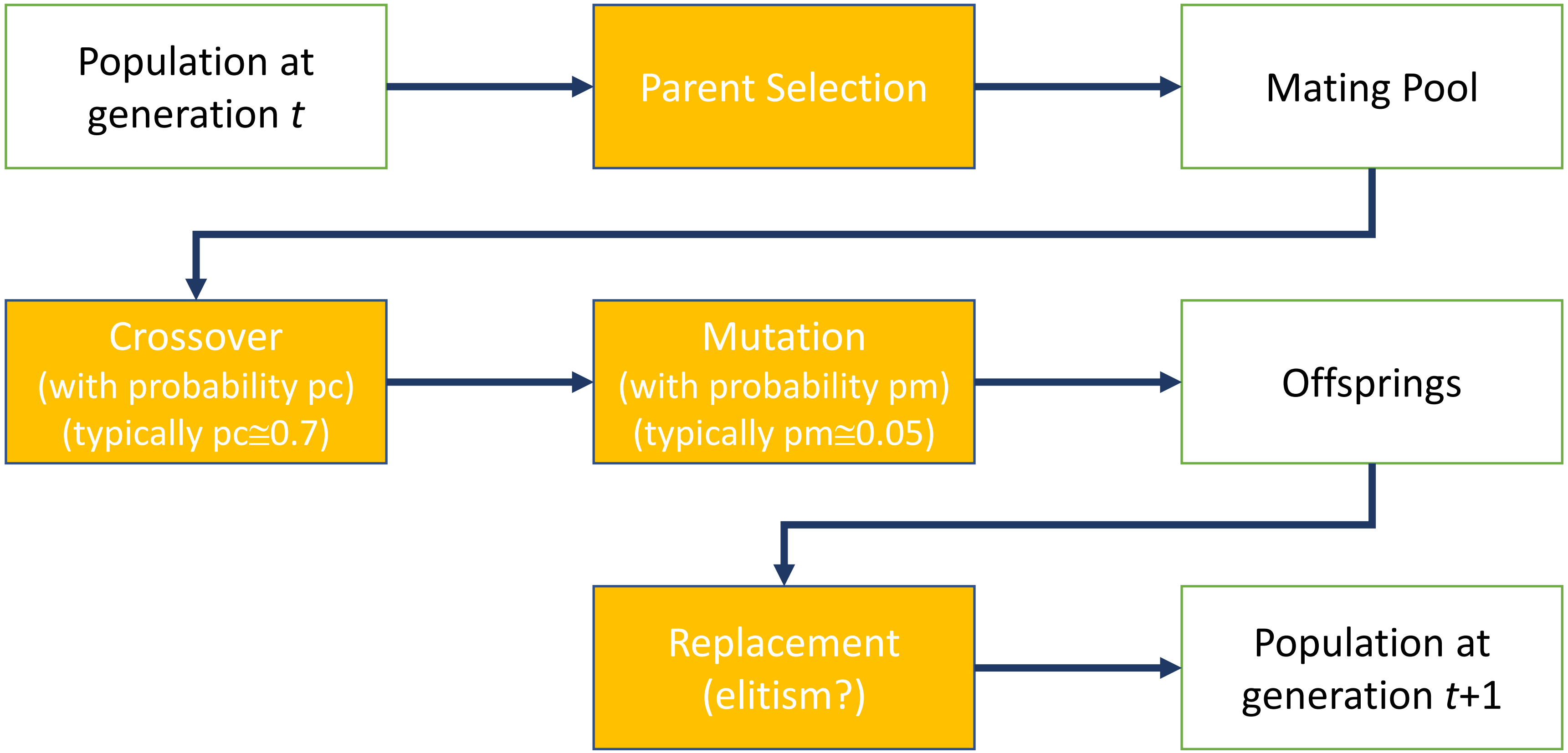
Selection Operators

- Select individuals from a given population
- Selection is based on the fitness of individuals (survival of the fittest)
- Independent from the chosen representation
- EX: selection and replacement in GAs

Outline

- What are Evolutionary Algorithms?
- Ingredients of an EA
- <u>A simple Genetic Algorithm</u>
- Current trend in Evolutionary Computation
- Analysis of EAs

Workflow of a Genetic Algorithm (GA)



An example of a GA: Fitness + Representation

- Fitness function. Two example problems:
  - Number Partitioning Problem (NPP): given a set of *n* numbers, divide them into two partitions such that the sums of the numbers in the two partitions are as close as possible.
  - Features Selection (FS): given a data-matrix of *n* features + a target class, find a subset of the features which gives the maximum accuracy for a chosen classifier.

- Representation. Both NPP and FS can be seen as binary problems, so every solution may be represented as a string of *n* bits.

An example of GA: Selection Operator

- We have a population of *N* candidate parents
- We need to select a population of *M* parents (possibly duplicated)

- Three possible alternatives are:
  - Truncament Selection
    - Select the best *M* individuals among the candidate parents.
  - Tournament Selection
    - Given a *tournament_size* parameter, repeat *M* times: select a parent as the best one among *tournament_size* randomly selected parents.
  - Roulette Wheel Selection
    - Repeat *M* times: select an individual (with replacement) with probability proportional to its fitness (or a monotonic transformation of the fitness, such as: ranking, square, logarithm or exponentiation).
    - Actually, it is an application of the roulette wheel sampling algorithm for multinomial distributions.

- Exploration vs Exploitation dilemma
  - Truncament selection is more biased towards best solutions. This allows a quicker convergence but does not exclude the convergence is towards a not very good solution.
  - Truncament S. is more exploitative, while Roulette Wheel S. and Tournament S. are more explorative.

An example of GA: Crossover Operator

- Given two parents, generate two offsprings which recombine the bits in the parents

- Two possible alternatives are:
  - One Point Crossover
    - Randomly select a cut-point that divide any parent and any offspring in left and right part
    - `Child1` takes the left bits from `Parent1` and the right bits from `Parent2`
    - `Child2` takes the left bits from `Parent2` and the right bits from `Parent1`
  - Uniform Crossover
    - Both offsprings are generated by taking each bit at random from `Parent1` or `Parent2` (with equal probability, i.e., 1/2)

- Note: sometimes the two crossover operators can also be restricted to produce just one offspring. In this case, think to the offspring as selected randomly from the two child solutions as described above.

- One-point crossover preserves larger parents' chunks than uniform crossover, so they may have different behaviors if the positions of the bits are important for the problem at hand or not. What about NPP and FS?

An example of GA: Mutation Operator

- Given an individual (an offspring) randomly mutates something in each genotype (its bits)

- Two possible alternatives are:
  - Single Bit-Flip Mutation
    - Randomly select a bit and flip it
  - Uniform Mutation
    - For each bit, flip it with probability 1/n (where *n* is the number of bits in the search domain)

- Single bit-flip mutation flips exactly one bit, while uniform mutation flips one bit in average, so sometimes it may flip more than one bit. This is useful because it may allows to escape from local optimal solutions.

An example of GA: Replacement Operator

- Given two populations, both of size *N*:
  - X = the previous iteration population
  - Y = the offspring generated during the last iteration
- Combine them and generate the population X for the next iteration

- Two alternatives:
  - + strategy ('plus strategy')
    - Take the 'n' best individuals from $X \cup Y$
  - , strategy ('comma strategy')
    - The new generation population is simply Y

- Plus-strategy is elitist, i.e., it never forgets the best individual, but it may converge too fast to a solution which is not good enough. Comma-strategy may be better if the budget of time is enough.

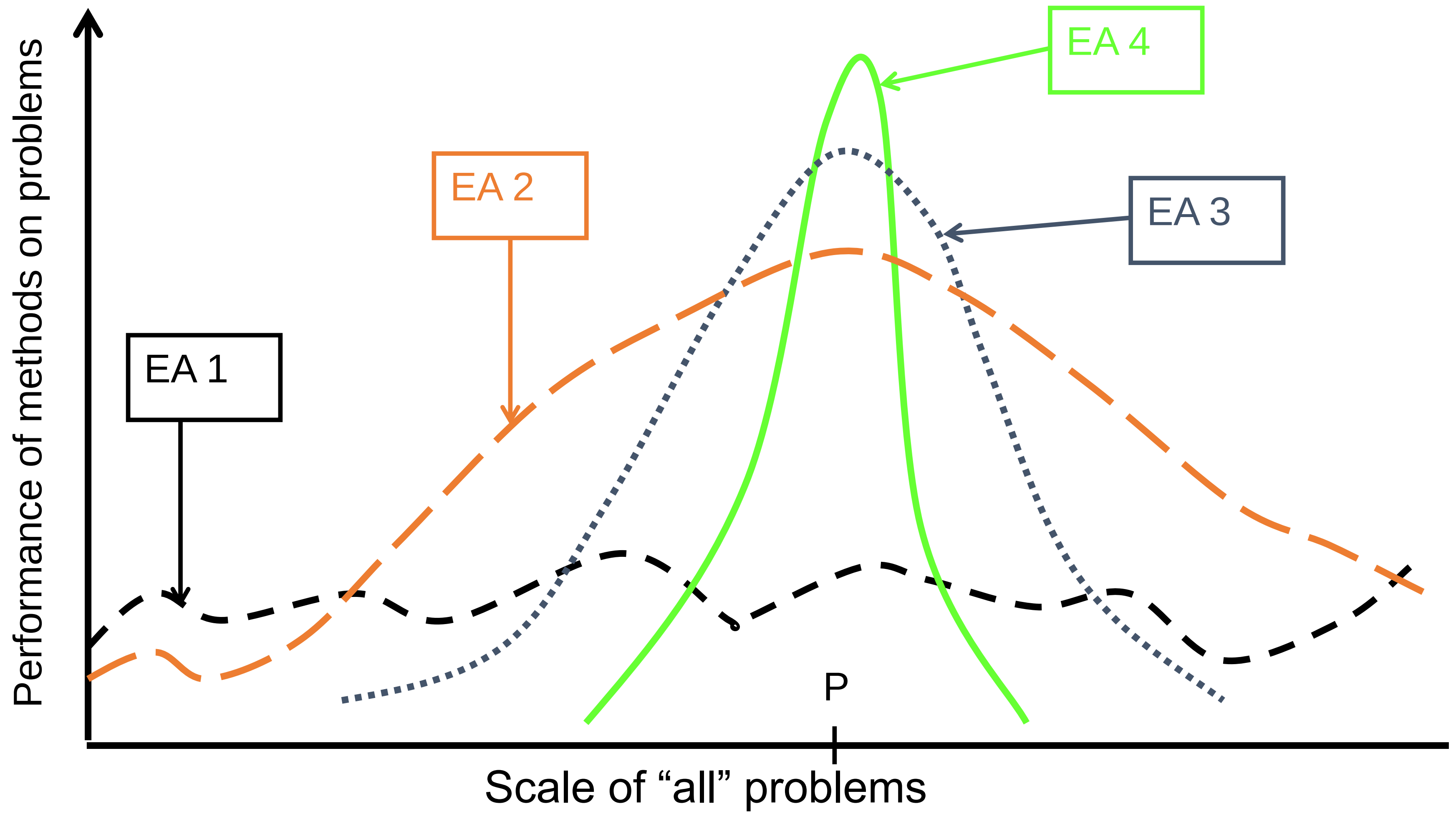
Hands on GA in Python

- See the files in `binary_ga_example.zip`

Outline

- What are Evolutionary Algorithms?
- Ingredients of an EA
- A simple Genetic Algorithm
- <u>Current trend in Evolutionary Computation</u>
- Analysis of EAs

Michalewicz (1996)



Outline

- What are Evolutionary Algorithms?
- Ingredients of an EA
- A simple Genetic Algorithm
- Current trend in Evolutionary Computation
- <u>Analysis of EAs</u>

Analysis of EAs

- EAs are complex systems where entities interact in order to reach a common goal (finding the optimum of a given optimization problem)
- Deriving theoretical guarantees is not easy, though run time analysis involving probability theory are currently available for simple variants of EAs on simple optimization problems
- A very basic concept useful for analyzing the behaviour of EAs:
  - Uniform crossover and uniform mutation independently flip each bit in a bit-string, so the probability of combined events such as "a given set of bits are flipped" is the product of the single probabilities.

References

- Eiben, Smith. "Introduction to Evolutionary Computing"

https://warin.ca/ressources/books/2015_Book_IntroductionToEvolutionaryComp.pdf

- This blog post is also interesting: https://medium.com/@AnasBrital98/genetic-algorithm-explained-76dfbc5de85d (open it in anonymous browser window)

- Nevergrad Documentation: https://facebookresearch.github.io/nevergrad/