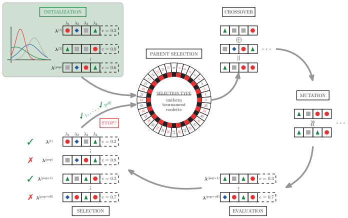
Optimization in Machine Learning

Evolutionary Algorithms
Introduction



Learning goals

- Evolutionary algorithms
- Encoding
- Parent selection, variation, survival selection

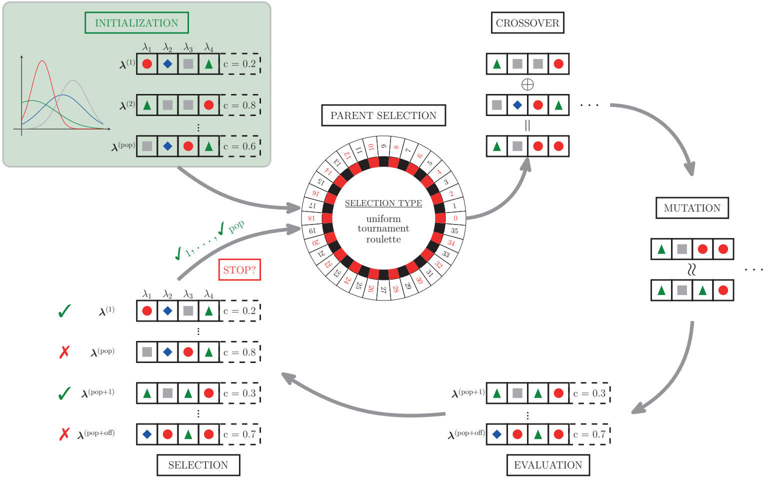
EVOLUTIONARY ALGORITHMS

Evolutionary algorithms (EA) are a class of stochastic, metaheuristic optimization techniques whose mode of operation is inspired by the evolution of natural organisms. History of evolutionary algorithms:

- **Genetic algorithms**: Use binary problem representation, therefore closest to the biological model of evolution.
- **Evolution strategies**: Use direct problem representation, e.g., vector of real numbers.
- **Genetic programming**: Create structures that convert an input into a fixed output (e.g. computer programs); solution candidates are represented as trees.
- **Evolutionary programming**: Similar to genetic programming, but solution candidates are not represented by trees, but by finite state machines.

The boundaries between the terms become increasingly blurred and are often used synonymously.

STRUCTURE OF AN EVOLUTIONARY ALGORITHM



NOTATION AND TERMINOLOGY

- A chromosome is a set of parameters which encodes a proposed solution to the problem that the genetic algorithm is trying to solve. The chromosome is often represented as a binary string, although a wide variety of other data structures are also used.
- The set of all solutions is known as the population.

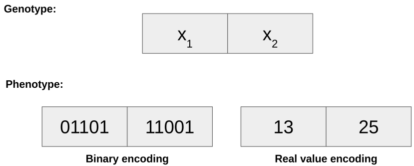
Symbols	**EA Terminology**
solution candidate $\boldsymbol{x} \in \mathcal{S}$	chromosome of an individual
x_j	j-th gene of chromosome
set of candidates P with $\mu = \lvert P \rvert$	population and size
λ	number of generated offsprings
$f : \mathcal{S} \rightarrow \mathbb{R}$	fitness function

Note: Unintuitively, we are minimizing fitness because we always minimize f by convention.

ENCODING

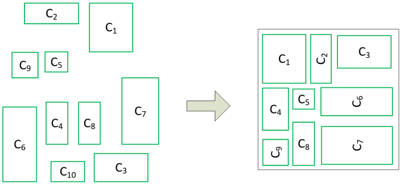
Encoding of chromosomes is the first step of solving a problem with EAs. Technically: Mapping from **genotype** to **phenotype**. Encoding depends on the problem, and eventually decides performance of problem solving. **Encoding methods**:

- Binary encoding: Strings of 0s and 1s
- Real value encoding: Real values



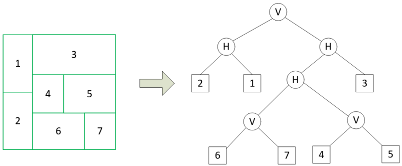
ENCODING

- Tree encoding: Tree objects



10 circuits

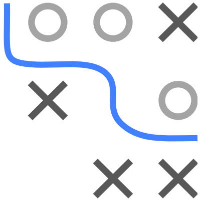
A floor plan



▸ Click for source

Floor planning problem. Given are *n* circuits of different area requirements. Goal: arrange them into a floor layout so that all circuits are placed in a minimum layout. Each solution candidate can be represented by a tree.

STEP 1: INITIALIZE POPULATION

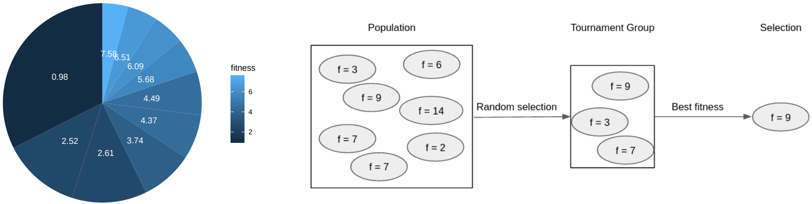


- Evolutionary algorithms start with generating initial population $P = \{\boldsymbol{x}^{(1)}, ..., \boldsymbol{x}^{(\mu)}\}$.
- Usually: Initialize uniformly at random.
- Introducing prior knowledge possible.
- Population is evaluated: objective function is computed for each initial individual.
- Initialization influences quality of solution, so many EAs employ *restarts* with new randomly generated initial populations.

STEP 2: PARENT SELECTION

Choose a number of λ parents pairs creating λ offsprings.

- **Neutral selection:** Draw parents uniformly at random.
- **Fitness-proportional / Roulette wheel selection:** Draw individuals with probability proportional to their fitness.
- **Tournament selection:** Randomly select *k* individuals for a "tournament group" and pick the best one (according to fitness value).



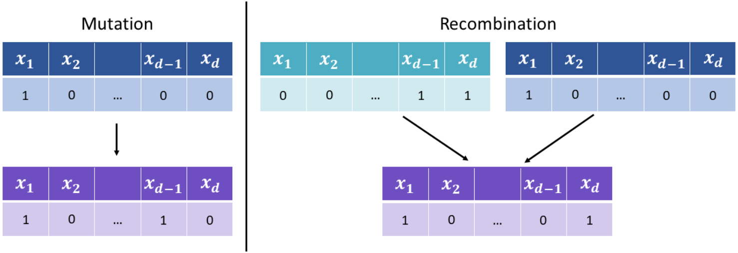
Left: Fitness-proportional selection. Fitness values of $\mu = 10$ individuals are converted into probabilities. **Right:** Tournament selection.

STEP 3: VARIATION

New individuals (offsprings) are generated from parents.

- Recombination/Crossover: Combine two parents into offspring.
- Mutation: Modify the offspring locally.

Sometimes only one of both operations is performed.



Note: Particular operation depends on encoding. Examples for binary and numeric encodings follow later.

STEP 4: SURVIVAL SELECTION

Choosing surviving individuals. Two common strategies are:

- (μ, λ)**-selection:** Select μ best individuals *only from set of offsprings* ($\lambda \geq \mu$ necessary). **But:** Best individual can get lost!

- $(\mu + \lambda)$**-selection:** Select μ best individuals from set of μ parents and λ offsprings **Now:** Best individual certainly survives.

EVOLUTIONARY ALGORITHMS

- **Advantages**
- Simple but enough to solve complex problems
- All parameter types possible in general
- Highly parallelizable
- Flexible through different variation operations

- **Disadvantages**
- Little mathematical rigor (for realistic, complex EAs)
- Hard to find balance between exploration and exploitation
- Quite some parameters, hard to determine them
- Customization necessary for complex problems
- Not suitable for expensive problems like HPO as large number of function evaluations necessary