NTNU | Norwegian University of Science and Technology

EVOLUTIONARY ALGORITHMS

Optimisation Metaheuristics

Xavier Sánchez Díaz

October 28, 2021

Contents

Evolutionary Computation

Introduction

Evolutionary computation is a branch of artificial intelligence focused on solving optimisation problems.

- **Parametric:** Real-valued functions in continuous search spaces.
- **Discrete:** Vectors in $\mathbb{B}^n$ or $\mathbb{Z}^n$.
- **Combinatorial:** Combinations or permutations of numerical values; usually in discrete settings.

The Evolutionary part

Key concepts

The evolutionary part of an algorithm refers to its metaphor with respect to how **evolution** and **natural selection** work.

1. Start with a population of candidate solutions
2. Evaluate all and select some of them
3. Evolve some of them through *natural* operators
   - Sexual reproduction (crossover)
   - Mutation
4. Replace *some* of them and keep best solutions for the next generation

The process can go on for as many generations you need.

The Evolutionary part

Key concepts

The evolutionary part of an algorithm refers to its metaphor with respect to how **evolution** and **natural selection** work.

1. Start with a population of candidate solutions
2. Evaluate all and select some of them
3. Evolve some of them through *natural* operators
   - Sexual reproduction (crossover)
   - Mutation
4. Replace *some* of them and keep best solutions for the next generation

The process can go on for as many generations you need.

The optimisation part

Key concepts

These algorithms work by using the following **operators**:

- *Exploitation* of the best solutions is done by selecting and keeping record of these solution candidates.
- *Exploration* of the search space is done by mutation and crossover (if present).

Many research has been done on multiples strategies for selection, mutation and crossover.

What can I do with them?

Uses and applications

From my own experience:

- **Knapsack applications**: select a subset of items which maximises the value of their price if you have a container with limited capacity.
- **Constraint satisfaction problems**: find assignments of values to variables that do not violate any constraints between them.
- **Hyper-parameter setting**: find a set of parameter values to maximise a machine learning model statistic (accuracy, F-score, AUC, etc.)

We would like to test on industrial layouts design and delve further into hyper-parameter setting!

How can I use them for my research?

Python Implementations

- TPOT can be used to optimise machine learning pipelines using **genetic programming** if you are using `sklearn`.
- DEAP has some algorithms already implemented (like **simple** or the **1+1 EA**).

…or you can **make your own** with DEAP!

https://saxarona.github.io/project/evo-intro