TD3: Evolutionary Algorithms

This exercise sheet also serves as a model exam, in that it has the same appropriate structure and level of questions as the actual exam. However, this sheet is restricted to the topic of evolutionary algorithms, whereas the exam covers the whole course.

Instructions:

- You have 90 minutes to finish the exam.
- No calculator or documents are allowed, other than an A4 sheet of notes, front and back.
- Fully read the exercise statement before starting to solve it, and always justify your answers.
- If an exercise asks for an algorithm, describe it in pseudo-code or Python.

1 Short Questions

The following are short questions from various topics, meant to test your understanding of the material and to be answered in a few lines.

Exercise 1. For this question, we will simulate parent selection by tournament. Consider the population $P = \{A, B, C, D, E, F\}$, with fitness values $6, 5, 4, 3, 2, 1$ respectively. Your task is to select 6 parents using best-2-out-of-5 tournaments.

For the random selections, use the sequence

$$C, D, E, B, F, B, E, C, A, D, A, C, F, D, E, E, A, F, B, D, B, E, F, C, A, C, B, F, D, A$$

Exercise 2. Consider a genetic algorithm with the following characteristics:

- Solutions are encoded as permutations;
- No mutation operator;
- The recombination operator is fixed-point permutation insertion, which inserts a subinterval of the second parent in the first at the *same position*, and removes duplicates coming from the first parent.

What is the effect of using no mutation in the population diversity? Use as an example the case where all individuals have the same values in the first 3 positions (e.g. every permutation starts with $(1, 4, 3)$).

Exercise 3. Consider a genetic algorithm with the following characteristics:

- Population of size 1;
- Two children are generated, and the best one is selected for next generation;
- A new individual is a mutation of the parent with probability 0.5 or the same as the parent with probability 0.5;
- Recombination is not used.

Compare this algorithm with simulated annealing, assuming that the mutation operator is the same as the random neighbor operator in simulated annealing.

Exercise 4. Consider a genetic algorithm for the TSP problem with the following characteristics:

- Solutions are represented as a binary string by identifying each vertex by a number, writing those numbers as binary, and concatenating the sequence of binary numbers in a solution to obtain a binary string. For example,

$$(v_1, v_4, v_3, v_2, v_5, v_1) \to (1, 4, 3, 2, 5) \to (100, 011, 010, 101, 001) \to 100011010101001$$

- Bit-flip mutation and binary crossover recombination.

Using this algorithm, explain two disadvantages of this representation for the TSP problem.

2 The Bin Packing Problem

In the bin packing problem, we are given a set of n items with weights $w_1, \ldots, w_n$, as well as a value C, called capacity. Our goal is to pack all of the items into as few boxes as possible, while respecting their capacity. In other words, we assign each item to a bin, such that the total weight of items assigned to a bin does not exceed C, and we want to minimize the total number of bins used.

Exercise 5. Design a genetic algorithm for bin packing, by specifying:

1. Solution representation;
2. What is the fitness function and how to compute it;
3. Mutation and recombination operators.

Justify your choice of solution representation.

Exercise 6. Suggest a method for parent selection and another for survival selection. How do these methods perform in terms of selection pressure and genetic diversity?

3 Genetic Algorithms with Local Search

An approach that has achieved some success is to improve genetic algorithms by use of local search algorithms. The general approach is that, after generating the children, a local search operator (such as repeated 2-opt) is applied to improve the individuals.

There are two approaches regarding the role of local search in the algorithm:

- The *artificial approach* says that the local search operator should be applied after mutation and **change the genome** of the individuals, whereas
- The *natural approach* says that the local search operator should be applied **only as part of the fitness**, that is, survival selection takes into account the fitness of the solution after local search, but the **genome remains unchanged** and thus the changes from local search are lost in the next iteration.

Exercise 7. Compare these two methods in terms of selection pressure and genetic diversity.

Exercise 8. A third approach, which mixes the two above, keeps both the original genome and the genome improved by local search (as different individuals). In terms of results and solutions that are produced, is this approach closer to the artificial or natural approach?

Exercise 9 (extra). As we know, greedy local search can be computationally heavy, due to testing every neighbor for a possible improvement. Thus, it makes sense to apply local search only to a few individuals.

For this exercise, we assume a population of size n, individuals of size d (for example, a bit string of length d), and g generations. We also assume that mutations, recombinations, and fitness computation take time $O(d)$ per element, and that selection takes time $O(n)$.

1. What is the running time of the genetic algorithm without local search?
2. Now, assuming that a local search algorithm takes time $O(d^3)$, and we apply local search to every element, how does the running time change? Does the running time depend on whether we use artificial or natural approach?
3. Finally, we change our algorithm so that only a constant number of individuals in each iteration get improved by local search. What is the new running time in this case?