CS4205 - Evolutionary Algorithms

Assignment 2

This document describes the final practical assignment of the TU Delft course Evolutionary Algorithms (CS4205). This assignment will count for 30% of the final grade and can be done on one of the different topics listed below, each with a respective supervisor:

- *Evolutionary Van Gogh* (Single-Objective Discrete Optimization)
  Supervisor: Damy Ha
- *Genetic Programming for Symbolic Regression* (Genetic Programming)
  Supervisor: Johannes Koch
- *Circles in a Square* (Single-Objective Real-Valued Optimization)
  Supervisor: Anton Bouter
- *Evolving a Lunar Lander with Genetic Programming*
  Supervisor: Thalea Schlender

A more detailed description for each of these assignments is provided in the appendix.

The basic principles of the assignment are the same across the topics. This assignment is expected to be done in *groups of 5 students*. You are supposed to self-arrange into groups, there is a discussion forum for group finding on Brightspace to help you with this. Groups of less than 5 students may receive additional members or be split up at the discretion of the teaching staff.

Once you have a group, each of you needs to sign up on the same group in the Brightspace group interface. We will assign the topics to groups using a preference-maximization approach. For this, prior to the start of this assignment, each group of students must denote their relative preference for each of the 4 possible exercises in the Brightspace survey. The deadline for signing up for a group *and* for indicating your topic preference is **Sunday, May 18, 2025, 23:59**.

Requirements

Weekly progress meetings will be held between the group and the designated supervisor. During each of these meetings, you are required to give an update on your progress and you are given the opportunity to ask questions. For the scheduling of these meetings, each group will be individually contacted by their designated supervisor.

The requirements of the final deliverable are as follows. Each group must hand in the following *deliverables* on Brightspace in a single zip file prior to the deadline of **Sunday, June 8, 2025, 23:59**:

- All source code used to produce experimental results in a *zip* file.
- Presentation slides (PDF/PPTX format) to be presented in a **9** minute final presentation on **June 11** or **June 12, 2025**. The schedule for the final presentations will follow at a later date.

Grading

A **complete** submission of the deliverables is required and the source code should run (else, it is a fail). The final grade is based on the following criteria, for which a rubric will be made available at a later stage:

- [**60%**] Content
- [**20%**] Presentation
- [**20%**] Defense

Solving Circles in a Square

CS4205: Evolutionary Algorithms - Assignment 2
Supervisor: Anton Bouter (bouter@cwi.nl)
Topic: Real-Valued Single-Objective Optimization

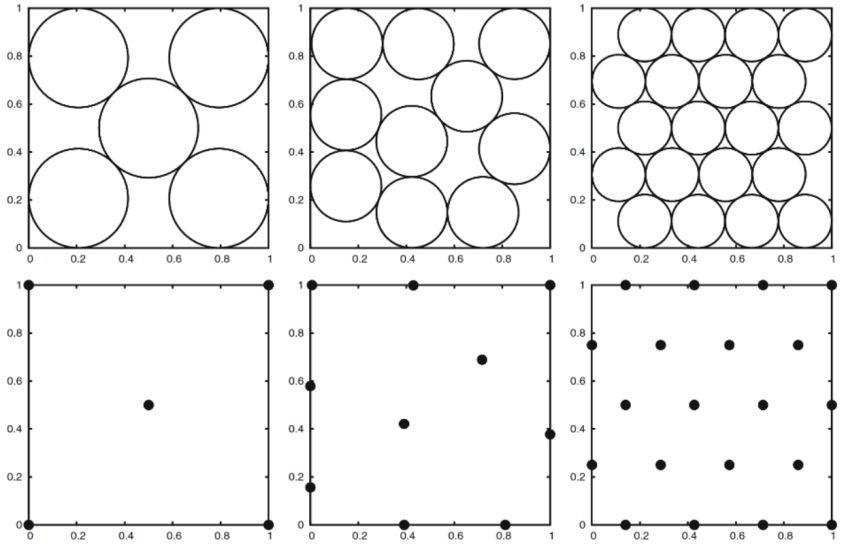

Figure 1: Top row: optimal circle packings for 5, 10, and 20 circles. Bottom row: corresponding point scatterings [1].

1 Problem Definition

This final assignment considers a scalable and real-valued optimization problem called "Packing n Circles in a Square (CiaS, see, e.g., [1])." This problem has many applications in real-world optimization problems like loading containers, communication networks, facility dispersion/layout, etc. Even though CiaS can be compactly formulated and efficiently evaluated, it is not trivial to solve and has interesting characteristics that differ from many other commonly used benchmark problems. For at least 2 circles, the CiaS problem is equivalent to finding an optimal scattering of points in the unit square. To obtain the optimal scattering of points, the smallest distance between any pair of points has to be maximized. Examples of optimal circle packings and their corresponding point scatterings are shown in Figure 1. The fitness function for the point scattering problem is as follows:

$$f_{scatter}(\boldsymbol{x}) = \min_{0 \leq i < j < \ell/2} ||(x_{2i}, x_{2i+1}) - (x_{2j}, x_{2j+1})||$$

2 Goal of assignment

To obtain state-of-the-art performance on CiaS is not trivial, as specific properties of the CiaS problem must be considered and exploited. It would be interesting to see to what extent EAs can be tuned to CiaS to also obtain state-of-the-art results with an EA. You will be doing so by improving one of the provided EAs and perform a performance comparison before and after it has been tuned for CiaS.

An especially useful resource is the Packomania website [2]. On this website a large list of the optimal, or best known, packings is maintained. Currently, the list goes up to 9996 circles, with some intermittent omissions. You can also find other references and resources for more information on the website.

3 Provided algorithms

In order to give you a head start with some readily available code, the implementations for two algorithms are provided. Both have been implemented in a different language in order to give you a choice that suits your preference. The two algorithms are:

- AMaLGaM [3] (C code)
- Evolution Strategies [4] (Python code)

4 Recommended Project Timeline

This section describes an outline of the expected progress over the course of the assignment. Nevertheless, these guidelines are not enforced and only your final results are graded.

Week 1. Your assignment is to improve the baseline performance of an EA and report the findings. To measure the baseline, you are to benchmark the EA as provided and produce graphs or tables with the results. You should also make yourself familiar with the code and think of ideas for the improvement that you are going to make (see week 2).

Week 2. In the second week you should start with implementing the proposed changes in order to improve the performance of the EA on the CiaS problem. To give you an idea of what you should work on, you should change or work on at least the following three points for the EA:

1. Change the constraint handling technique by implementing either boundary repair [1] or the more advanced constraint domination [5].
2. Create a problem specific initialization scheme
3. Optimize the hyperparameters (population size, etc)

A guideline for the second week is to have an implementation for the points mentioned above, and have some preliminary results on whether they are working as expected.

Week 3. At the end of the third week you are required to hand in the final deliverables (code and presentation slides). Thus, you should finish the benchmarks for the algorithm and complete the presentation. The choice of the proposed changes must be motivated and the outcome of the comparison clearly conveyed. Make sure that the comparison is done in a fair and reproducible manner and that statistical significance testing has been performed to determine if the changes resulted in a significant improvement.

Related work (also to get you started)

[1] Peter A. Bosman and Marcus Gallagher. The importance of implementation details and parameter settings in black-box optimization: A case study on gaussian estimation-of-distribution algorithms and circles-in-a-square packing problems. *Soft Comput.*, 22(4):1209–1223, feb 2018.

[2] Eckard Specht. Packomania website, Feb 2022. `www.packomania.com` (accessed: 30.03.2022).

[3] Peter A. N. Bosman, Jörn Grahl, and Dirk Thierens. Benchmarking parameter-free amalgam on functions with and without noise. *Evolutionary Computation*, 21:445–469, 9 2013.

[4] Nikolaus Hansen, Dirk V. Arnold, and Anne Auger. Evolution strategies. In Janusz Kacprzyk and Witold Pedrycz, editors, *Springer Handbook of Computational Intelligence*, pages 871–898. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[5] Kalyanmoy Deb. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186(2):311–338, 2000.