ECMM423 – Evolutionary Computation and Optimisation

Assignment Two: Search & Optimisation Project

Hand-out date: **3rd March 2024**
Hand-in date: **2th May 2024**
Feedback by: **25th May 2024**
This CA is worth **60%** of the overall module mark

This is an **individual assessment** and you are reminded of the University's Regulations on Collaboration and Plagiarism, and also the University's Regulations on Using generative Artificial Intelligence (AI) tools such as ChatGPT in academic work.

Overview

This coursework is intended to enhance your understanding of evolutionary optimisation techniques, and give you practice in the design, implementation, experimentation and written presentation of your work in an academic fashion. The gold standard of modern academic research is to make published work repeatable and verifiable. To support this, researchers are increasingly publishing the codebase of their work, via sites like GitHub and Sourceforge, as well as on their personal pages. A more recent trend is to publish code, documentation, experiments and analysis as a Jupyter notebook.

Assignment

The project requires you to apply a search algorithm to an optimisation problem and to conduct computational experiments with the algorithm. You should produce a Jupyter notebook and an 'Article Style' report describing your work, described in more detail below.

Jupyter Notebook

You should produce a Jupyter notebook which is reproducible and self-contained (i.e., a single *.ipynb file) and that includes well-documented implementation in Python of the following:
(i) the search algorithm,
(ii) the problem instances considered,
(iii) parameter tuning,
(iv) experimental comparison with random search and a stochastic hill-climber,
(v) analysis of results (i.e., graphs and tables).

Clear instructions on how to reproduce experiments should be provided in the notebook along with the results clearly shown in the notebook cells. The notebook will be tested in Google Colaboratory[1], so make sure that it runs there.

Report

You should produce a self-contained report of the work done. The report should be presented in an 'Article Style', inspired by those we have covered in the seminars. This should include a title, abstract and keywords, along with an introduction, literature review covering both search algorithm and optimisation problem, the main body (including problem formulation, search algorithm, adaptation of search algorithm to the problem i.e., problem-specific representation and search operators), experiments (including an experimental plan covering performance criteria, problem instances, algorithms in the comparison, parameter tuning, and experimental results with tables and graphs and

[1] https://colab.research.google.com/

analysis of these), conclusion and references. You must use the ACM SIG Proceedings style, for which there are Word and LATEX templates available[2], it must not exceed 6 pages in length. You are required to pick one search algorithm (either local search or population-based search) and one type of optimisation problem from the lists below. You may find it useful to identify a reference paper and a set of benchmark problems and aim at reproducing some of the results in the paper, as well as some variations on these. The set of search algorithms and optimisation problems that you may choose from are shown below:

Search Algorithms (Local Search)

1. Simulated Annealing
2. Tabu Search
3. Iterated Local Search
4. Guided Local Search
5. Variable Neighborhood Search
6. Greedy Randomized Adaptive Search

Search Algorithms (Population-Based)

1. Evolutionary Algorithm (including multi and many-objective algorithms)
2. Differential Evolution
3. Particle Swarm Optimization
4. Estimation of Distribution Algorithm
5. Covariance Matrix Adaptation Evolutionary Strategy

Machine Learning Algorithms

1. Convolutional neural networks (CNN)
2. Recurrent neural networks (RNN)
3. Long short-term memory (LSTM) network
4. Transformer

Optimisation Problems

1. Standard Benchmark on Binary Strings
2. Standard Benchmark on Real Vectors
3. Three real-world web application firewalls in the DaNuoYi paper.
4. Textbook Combinatorial Problem on Permutations (e.g., Travelling Salesman Problem, Vehicle Routing Problem, Job Shop Scheduling Problem)
5. Standard Benchmark for Symbolic Regression (i.e., Genetic Programming or Grammatical Evolution)
6. Standard Benchmark for Finite-State Machine Induction (i.e., Evolutionary Programming)
7. Standard Regression/Classification Problems (i.e., Neural architecture search)

Submission

You should submit a zip file containing both the report (as *.pdf document) and the Jupyter notebook including all your code, with clear and precise instructions to reproduce your results by 12noon on the submission date at the top of the first page of this CA (note that this is the first week in term 3). You are allowed to use only the Python Standard Library and de-facto standard libraries such as Numpy, Matplotlib and Pandas (additional external dependencies are not allowed). The electronic submission should be made using the EBART system.

[2] https://www.acm.org/publications/proceedings-template

Marking Scheme

This assessment will be marked using the following criteria:

Introduction **10%**

The quality of the abstract and introduction

Literature Review **10%**

The quality of the literature review

Problem and Algorithm Description **10%**

The degree to which the report contains a well-justified and well-described problem formulation and design of search algorithms, search operators and solution representation

Pseudocode **10%**

The degree to which the report contains well-described and well-presented pseudo-code of algorithms

Experiments **10%**

The degree to which the report contains well-designed, well-described, reproducible experiments

Presentation of Results **10%**

The degree to which the report contains an understandable presentation of results, with tables and graphical illustrations

Analysis **10%**

The degree to which the report contains insightful analysis of all major experimental results

Coding and Documentation **10%**

The degree of correctness of the code and of clarity of comments and any documentation.
The degree to which the submission includes details on how to replicate the results in your submitted report.

Report **10%**

The degree to which the report is appropriately organised (logical, not necessarily chronological) and uses a clear writing style.

Excellence **10%**

The degree of additional excellence, e.g., complexity of problem and algorithm, novel design of search operators, elegance of implementation, range of experimentation, depth of analysis, excellence in presentation.

Total: **100%**