AI420 EVOLUTIONARY ALGORITHMS SPRING 2024

COURSE PROJECT INSTRUCTIONS

Instructions to Students:

- This is a group work project. Each group consists of **Four to Six students** *(the Teaching Assistant must approve group members through registration)*. Each group must develop the idea assigned to them using Python.
    - **Project Objectives:** The objectives of this project can be summarized as applying the main ideas, fundamental concepts, and basic algorithms in the field of Evolutionary Algorithms & Computing.
- **Submission:** Submission is done according to the following schedule:
    - **Saturday, May 11th 2024: Submission and Discussion of the** (1) **Project** and (2) **Documentation**. *The report should include the following: (1) Project idea in detail, (2) Main functionalities, (3) Similar applications in the market, (4) A literature review of Academic publications (papers) relevant to the idea (at least 4 to 6 papers, as per the number of team members), (5) the Dataset employed (preferably a publicly available dataset), (6) Details of the algorithm(s)/approach(es) used and the results of the experiments, and (7) Development platform.*
- **Assessment:** Assessment will be on the reports, code submitted, and discussions with team members. All the team members must contribute to all the phases, and the role of each member must be clearly stated in each report.
    - The Project will be assessed based on the following criteria:
        - The complexity of the problem, & the correctness of the algorithms employed.
        - The quality/comprehensiveness of your experiments & documentation.
        - The correctness of your analysis and design diagrams.
        - Implementation correctness.
- **Feedback:** If requested, further details and feedback could be provided for each group through discussions with the teaching assistant(s) during the weekly-labs/office-hours.
- You can only submit your work. Any student suspected of plagiarism will be subject to the procedures set out by the Faculty/University (including failing the course entirely).
    - **Academic Integrity:** The University's policies on academic integrity will be enforced on students who violate University standards of academic integrity. Examples of behaviour that is not allowed are:
        - Copying all or part of someone else's work and submitting it as your own;
        - Giving another student in the class a copy of your work and
        - Copying parts from the internet, textbooks, etc.
        - If you have any questions concerning what is allowed, please don't hesitate to discuss them with me.
    - **We understand that you might positively refer to AI tools to support your research. Please note that, in case there are any concerns about AI-generated submission content, you will be invited for an opportunity to verify and defend your work.**

AI420 Evolutionary Algorithms Spring 2024

Course Project Descriptions (8 Ideas)

Table of Contents

Important Guidelines for ALL Projects to follow:

a) Clearly define and formalize your problem as one of the problem types we've studied throughout this module: Optimization, Modelling, Simulation, Constraint Satisfaction, Free Optimization, Constrained Optimization, etc.
b) If your selected idea mandates Constraint Handling, clearly implement an approach to handling constraints (i.e., Penalty Functions, Repair Functions, Restricting Search to the Feasible Region, Decoder Functions, etc.).
c) If your selected idea mandates Coevolution, clearly implement a coevolutionary approach (cooperative or competitive).
d) Clearly define the Components of your Evolutionary Algorithm: For instance, in the case of a genetic algorithm, clearly define the Representation (Definition of Individuals), Evaluation Function (Fitness Function), Population, Parent Selection Mechanism, Variation Operators (Mutation and Recombination), Survivor Selection Mechanism (Replacement), Initialisation, and Termination Condition(s).
e) Select variation operators (mutation and recombination) suitable to the selected representation.
   - If possible, use at least 2 parent selection techniques (each independently) and report the results for each.
   - If possible, use at least 2 recombination techniques (each independently) and report the results for each.
   - If possible, use at least 2 mutation techniques (each independently) and report the results for each.
f) If possible, use at least 2 population-management-models/survivor-selection (each independently) and report the results for each.
g) Clearly describe and implement approaches to control/tune the parameters.

h) Describe and implement a suitable approach for preserving diversity (i.e., Fitness Sharing, Crowding, Automatic Speciation Using Mating Restrictions, Running Multiple Populations in Tandem such as the Island Model EAs, Spatial Distribution within One Population such as Cellular EAs, etc.).
i) Incorporate a functional user interface demonstrating the algorithm, parameters, inputs, and results.
j) The students may be awarded bonus marks in the following cases (only if the experiments are carried out properly and the results/performance were measured, reported, and analysed adequately:
   - [6 marks] Investigating the effect of multiple (at least 2) representations (when possible).
   - [4 marks] Investigating the effect of multiple (at least 2) initialization approaches (when possible).
   - [4 marks] Investigating the effect of over-selection for large populations (when possible).
   - [6 marks] An educational visual interface that illustrates (simulates) the changes in the evolutionary process and solutions when different parameters/options/approaches are selected. I.e., an interface/simulation that can be later used to teach students the effects of varying selected approaches/parameters, .. etc.
k) Report the results (independently) for each setting (e.g., a choice of a mutation operator, a recombination operator, a representation, a parent selection approach, a survivor selection approach, an initialisation, an approach for preserving diversity, .. etc.).
   - The evolution should be carried out multiple times (optimally, 30 runs per setting).
   - The list of seeds (used to initialise the random number generator before each run) should be stored & provided.

[1] Solve the following Large-Scale Route Optimization Problem using an Evolutionary Algorithm of your choice.

Source: Route Optimization - A Real World Scenario: https://www.kaggle.com/datasets/mexwell/large-scale-route-optimization

The example demonstrated in this dataset is inspired by a real-world optimization scenario. The customer is a manufacturing company. They have warehouses in different locations. When they receive orders from their clients, a planner needs to plan the item-to-truck assignment for order delivery. The planner also needs to decide the route of each delivery truck, namely, the sequence of the stops to deliver orders to different destinations. A delivery assignment has its associated cost determined by the type of the assigned delivery truck and the corresponding travelling distance. The optimization objective here is to minimize the overall delivery cost.

This is a variant of the vehicle routing problem (VRP). The constraints modelled are:

- There are different types of trucks from which we can choose. A truck has a capacity limit on both area and weight. (We assume that there is no limit on the number of trucks for each type)

- An item is only available by a specific time. A truck can start only when all items assigned to it are available.

- The available time difference between the earliest and last available items in the same truck should be less than a user-defined limit (e.g., 4 hours).

- All items need to be delivered to their destinations before their deadlines.

- Depending on their properties, some products can be put in the same truck, but some cannot.

- A truck can have N stops at most, where N is a user-defined number.

- A truck must stay at each stop for M hours to unload the items, where M is a user-defined number. Each stop will incur a fixed cost in addition to the delivery cost.

[2] Genetic Algorithms and Differential Evolution for Function Optimisation.

Implement (and compare) – independently – a Genetic Algorithm and a Differential Evolution Algorithm to find the global minimum of 5 benchmark optimisation functions. You may select any 5 benchmark optimisation functions from the following list: https://www.sfu.ca/~ssurjano/optimization.html

[3] Swarm Intelligence for Function Optimisation.

Implement (and compare) – independently – two Swarm Intelligence algorithms to find the global minimum of 3 benchmark optimisation functions. You may select any 3 benchmark optimisation functions from the following list: https://www.sfu.ca/~ssurjano/optimization.html

Swarm Intelligence algorithms include:

- Particle Swarm Optimization (PSO)

- Ant Colony Optimization (ACO)

- Artificial Bee Colony (ABC)

- Artificial Fish Swarm (AFS)

- Bacterial Foraging Optimization (BFO)

- Grey Wolf Optimization (GWO)

- etc. ..

[4] Coevolving Intelligent ANN-based Checkers Players

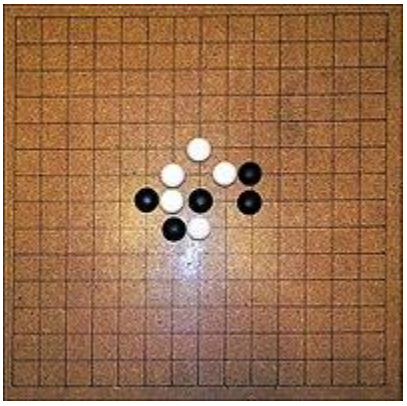
Use coevolutionary systems (genetic algorithms or any other Evolutionary algorithm of your choice) to evolve a neural network-based Checkers player. The aim is to evolve an artificial neural network that can win or draw in a game of checkers.

In this two-player game, a standard 8 × 8 squared board is used, and each player has an (initially fixed) number of pieces (checkers), which move diagonally on the board. A checker can 'take' an opponent's piece if it is adjacent, and the checker jumps over it into an empty square (both players use the same-coloured squares on the board). If a checker reaches the opponent's home side, it becomes a 'king,' which means it can move backwards and forwards.

[5] Coevolving Intelligent ANN-based Gomoku Players

Use coevolutionary systems (genetic algorithms or any other Evolutionary algorithm of your choice) to evolve a neural network-based Gomoku player. The aim is to evolve an artificial neural network that can win or draw in a game of Gomoku.

Gomoku, also known as Five in a Row, is a classic board game where two players take turns placing their markers on a (usually 15x15) grid. The objective is to be the first to achieve a continuous sequence of five markers either horizontally, vertically, or diagonally.



Game Rules:

> ***Board****: Gomoku is typically played on a 15×15 grid, though variations with different board sizes exist.*
>
> ***Players****: The game is played between two players; Black plays first.*
>
> ***Turns****: Players take turns placing their markers (usually black and white stones or "X" and "O") on the vacant intersections of an empty grid.*
>
> ***Objective****: The first player to achieve an unbroken row of five symbols wins. The row can be horizontal, vertical, or diagonal.*
>
> ***Blocked Board****: The game is a draw if the board is filled without a winner.*

In some rules, this line must be exactly five stones long; six or more stones in a row do not count as a win and are called overlines.

[6] Particle Swarm Optimization (PSO) and Simulated Annealing for the Traveling Salesperson Problem (TSP)

Source: Traveling Salesman Problem - Random coordinates to solve TSP:
https://www.kaggle.com/datasets/mexwell/traveling-salesman-problem

Implement (and compare) – independently – a Particle Swarm Optimization (PSO) algorithm and a Simulated Annealing algorithm for the Travelling Salesman Problem (TSP). Use the Large Dataset (containing 1,000 cities) provided at the following link: https://github.com/acu192/fun-tsp-challenge/blob/master/README.md

[7] Particle Swarm Optimization (PSO) for University Timetable Scheduling

Implement a Particle Swarm Optimization (PSO) algorithm to solve the University Timetable scheduling problem (preferably for our faculty). Timetable scheduling is a complex problem and belongs to a class of NP-hard computational problems. This means it isn't possible to generate a solution in polynomial time, but it is possible to identify a correct solution. Additionally, timetable scheduling is a constraint satisfaction problem where one needs to identify a valid combination of events and resources based on various constraints. For example, one needs to identify a valid combination of a classroom, faculty, time, and batch while ensuring constraints like a given classroom and a given batch are free at that particular time. A possible definition for the problem is: Given a set of lecturers, a set of courses on individual topics and a Course Requirements matrix with integer elements representing the number of hours a lecturer teaches a course during each week, the problem is to allocate times to these hours so that a student may take as many suitable combinations of courses as possible. Alternatively, to create a practical timetable for the whole faculty, courses offered by different departments may be combined in various ways to suit individual students.

[8] NeuroEvolution: A Differential Evolution-based Optimizer for Neural Networks

The idea is to apply Differential Evolution to optimise NN's weights while considering the network's structure. You may test the Differential Evolution Neural Network on various classification datasets, for instance, from the UCI repositories (https://archive.ics.uci.edu/ml/datasets) (e.g., MAGIC, QSAR, and GASS) and also on the well-known MNIST (http://yann.lecun.com/exdb/mnist/) dataset for hand-written digit classification.