Artificial Intelligence - Project 1 Checkpoint

Optimization Methods/Meta-heuristics
Solving Google Hashcode 2021: Traffic Signaling

Given the description of a city plan and planned paths for all cars in that city, optimize the schedule of traffic lights to minimize the total amount of time spent in traffic, and help as many cars as possible reach their destination before a given deadline.

Work References

- Hash Code 2021
- Sagishporer - Hashcode 2021 Qualification
- A Comparison of a Genetic Algorithm and Simulated Annealing Applied to a Traffic Light Control Problem
- Hashcode 2021: A lucky ride
- Genetic Algorithms - Crossover

Problem Formulation

Given Input: Duration of the simulation; number of intersections; number of streets and, for each one, its name, time taken to cross it and identifiers to the intersections at the start and end of the street; number of cars and, for each one, the name of the streets it will traverse; the number of points given for each car reaching its destination before the simulation ends.

Solution Representation: Number of intersections and, for each one, its ID, the number of incoming streets covered by the scheduler and, for each street, the order and duration of the green lights.

Hard Constraint: The cars must reach their destination before the simulation ends. Otherwise, they're not accounted for in the solution value.

Evaluation Function: A value is given for each car that finishes its path before the end of the simulation. For a car that finishes its path at time T, it's given a value of X points (given by input) and an additionally (D - T) points, where D is the duration of the simulation.

Neighborhood/Mutation and Crossover Functions

Possible functions for neighborhoods and mutations:

- For each intersection and pair of green lights, swap their order.
- For each intersection and green light, randomly switch their order in the intersection.
- For each intersection and light, randomly increase/decrease its duration (if it becomes 0, the light will always be red).

Possible crossover functions:

- Uniform crossover, using a probability function to choose between Parent 1 and Parent 2 in each intersection.
- Order-based crossover, choosing half of each intersection’s lights of Parent 1 randomly, and filling the other half with the Parent 2’s lights in order.

Implementation Work

- Programming Language: Python
- Development Environment: PyCharm / VSCode for development; Git and GitHub for version control.
- Data Structures:
    - Classes:
        - Car with id and array of streets
        - Intersection with id, outgoing streets, incoming streets and cars that drive through the intersection
        - Street with id, name, starting intersection, ending intersection and time to cross the street.