OrieenteringGA: a genetic algorithm for Orieentering

Esposito Giorgio Angelo
Student ID: 664640
Artificial Intelligence Fundamentals
University of Pisa - Computer Science Department
GitHub repository of the project

Contents

Abstract

This document illustrates the work done for the Artificial Intelligence Fundaments course. A genetic algorithm has been implemented to optimize routes for the sport of orienteering. The objective of this project is to identify the most efficient path for an orienteer to traverse multiple checkpoints (control points), taking into account obstacles and distance metrics. The objective is to investigate the advantages of evolutionary computing techniques in enhancing orienteering strategies using genetic algorithms.

1 Introduction

This section provides a brief explanation of what orienteering is and how genetic algorithms work.

1.1 Orieentirng

Orienteering is a sport where participants navigate through a specific number of checkpoints, referred to as *control points*, in the shortest possible time. Typically, the sport is performed in an unfamiliar terrain. Orienteers rely on a topographical map and a compass to navigate and locate control points.

1.2 Genetic algorithms

Genetic algorithms are a type of algorithm used to solve optimization problems, which are inspired by the process of natural selection. These algorithms fall within the larger category of evolutionary algorithms. Genetic algorithms mimic the principles of biological evolution to iteratively improve and find the best possible solution to an optimization problem, starting with an initial population of solutions. The process starts with evaluating each solution's fitness, which represents its effectiveness in the given problem domain. Solutions with higher fitness have a greater chance of surviving and passing their traits on to the next generation. In the selection stage, solutions are chosen based on their fitness to create a mating pool. Solutions with a higher level of fitness have an increased probability of being selected, which simulates the natural selection process, where fitter individuals are more likely to reproduce.

Following the selection process, the chosen solutions undergo genetic operations in the reproduction phase, including crossover (also referred to as recombination) and mutation.

Two parent solutions swap genetic information to produce offspring in a crossover, incorporating each other's attributes. Mutation introduces unexpected alterations to the offspring's genetic makeup, thus exploring new aspects of the problem domain.
Following the creation of the new generation of offspring, the fitness evaluation process, as well as selection, crossover, and mutation, continue for a specific number of generations or until a termination criterion has been satisfied. As the process repeats, the population evolves, and solutions tend to enhance in fitness, resulting in convergence towards improved solutions.

1.3 Releted works

Although there is no prior work on the specific problem discussed here (while a related, constrained problem exists called the 'Orienteering problem', it is distinct from the one discussed here), there are several challenges associated with searching for optimal paths using genetic algorithms. In Leigh et al. (2007) authors evaluated the performance of their approach on a 3D naval real-time strategy game named Lagoon. The researchers discovered that the genetic algorithm was capable of identifying nearly optimum routes in Lagoon, despite the inclusion of numerous agents and large maps. Moreover, the routes identified by the genetic algorithm were more credible than those identified by A*, as they factored in the environmental attributes such as agent speed and obstacles in the path.
Machado et al. (2011) proposes a method to optimize the process of finding paths for real-time systems, such as video games and virtual reality environments, by using a model based on genetic algorithms and A* algorithm. To achieve this, the proposed solution utilises obstacle pattern detection based on an online training system which is commonly used in real-time systems and dynamic environments. The proposed architecture, named Real Time Pathfinding with Genetic Algorithm (RTP-GA), employs Genetic Algorithm to create an agent that is able to optimize the search for paths even in the presence of obstacles and is adaptable to its environment. In specific cases, the RTP-GA architecture offers better complexity than the A* algorithm.

2 OrieentirngGA

This section explains how the proposed algorithm operates.

2.1 Individual and Map

The first topic to address is the concept of an individual within this context and the 'environment' in which they operate. Ideas for them are taken from this GitHub repository

2.1.1 Map

The first step in this project was to define the type of 'environment' in which the race takes place. A simple solution was employed: a matrix of size mn was created with a specific number of obstacles and control points. Anything that is not designated as an obstacle is 'walkable', allowing the individual to move freely in those areas. Obstacles, on the other hand, must be avoided. Very similar to a *maze*. In figure 1 an example of a map.

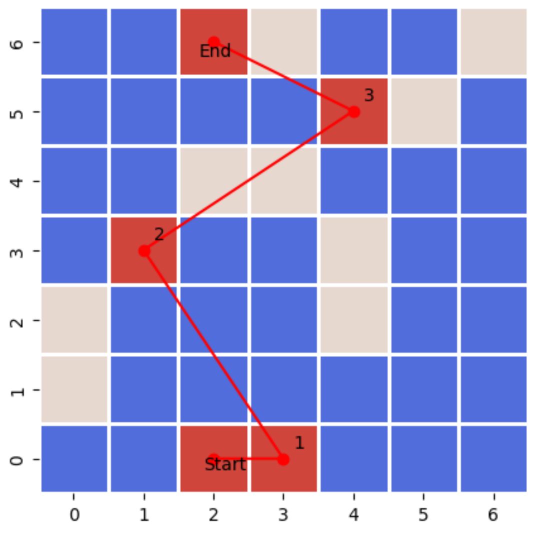

Figure 1: A 7x7 map, with 3 control points. Walkable ground is in blue, obstacles are in grey and starting point, control points and arrival point in red.

2.1.2 Individual

A genetic algorithm implementation requires defining the solution type for problem-solving. In this situation, the objective is to discover the optimal route that begins and ends at specific points while passing through all the control points. However, defining what constitutes the *best* path is necessary. In orienteering, participants must complete the route in the shortest time possible. As our 'environment' is represented as a matrix, the individual will be a list of movements on the map. We make an individual move over the map and store the movement every time it moves, given the available moves to go South, East, West, and North represented by numbers (0, 1, 2, and 3). Every individual has a compass to orient themselves towards the current control point.

2.2 Proposed genetic algorithm

This section outlines the processes involved in creating the initial population, the selection algorithm employed, and the reproduction and mutation processes.

2.2.1 Initial population

The initial population is created through the following process: the individual is allowed to move freely on the map from the starting point, with the direction selected at random. However, this method does not guarantee that a control point will be located. To address this issue, every third step, the individual will use the available compass to move one step closer to the control point that it needs to reach. Furthermore, the individual will avoid exiting the map and encountering obstacles during the exploration. Once the process is complete, we have individuals that have reached all the control points and arrived at the endpoint. Subsequently, we compute the fitness for each individual as the ratio between the number of steps taken by the individual and the optimal number of steps between control points.

2.2.2 Selection

The selection algorithm is used to choose individuals from the present population for the creation of the next generation. In this scenario, the algorithm used is the Roulette Wheel Selection: each individual is assigned to a specific section of a roulette wheel proportionally to their fitness. Individuals with lower fitness possess larger sections on the wheel, making them more likely to be selected for reproduction.

2.2.3 Reproduction

Reproduction is employed to generate a new individual. In this scenario, the new individual is created as a combination of the paths taken by both parents. The path of the first parent is chosen until a particular control point, while the path of the second parent is selected from that control point to the destination. The two paths are merged, creating the new individual's path.

2.2.4 Mutation

The mutation process applied was the swap method: Two control points were randomly selected, and the path between them was obtained, then two moves in that path were randomly selected and swapped.

2.2.5 Next generation production

For a given number of iterations, the process of selection, reproduction and mutation will be repeated to produce a new population.

2.2.6 Wrap up

The algorithm is summarised as follows: beginning with an initial population (2.2.1), we execute selection (2.2.2), reproduction (2.2.3) and mutation (2.2.4) for a specific number of iterations to obtain new individuals and generate a new population (2.2.5). The algorithm terminates when the population size drops below five individuals. Finally, the individual with the least fitness is selected as the optimal solution.

3 Improvments

Listed below are some improvements that could be made to the project.

- **Terrain types**: The map construction considers all cells where movement is possible uniformly. An improvement could be to categorise the terrains into different types such as muddy or sandy, which would decrease the speed of the orienteers, or asphalt, which would increase it. Naturally, the terrain will not be assigned randomly, but rather in a uniform manner such that adjacent cells in the matrix will be of the same terrain type.
- **Elevation**: as previously mentioned about map construction, the terrain is currently at a common level: adding elevation will make the orieenters move faster or slower.
- **Reinforcment Learning**: Reinforcement Learning techniques, such as policy iteration, can be utilised to help the agent navigate the map more efficiently, avoiding obstacles and maximising the number of control points visited. Furthermore, we can ensure that it learns the ups and downs of different types of terrain (if implemented), so that it learns to avoid some and continue over others.
- **Other types of selection and mutation**: The proposed genetic algorithm uses only one type of available methods for selection and reproduction. The utilization of alternative methods may lead to distinct outcomes. Additionally, introducing the concept of *elitism* could potentially yield divergent results.

4 Conclusion

A genetic algorithm was implemented in this project to solve the orienteering game. The problem statement includes a game map and the individual which attempts to solve it. Ultimately, the most optimal solution discovered by the algorithm is a path that passes

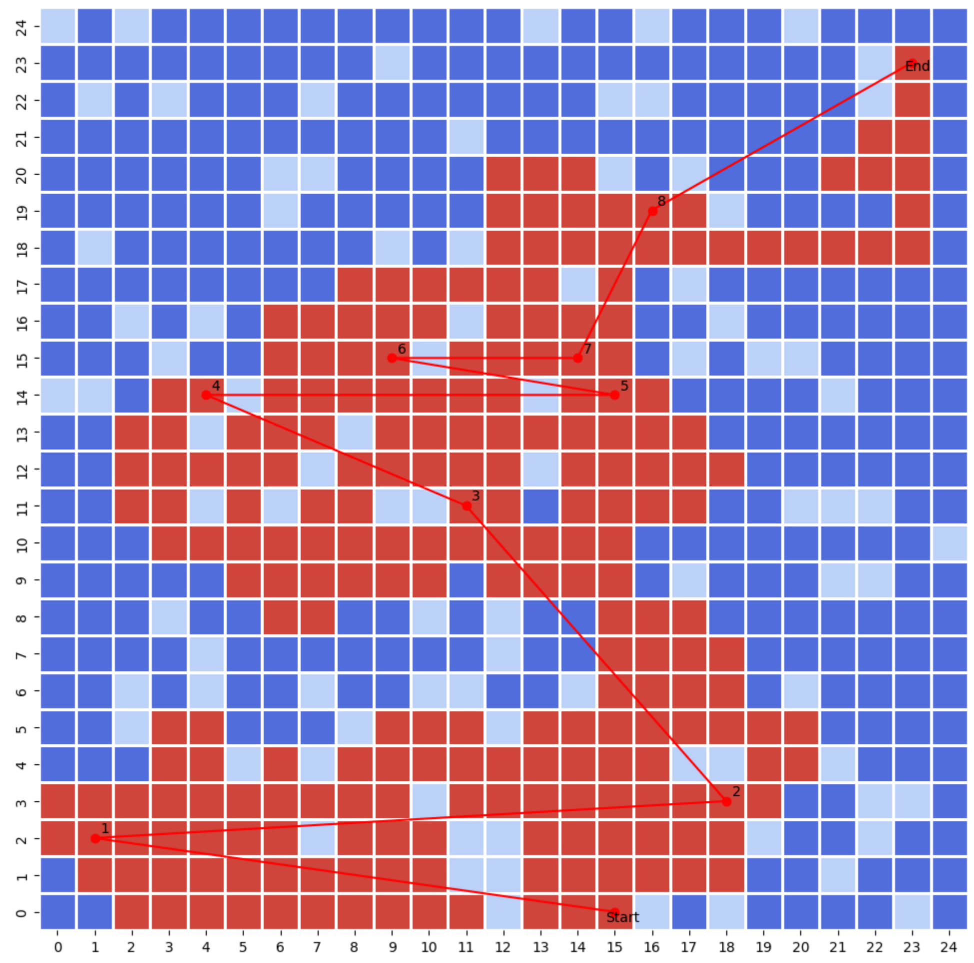

Figure 2: An example of best individual. Initial population: 100.000, number of iterations: 10.000, mutation rate: 0.3

through all control points and reaches the end point. In figure 2 an example of the best individual at the end of the execution.

References

Ryan Leigh, Sushil J Louis, and Chris Miles. Using a genetic algorithm to explore a*-like pathfinding algorithms. In *2007 IEEE Symposium on Computational Intelligence and Games*, pages 72–79. IEEE, 2007.

Alex Fernandes da V Machado, Ulysses O Santos, Higo Vale, Rubens Gonçalvez, Tiago Neves, Luiz Satoru Ochi, and Esteban W Gonzalez Clua. Real time pathfinding with genetic algorithm. In *2011 Brazilian Symposium on Games and Digital Entertainment*, pages 215–221. IEEE, 2011.