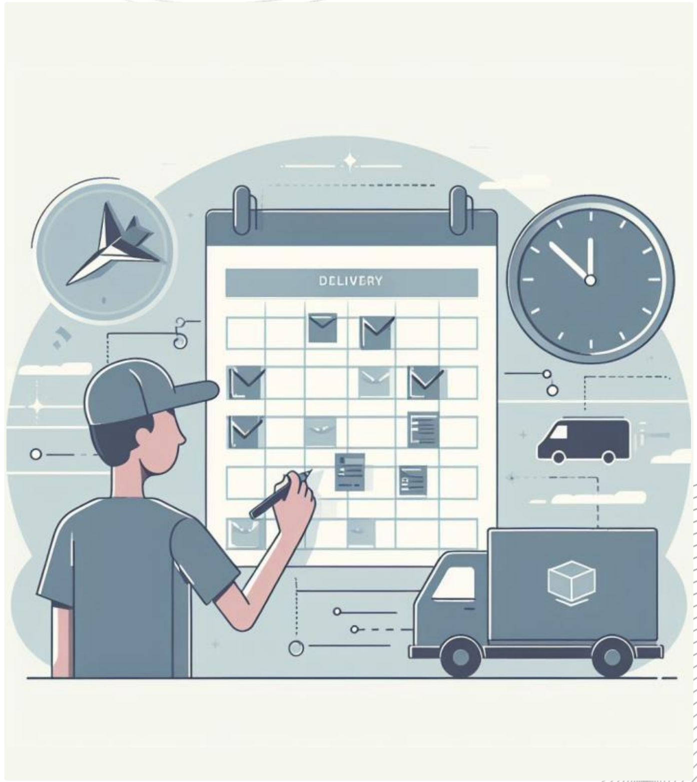
Delivery Scheduling

Artificial Intelligence | T11 | A1_110

Adriano Machado – 202105352
Diogo Fernandes – 202108752
João Torre Pereira - 202108848



The Problem

Design algorithms to optimize the delivery order of packages, considering minimizing fragile damage, travel costs, and adhering to urgent delivery constraints.
The full description of the problem can be found here.

Key Components:

- **Package Types:** Fragile packages have a chance of damage during transportation, normal packages incur no risk, and urgent packages face penalties for delivery delays.
- **Cost Factors:** Costs include travel costs per kilometer and penalties for damaged packages or late deliveries.
- **Constraints:** Limited availability of one vehicle, fixed travel speed and the specified delivery locations.

Related work

In our exploration of optimization algorithms for the delivery scheduling problem, we drew inspiration from various sources:

1. **Travelling Salesman Problem Using Simulated Annealing**: This Medium article by Francis Allanah elucidated the application of simulated annealing to the Travelling Salesman Problem, offering insights into heuristic approaches for optimization. (Medium Article)

2. **Genetic Algorithm Explained**: An article authored by Anas Brital provided a comprehensive explanation of genetic algorithms, which are relevant for optimization tasks involving evolutionary principles. (Medium Article)

3. **Tabu Search Method for Solving the Traveling Salesman Problem**: We referred to a research paper that explores the application of tabu search method for solving the Travelling Salesman Problem, offering insights into advanced optimization techniques. (Paper)

4. **Hill Climbing Algorithm & Artificial Intelligence - Computerphile**: We also benefited from a video by Computerphile, which explains the hill climbing algorithm and its relevance in the domain of artificial intelligence. (YouTube Video)

Problem Formulation

1. **Solution representation**: The problem is represented as a state containing the current delivery order(package stream) among with other relevant variables as broken packages, etc

2. **Neighborhood/mutation and crossover functions**:

   Hill Climbing / Simulated Annealing / Tabu Search - Our definition of neighborhood involves swapping the positions of two randomly selected packages in the delivery order.

   Genetic Algorithms – A new child is created by cross overing two selected parents by three different ways: one-point, multi-point and uniform. In every aproach we priorize the order of the first parent for the selected segments and ended by filling the not used packages in the order of the second parent. Our mutation function, which runs with low probability, involves randomly swapping the positions of two packages of the order.

3. **Hard constraints**:
   - Only one vehicle is available for delivery.
   - The vehicle drives at a constant velocity and takes no time to deliver goods.
   - Each package must be delivered exactly once.
   - Fragile packages have a chance of damage during transportation.
   - Urgent packages incur penalties for delivery outside the expected time.

4. Evaluation function: The evaluation function calculates the total cost associated with a delivery order. It considers the travel costs (fixed cost per kilometer), fragile damage probability and urgent delivery penalties **(distance_cost + state.broken_packages_cost + urgent_cost)**.

Work overview

Programming language

We chose Julia for its exceptional performance and expressive syntax, but also for the challenge of learning a new language.

Interaction

We developed a UI control panel to allow the execution and comparation of the different algorithms. It is also used for modifying the algorithms parameters

Main data structures

```
struct Package
    id::Int
    type::String
    coordinates_x::Float64
    coordinates_y::Float64

    breaking_chance::Union{Float64, Nothing}
    breaking_cost::Union{Float64, Nothing}
    delivery_time::Union{Float64, Nothing}
end

mutable struct State
    packages_stream::Array{Package, 1}

    total_distance::Float64
    total_time::Float64
    total_late_minutes::Float64

    broken_packages::Array{Package, 1}
    broken_packages_cost::Float64

    veichle_velocity::Int64
end
```

Approach & Implemented Algorithms

1. **Hill Climbing**

   This algorithm iteratively adjusts the order of packages by swapping the positions of two randomly selected packages if it leads to an overall better fitness. Its main advantage is the simplistic implementation and fast performance, however it can stagnate in local maximum.

2. **Simulated Annealing**

   Similar to the Hill Climbing algorithm, however it introduces a probabilistic acceptance of worse solutions, which helps prevent getting stuck in local optima. This probability is determined by two main factors: the current temperature and the difference in fitness between the current and potential solutions. As the algorithm progresses, the temperature gradually decreases, allowing for a more stringent selection process that increasingly favors better solutions.

Approach & Implemented Algorithms

3. Tabu Search

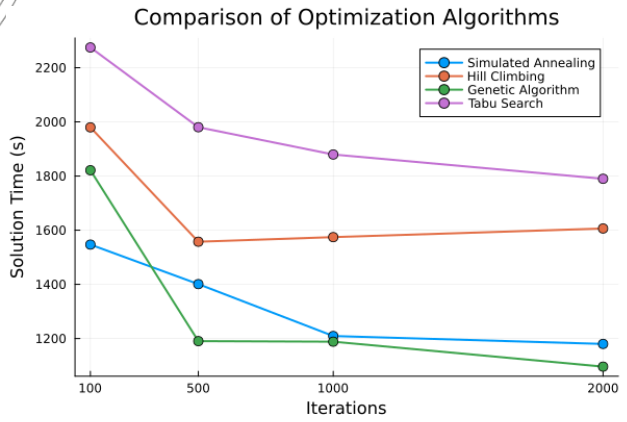
Begins with a given solution and iteratively explores neighboring solutions while avoiding revisiting recent ones, thus preventing cycling and promoting diversification in the search process.

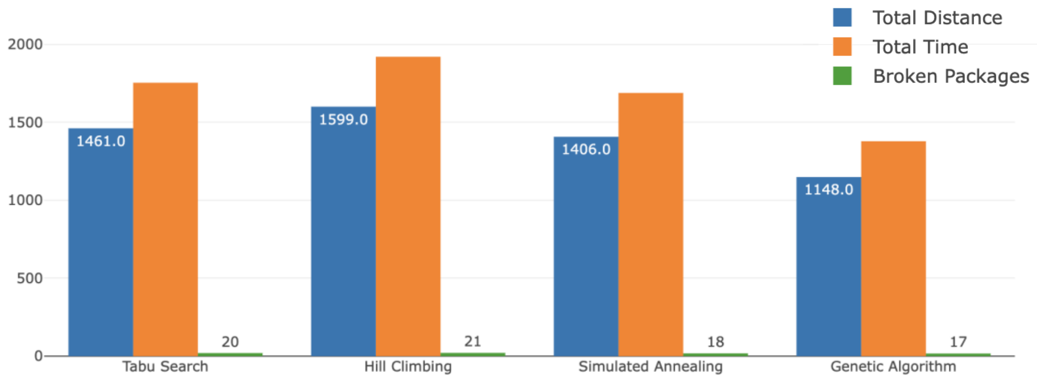
4. Genetic Algorithm

By utilizing principles inspired by natural selection (selection, crossover, and mutation) we start by evolving a random population of potential solutions towards better fitness, aiming to efficiently explore the solution space and find near-optimal solutions. The packages order was defined as our individual DNA. We use elitism selection to select the best parents from the previous generation without changing them, and the rest of the children are reproduced by one of the three different types of crossover operations (one point, multi point, uniform) of two parents selected by one of three different types of implemented selections algorithms (tournament, roulette wheel, rank-based).

Experimental results

A vehicle traveling at the speed limit, 50km/h, delivering 100 packages with a fixed number of 1000 iterations in a map of 60x60.





By tunning some of the parameters to improve each algorithm performance we can finally analyze that the worst algorithm is the Hill Climbing and the Genetic is mostly likely to produce the best results. The Simulated Annealing consistently produces good results among with the tabu search. As for the broken packages, every algorithm performances the same.

Conclusions

Throughout our project, we tried out various methods, testing different heuristics, algorithms, parameters, and neighborhood definitions to see what worked best. We successfully saw the advantages and limitations of each algorithm by analysing our interesting results.

While using Genetic algorithms we conclude the tournament selection was the best selection implementation both in performance aspects as also with the best results. One aspect we can't really explain is mutation rate, low mutation rate don't give us so much results as if we increase it to 20% for example, while the other variables stay the same.
Also, in this type of algorithm we needed to have attention to our definition of the genotype which is the same as the phenotype, both the packages delivery order. This means that the encoding of the individual is the same as its visualization. This leads us to implement strategies for not repeat or lose packages during the crossover operations.

This project has provided us with valuable insights into the creative process involved in basic artificial intelligence projects.

References

Computerphile. (2016, June 21). Hill Climbing Algorithm & Artificial Intelligence [Video].

ChallengingLuck. (2019, August 7). Simulated Annealing Explained By Solving Sudoku - Artificial Intelligence [Video].

Alkallak, I. N., & Sha'ban, R. Z. (2008). Tabu Search Method for Solving the Traveling Salesman Problem. Retrieved from https://www.iasj.net/iasj/download/c8717a7cde6ba605

Brital, A. (2021). Genetic Algorithm Explained. Retrieved from https://medium.com/@AnasBrital98/genetic-algorithm-explained-76dfbc5de85d

Allanah, F. (2022). Travelling Salesman Problem Using Simulated Annealing. Retrieved from https://medium.com/@francis.allanah/travelling-salesman-problem-using-simulated-annealing-f547a71ab3c6