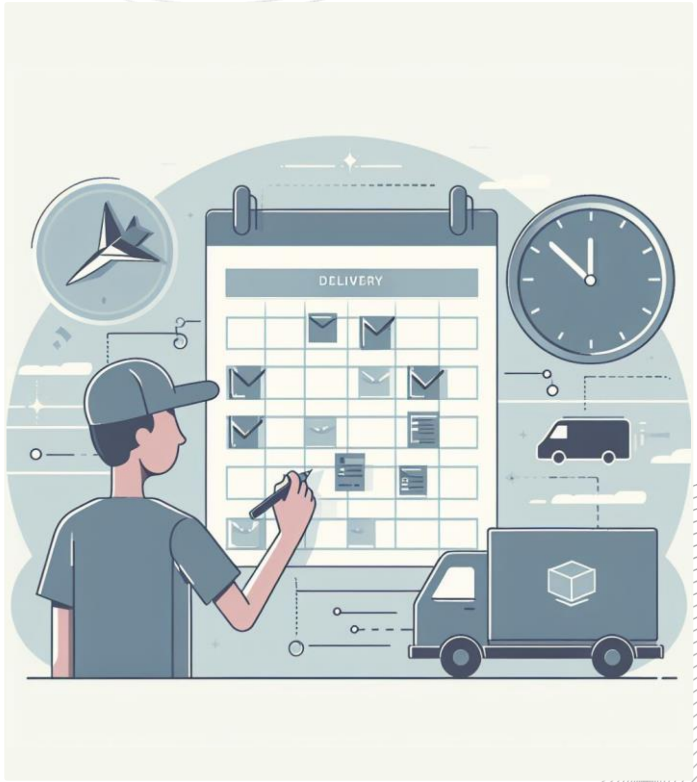
Delivery Scheduling

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).

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

Hill Climbing / Simulated Annealing - The neighborhood function involves swapping the positions of two randomly selected packages in the delivery order.

Genetic Algorithms - Various crossover functions (one-point, multi-point and uniform) are used to create new delivery orders by combining packages from two parents. Mutation involves randomly swapping the positions of two packages in an individual.

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

Work overview

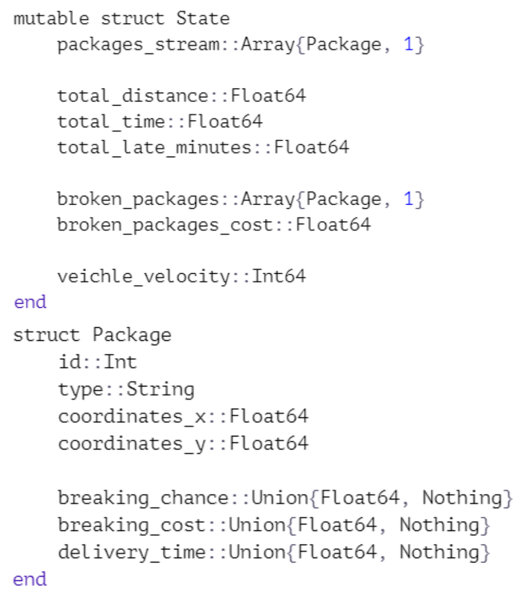
Programming language

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

Work developed

- Hill Climbing, simulated annealing, tabu search and genetic algorithms
- We developed a controll pannel that allows the execution and comparation of the different algorithms

Main data structures



```
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
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
```

The approach

1. **Evaluation function**
   The evaluation function, fitness(state::State), calculates the overall cost associated with a delivery order. It assesses various factors including travel costs (incurred at a fixed rate per kilometer), likelihood of fragile item damage, and penalties for urgent deliveries (sum of distance_cost, state.broken_packages_cost, and urgent_cost).

2. **Heuristic**
   The heuristic used was the global score(fitness) obtained by the solution.

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.
2. **Simulated Annealing**
   Similar to the Hill Climbing algorithm, but with an added feature that allows accepting moves to lower fitness with a probability that decreases over time (depends on the temperature), allowing exploration of a wider solution space.
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 Algorithmn**
   Utilizes principles inspired by natural selection, including selection, crossover, and mutation, to evolve a population of potential solutions towards better fitness, aiming to efficiently explore the solution space and find optimal or near-optimal solutions.

Experimental results

Conclusions

Throughout our project, we tried out various methods, testing different heuristics, algorithms, parameters, and neighborhood definitions to see what worked best. While our results weren't always perfect, we did see noticeable improvements thanks to our algorithms.

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]. YouTube.
11

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