DELIVERY SCHEDULING

Adriano Machado
Diogo Fernandes
João Torre Pereira

GOAL

Design an algorithm to optimize the delivery order of packages, considering the following criteria:

1. Minimize Fragile Damage
2. Minimize Travel Costs
3. Adhere to Urgent Delivery Constraints

SCENARIO

Consider a delivery scenario where packages of three types - fragile, normal, and urgent - must be transported from a starting point at coordinates (0, 0) to various delivery locations. Each package type incurs different costs and penalties during transportation.

CONSTRAINTS

1. You only have one vehicle available.
2. The delivery locations are specified by their coordinates.
3. Routes between all delivery coordinates are available.
4. The driver drives at 60km per hour and takes 0 seconds to deliver the goods.
5. The cost per km is C=0.3.

RELATED WORK AND REFERENCES

- Hill Climbing [1]
- Simulated Annealing [1] [2] [3]
- Tabu search [1] [2] [3]
- Genetic Algorithm [1]

ACTIONS

- Modify the order of the packages

TRANSITION MODEL

Instantiates a new state with the new package_stream and the vehicle's properties.

GOAL TEST

Check if all packages have been delivered.

STATE REPRESENTATON

The state consists in a list of packages (package_stream) where the first element is the first package to be delivered and so on.

All the necessary information is stored in the state, on is instantiation:

- Total Distance Traveled
- Total Time Traveled
- Total Late Minutes of the urgent packages
- The Broken Packages
- The Total Cost of the Broken Packages

COST FUNCTION

- Cost of traveling between locations (distance * cost per km)
- Delayed delivery of urgent packages penalties
- Cost of packages damage during transportation

```
function fitness(state::State)
    C = 0.3
    distance_cost = C * state.total_distance
    urgent_cost = C * state.total_late_minutes
    return distance_cost + state.broken_packages_cost + urgent_co
end
```

HEURISTICS

SIMULATED ANNEALING

Based on the analogy of the physical process of annealing in this algorithm we start with a random order of the packages and then try to improve it by accepting worse solutions with a probability that decreases over time (Temperature variable). With this aproach we should be able to escape local optima.

HILL CLIMBING

Starting with a random order of the packages, this algorithm intends to improve the state by swapping the order of two random packages, discarding changes that do not improve the state, and accepting changes otherwise.

We swap the packages until we can't improve the state by a given number of iterations.

The algorithm is not guaranteed to find the best solution, but it is guaranteed to find a local optimum.

TABU SEARCH

Tabu search is a metaheuristic search method that enhances the performance of local search by relaxing their main rule: worsening moves are not allowed. Tabu search allows moves that worsen the solution if there is no other option. It also uses a memory structure to store solutions that have been visited, and these are used to avoid revisiting the same solution.

GENETIC ALGORITHM

Inspired by the process of natural selection, we will have the concept of a population of solutions, and we will evolve it by applying genetic operators such as mutation and crossover. In this case, our package_stream can be seen as the genome of the individuals.

DATA STRUCTURES

```julia
mutable struct Veichle
    coordinates_x::Float64
    coordinates_y::Float64
    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}
```

PROGRAMMING LANGUAGE

Julia is a high-level programming language for technical computing. It's syntax is similar to Python making it easy to learn. It is also know for its high-performance (i) and for the vast ammount of compatible libraries. Another strong aspect about Julia lies in its accessible support for parallelism, which significantly enhances its efficiency, especially for optimization problems such as the one we are currently tackling.