Inteligência Artificial - Projeto 1

Delivery Scheduling (3C)

João Guedes (up202108711)
João Sousa (up202106996)
Armando Martins (up201603566)

Specification

O objetivo do Delivery Scheduling é otimizar a entrega de três tipos de encomenda: frágil, normal, e urgente. Cada encomenda tem de ser transportada do ponto de partida (0, 0) e chegar ao destino considerando três critérios:

- A minimização de dano causado às encomendas frágeis;
- A minimização dos custos de viagem (cada quilômetro tem custo fixo);
- Aderir às restrições das encomendas urgentes - existe uma penalização por cada minuto de atraso com um custo fixo.

As encomendas frágeis tem a chance de se danificarem durante o transporte; as encomendas normais não existe risco de dano; e as encomendas urgentes recebem uma penalização se forem entregues com atraso.

Objetivo final: optimizar a ordem de entrega para minimizar o custo total usando diferentes algoritmos.

Related Work

Zhuagenborn, 2021, Huawei Delivery Optimization,
https://github.com/Zhuagenborn/Huawei-Delivery-Optimization/blob/main/Huawei%20Delivery%20Optimization%20Competition.pdf

Baeldung, 2023, Partially Mapped Crossover in Genetic Algorithms,
https://www.baeldung.com/cs/ga-pmx-operator

What tools and techniques optimize delivery routing and scheduling?,
https://www.linkedin.com/advice/3/what-tools-techniques-optimize-delivery-routing#:~:text=Tools%20and%20techniques%20that%20optimize,efficiency%2C%20and%20enhance%20customer%20satisfaction.Message%20@despair

Subham Datta, 2024, Genetic Algorithms: Order One Crossover,
https://www.baeldung.com/cs/ga-order-one-crossover

Hard **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.

Optimization Problem (genetic algorithm)

Solution Representation:

Para representar a solução para este problema vamos usar uma “Chromosome Structure”, sendo neste caso o cromossoma a ordem de entrega de todas as encomendas e cada gene representa uma encomenda (objeto Package).

Path = [Package], onde Package = [(X,Y), type]

Neighborhood/Mutation Functions:

Mutação por troca - consiste na seleção aleatória de dois genes dentro do cromossoma e na subsequente troca das suas posições (simula uma alteração na ordem de entrega que permite explorar outras opções que podem minimizar o custo total). Mutação por inversão - consiste em escolher aleatoriamente um subconjunto do cromossoma e inverter a ordem dos genes (operação semelhante à mutação por troca mas mais “drástica”).

Crossover Functions:

Order Crossover (OX) - Um segmento de um *parent* é copiado para um *child* (descendente), e a informação restante é preenchida a partir do outro *parent*, respeitando a ordem das encomendas.

Partially Mapped Crossover (PMX) - Dois pontos de cruzamento são selecionados, e os segmentos entre eles nos *parents* são trocados para produzir *offspring* (descendência), resolvendo quaisquer conflitos para manter sequências válidas.

Evaluation Function:

A função de avaliação vai medir o quão boa uma solução é dependendo da: distância total percorrida, custo de danos de encomendas frágeis e penalização sobre encomendas urgentes tudo somado num custo total.

CustoTotal = DistanciaTotal*0.3 + ValorDanoTotal + TempoAtrasoTotal*0.3

Work Already Implemented

Programming Language: Python, com código implementado para a visualização de uma interface gráfica para observação das coordenadas (*matplotlib.pyplot*), cálculo do custo total e implementação de um algoritmo greedy.

Development Environment: VSCode, GIT

Data Structure: Package class, Path class (lista de packages que contém as coordenadas e outros atributos e um custo), ...

File Structure: Organização do projeto com o GitHub.