ALGAV

Sprint C

08/01/2023

# Turma: 3DC - 016

Diogo de Sousa Ferreira (1180969)  
Duarte Paulo (1171887)  
Rui Gonçalo Sil Gonçalves (1191831)  
Tiago André Macedo Pinto (1200626)  
Vasco Alexandre Valente Rocha (1161216)

1. **Introdução**

Foi proposta a implementação do Algoritmo Genético tendo em conta o problema de distribuição de entregas usando camiões elétricos que temos vindo a solucionar ao longo dos Sprints.

Foi usado, como base, o código disponibilizado pelos professores no Moodle para a realização deste Sprint.

Disto isto, foram implementados novos predicados e alterados outros para que o problema original fosse solucionado através da utilização do AG tendo em conta os requisitos impostos.

Todas as *User Stories* forma concluídas.

1. **Criação da população inicial do Algoritmo Genético (AG)**

A população inicial é gerada pelo predicado **gera\_populacao**. Este gera a população de forma aleatória, ou seja, a sequência de armazéns é sempre diferente.

Para isto ser possível, o predicado determina o número de entregas que irão ser realizadas determinando de seguida o número de camiões necessários tendo em conta a carga a transportar. É também calculada a distribuição das entregas pelos camiões e, por fim, é gerada uma lista aleatória de rotas.

O resultado final é armazenado na variável Pop que é a geração inicial.

É neste predicado que são incluídas duas heurísticas (**heuristicaDistancia** e **heuristicaMassaTempo**) para obtenção de variadas opções para a obtenção da melhor solução.

Este predicado é chamado no **gera** sendo este último, o que dá início a todo o processo.

gera\_populacao(Pop) :-

populacao(TamPop),

findall(EntregaArm, entrega(\_, 20221205, \_, EntregaArm, \_, \_), ListaArmazens),

**length**(ListaArmazens, NumT),

gera\_populacao(TamPop, ListaArmazens, NumT, Pop).

gera\_populacao(0, \_, \_, []) :- !.

gera\_populacao(TamPop, ListaArmazens, NumT, [Ind|Resto]) :-

TamPop1 is TamPop-1,

gera\_populacao(TamPop1, ListaArmazens, NumT, Resto),

((TamPop1 == 0, heuristicaDistancia(20221205, eTruck01, Ind, \_), !);

((TamPop1 == 1, heuristicaMassaTempo(eTruck01, 20221205, Ind, \_), !);

(gera\_individuo(ListaArmazens, NumT, Ind)))),

not(member(Ind, Resto)).

gera\_populacao(TamPop, ListaArmazens, NumT, L):-

gera\_populacao(TamPop, ListaArmazens, NumT, L).

1. **Aleatoriedade no cruzamento entre indivíduos da população**

De forma a evitar que os cruzamentos sejam em elementos sequenciais, é usado o predicado **random** que garante a aleatoriedade no cruzamento pois são escolhidos aleatoriamente os indivíduos da população e também os pontos de cruzamento.

cruzamento([ ], [ ]).

cruzamento([Ind\*\_], [Ind]).

cruzamento([Ind1\*\_, Ind2\*\_|Resto], [NInd1,NInd2|Resto1]) :-

gerar\_pontos\_cruzamento(P1, P2),

prob\_cruzamento(Pcruz),

random(0.0, 1.0, Pc),

((Pc =< Pcruz, !, cruzar(Ind1,Ind2,P1,P2,NInd1), cruzar(Ind2,Ind1,P1,P2,NInd2));

(NInd1=Ind1, NInd2=Ind2)),

cruzamento(Resto, Resto1).

gerar\_pontos\_cruzamento(P1, P2):-

gerar\_pontos\_cruzamento1(P1, P2).

gerar\_pontos\_cruzamento1(P1,P2):-

entregas(N),

NTemp is N+1,

random(1, NTemp, P11),

random(1, NTemp, P21),

P11\==P21, !,

((P11<P21, !, P1=P11, P2=P21); P1=P21, P2=P11).

gerar\_pontos\_cruzamento1(P1, P2):-

gerar\_pontos\_cruzamento1(P1, P2).

1. **Seleção da nova geração da população**

A nova geração da população é gerada pelo **gera\_geracao**. Depois de feito o cruzamento e a mutação, é feita uma avaliação à população (**avalia\_populacao**) e uma ordenação (**ordena\_populacao**).

Finalmente, são escolhidos os melhores indivíduos através do predicado **obterMelhores** que seleciona os 2 melhores na nova geração para a formação da próxima geração e o resto da população.

É calculada ainda a probabilidade de cada um dos restantes indivíduos passar para a próxima geração através do predicado **probabilidadeRes**.

Desta forma, não consideramos que seja um método **elitista** pois existe a possibilidade de passar para a próxima geração indivíduos que não estão entre os melhores.

obterMelhores([H|NPopOrd], 0, [], [H|NPopOrd]).

obterMelhores([Ind|NPopOrd], P, [Ind|Melhores], Restantes) :-

P1 is P-1,

obterMelhores(NPopOrd, P1, Melhores, Restantes).

probabilidadeRes([],[]) :- !.

probabilidadeRes([Ind\*Tempo|Restantes], [Ind\*Tempo\*Prob|ListaProb]) :-

probabilidadeRes(Restantes, ListaProb),

random(0.0, 1.0, NumAl), Prob is NumAl \* Tempo.

1. **Análise da eficácia**

Para esta análise foram usados os seguintes valores:

- nº de novas gerações: 6  
 - dimensão da população: 8  
 - probabilidade de cruzamento: 50  
 - probabilidade de mutação: 25

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nº de Entregas | Melhor Solução | Melhor Solução/Indivíduo da População Final do AG desenvolvido | Melhor Solução/Indivíduo da População Final do AG base | Valor Médio População Final do AG Desenvolvido | Valor Médio População Final do AG Base |
| 6 | 469.009 | [9,11,1,3,17,8]\*469.009 |  | 522.583 |  |
| 7 | 506.098 | [17,9,11,8,3,1,14]\*506.098 |  | 626.916 |  |
| 8 | 632.855 | [17,11,9,14,12,1,8,3]\*632.855 |  | 732.202 |  |
| 9 | 711.527 | [17,11,9,14,6,12,1,8,3]\*711.527 |  | 830.857 |  |
| 10 | 773.971 | [17,13,11,9,14,6,12,1,8,3]\*773.971 |  | 894.138 |  |
| 11 | 858.956 | [17,2,13,11,9,14,6,12,1,8,3]\*858.956 |  | 1010.847 |  |
| 12 | 891.091 | [17,2,13,11,7,14,6,12,1,8,3,9]\*891.091 |  | 1092.752 |  |

1. **Parametrização da condição de término do AG**

Como condição de término usamos o número de gerações, que é inicialmente definido por nós, o tempo máximo atingido ou a estabilização da geração.

Em relação à estabilização da geração, implica que a nova geração tenha valores similares ao já obtidos anteriormente, sendo assim, chega-se a um ponto que não adianta a continuação.

Para conseguirmos detetar se uma geração estabilizou, temos uma variação do predicado **gera\_geracao** em que a execução só corre quando a variável **GenIguais** tem o mesmo valor da variável **Estabilizacao**. Desta forma, se isso acontecer, podemos garantir que quando o valor de **GenIguais** for igual ao de **Estabilizacao**, a geração está estabilizada. Usamos o valor 1 para a **Estabilizacao** enquanto, o **GenIguais** é um contador que irá verificar quando é que uma geração é igual à anterior.

gera\_geracao(G, \_, Pop, \_, \_, 0, Estabilizacao, Estabilizacao, MViagem, \_, \_) :- !,

Pop = [MViagem | \_],

**write**('Geracao '), **write**(G), **write**(':'), nl, **write**(Pop), nl,

**write**('Geracao Estabilizada.'),nl.

gera\_geracao(N, G, Pop, TempoInicial, TempoMax, 0, GenIguaisAnt, Estabilizacao, MViagem, NumCamioes, EntregasDistribuidas) :-

**write**('Geração '), **write**(N), **write**(':'), nl, **write**(Pop), nl,

random\_permutation(Pop, PopAleatoria),

cruzamento(PopAleatoria, NPop1),

mutacao(NPop1, NPop),

obterPopulacao(NPop, NumCamioes, EntregasDistribuidas, [], PopAtualizada),

avalia\_populacao(PopAtualizada, NPopAv, NumCamioes, EntregasDistribuidas),

append(Pop, NPopAv, Populacao),

**sort**(Populacao, Aux),

ordena\_populacao(Aux, NPopOrd),

obterMelhores(NPopOrd, 2, Melhores, Restantes),

probabilidadeRes(Restantes, ProbRestantes),

ordenaPopProbabilidade(ProbRestantes, ProbRestantesOrd),

populacao(TamPop),

ElementosF is TamPop-2,

retirarExtra(ProbRestantesOrd, ElementosF, ListaMelhores),

append(Melhores, ListaMelhores, ProxGen),

ordena\_populacao(ProxGen, ProxGenOrd),

N1 is N+1,

get\_time(Tf),

TempExec is Tf-TempoInicial,

obterTempoExec(TempExec, TempoMax, FlagFim),

verificarPopEstabilizada(Pop,ProxGenOrd, GenIguaisAnt, GenIguais),

gera\_geracao(N1, G, ProxGenOrd, TempoInicial, TempoMax, FlagFim, GenIguais, Estabilizacao, MViagem, NumCamioes, EntregasDistribuidas).

1. **Estudo de métodos de Aprendizagem Automática/Visão por Computador/Robótica Inteligente ao problema da distribuição de mercadorias e/ou uso de veículos elétricos**

### Introdução

A inteligência artificial e a aprendizagem automática estão continuamente a conquistar novas indústrias e esferas do nosso quotidiano, e a logística não é exceção. A sua utilização possibilita a otimização de processos, evita erros passiveis de serem cometidos por humanos e é capaz de prever futuras oportunidades e desafios.

A dificuldade da gestão de redes de logística tem-se tornado cada vez mais evidente, e quase diariamente são reportados os efeitos detrimentais das falhas na cadeia de abastecimento, causando impactos extremamente nefastos na economia a nível global.

Muitos dos incómodos existentes no setor da logística podem ser resolvidos, ou pelo menos col- matados, com a utilização de *machine* *learning* e *IA*. Apesar de ainda não estarem completamente inseridas no mercado, estas tecnologias estão a permitir que as organizações, pequenas ou grandes, se tornem mais sólidas, trabalhem de forma mais eficiente, minimizem custos e desperdício e, no geral, se torem mais lucrativas.

### Problemas da logística passíveis de serem resolvidos com o uso de Inteligência Artificial

Baseando-nos em artigos disponíveis no *Google Scholar*, selecionamos dois dos principais problemas existentes no setor da logística, que são passíveis de serem resolvidos/colmatados com ML.

#### Incerteza da Procura

Incerteza da procura refere-se à mudança na procura provocada por fatores externos que causam variações inesperadas na procura de um determinado bem ou serviço. A incerteza da procura pode ser categorizada em longo, médio e curto prazo. Já foram testados um conjunto enorme de algoritmos, desde máquinas de estados até ao desenvolvimento de algoritmos novos para a resolução deste problema.

#### Logística Reversa

O conceito de logística reversa inclui o processo de transportar bens, materiais e informação relacionada, desde o ponto de consumo até à sua origem, com o propósito de recapturar valor. Recentemente começou a ser discutida a possibilidade de utilização de *supervized learning* neste tipo de logística, com o uso de redes neurais convolucionais.

#### Gestão de Armazéns

Uma boa gestão de armazéns e inventário é essencial para um planeamento eficiente da cadeia de abastecimento. Quer a falta, quer o excesso de stock podem tornar-se num enorme desafio, até para as melhores estratégias de planeamento. *Machine learning* e a sua capacidade de previsão, podem não só solucionar o problema, como também mudar completamente a gestão de armazéns para melhor, tudo isto se deve à capacidade da inteligência artificial de analisar quantidades enormes de informação e eliminar o erro humano.

#### Otimização de rotas

*IA* pode ser utilizada para reduzir custos e tornar o processo de transporte mais rápido, quando utilizada para escolher as melhores rotas. Isto é especialmente importante no caso de empresas grandes com quantidades avultadas de clientes. Resultando em uma maior satisfação dos clientes e margens de lucro mais elevadas. Os algoritmos de *machine learning* dão melhores resultados que os métodos de otimizaçãao de rotas tradicionais e a presença de uma solução digital torna a gestão de frotas mais inteligente.

### Tipos de Machine Learning

*Machine Learning (ML)* é considerado parte integral da *IA*, e refere-se à deteção automática de padrões significativos em *datasets*. Ferramentas de *ML* tentam aumentar a eficiência de algoritmos, garantindo a capacidade de aprender e adaptar com base na análise de informação. Para além disto, *deep learning (DL)*, é definido como uma subclasse de *ML* que explora diversos níveis de processamento de informação não linear, análise e classificação de padrões. Nos últimos anos, *IA*, *ML* e *DL* têm vindo a ganhar relevância numa multitude de áreas de pesquisa como engenharia, medicina e também logística.

#### Supervised Learning

*Supervised Learning* ocorre quando o modelo aprende a partir de resultados pré-definidos, para determinar quais devem ser os resultados de saída. Estes valores servem como “supervisão” destas previsões, permitindo o ajuste nas previsões com base em erros. Isto é considerado *Supervised Learning*, pois o modelo possui uma referência do que é considerado certo e daquilo que é considerado errado.

Diagram

Description automatically generated

Figura 1 - Supervised Learning

#### Unsupervised Learning

Ao contrário do *Supervised Learning*, no *Unsupervised Learning* não existem resultados pré-definidos para o modelo utilizar como referência. O modelo tenta sozinho encontrar semelhanças e diferenças entre os dados de maneira a categorizar a informação da melhor forma possível. Não existindo uma referência (ou critério específico) para o modelo seguir, define-se como *Unsupervised Learning*.

Chart, diagram

Description automatically generated

Figura - Unsupervised Learning

#### Reinforcement Learning

*Reinforcement learning* é um método que se baseia em ”recompensar” comportamentos desejados e ”castigar” comportamentos indesejados. No geral, o agente de *reinforcement learning* é capaz de estar ciente e interpretar o ambiente no qual está inserido, tomar decisões e aprender através de tentativa erro. Este método atribui valores positivos às ações desejadas e valores negativos às indesejadas, o que programa o agente para procurar, a longo prazo, o valor máximo e obter uma solução ótima.

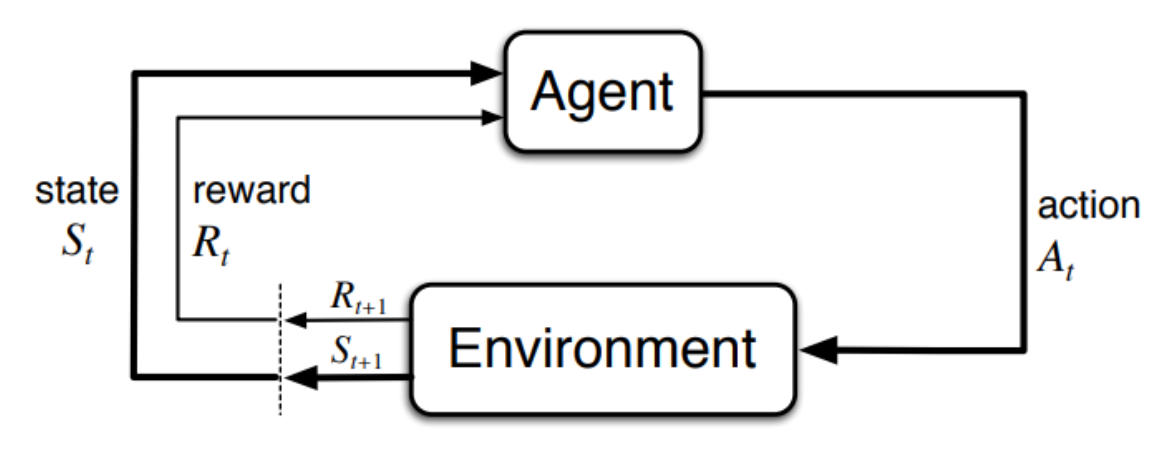


Figura 3 - Reinforcement Learning

#### Métodos

Os métodos podem sem categorizados pela tarefa que resolvem. Por exemplo, classificação e regressão são duas tarefas de *supervised learning*. Um exemplo de uma tarefa referente a *unsupervised learning* pode ser *clustering*. A figura seguinte mostra métodos de *machine learning*, agrupados pela tarefa que comumente resolvem. É importante referir que esta seleção de métodos está longe de estar completa, e que o mesmo método pode ser utilizado para resolver mais do que uma tarefa.

A picture containing graphical user interface

Description automatically generated

Figura 4 - Machine Learning Methods

### Utilização de Machine Learning no problema proposto

Um dos processos chave na logística é a entrega dos produtos que se encontram nos centros de distribuição. Uma entrega livre de erros depende de vários fatores, desde o processo de entrega em si até à escolha dos produtos corretos a enviar, ainda no armazém. De forma a integrar um mecanismo de controlo, o *METRO Group Cash Carry* implementou pórticos com *scanners RFID* que detetam a saída de paletes. As paletes são equipadas com um *transponder*, e assim que passam os pórticos, o dispositivo de leitura deteta a saída dos bens e ajusta o inventário disponível através de comunicação direta com o sistema de gestão do armazém.

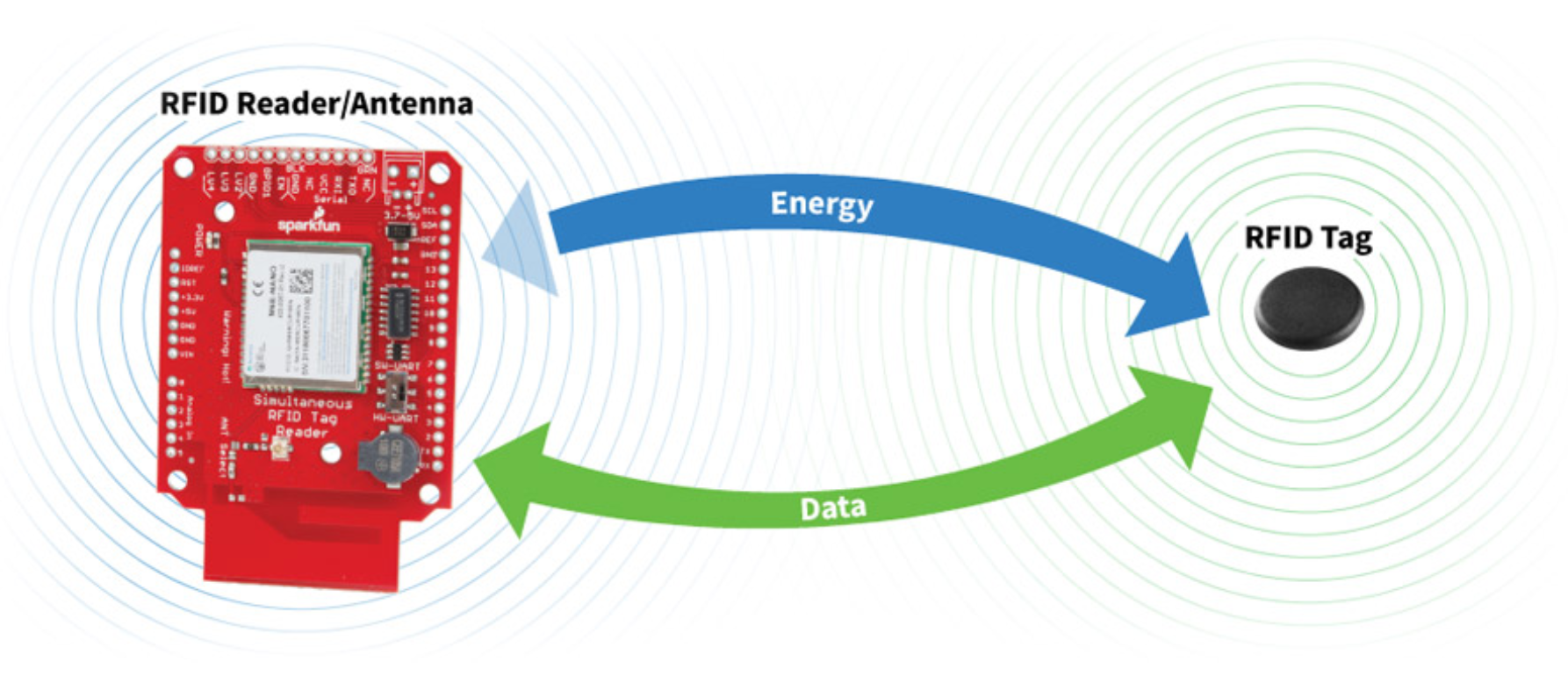


Figura 5 - Exemplo de comunicação RFDI

Esta solução apesar de detetar e ajustar o inventário automaticamente, tem um problema grave. As antenas de leitura usadas nos pórticos possuem um alcance de vários metros, identificando por isso, todas as paletes dentro do seu raio. Isto significa que se torna impossível distinguir entre paletes que de facto foram carregadas para um camião, e paletes que não estão relacionadas com qualquer entrega e foram acidentalmente lidas.

Para evitar estes falsos positivos, podem ser aplicados métodos de *machine learning*, com base em atributos conhecidos de cada palete. Estes atributos podem incluir a força do sinal recebido pelo leitor *RFDI (RSSI)* e o número de leituras por antena. Com isto, torna-se aparente que as paletes que não estão relacionadas com nenhuma entrega submetem valores *RSSI* diferentes, com base nas frequências lidas pelos pórticos. É aqui que o *machine learning* atua. Classificar as duas características de acordo com o seu valor limite é uma tarefa comum de um sistema *ML*. Caso o valor médio *RSSI* seja igual ou superior a -58.1dBm, trata-se de uma palete a ser carregada para entrega.

É importante referir que, para que esta avaliação ocorra por parte do algoritmo *ML*, é necessário um *dataset* de tamanho considerável. Após a aquisição completa dos dados, estes algoritmos conseguem classificar corretamente mais de 95.5 por cento das paletes, identificando assim, facilmente, os falsos positivos.

Diagram

Description automatically generated

Figura 6 - Funcionamento de um algoritmo ML

### Conclusão

É inegável que *machine learning* melhora a gestão da cadeia de abastecimento e a experiência do consumidor. Desde a redução de custos, otimização e planeamento de inventário e assegurar uma entrega livre de erros. As empresas que já adotaram *machine learning* notam melhoras significativas em todas estas áreas, resultando no final, em margens de lucro mais elevadas. *Machine learning* é uma tecnologia extremamente útil para as empresas de logística, pois pode ser aplicado em quase qualquer parte da cadeia de abastecimento. O seu objetivo é resolver problemas mecânicos e repetitivos, reduzindo o erro humano e auxiliando na tomada de decisão, permitindo assim aos trabalhadores que se foquem em trabalhos de maior valor.

1. **Adaptar o plano de modo a considerar as seguintes alterações dinâmicas**

Uma vez implementado o Algoritmo Genético, adaptamos o plano de entrega para realizar as seguintes alterações dinâmicas: o cancelamento de uma entrega, a inclusão de uma nova entrega e a alteração da dimensão de uma entrega. Em primeiro lugar, decidimos implementar um predicado para cada uma das alterações dinâmicas a realizar. O predicado de remover uma lista recebe uma lista, que irá ser removida, do plano de entregas do algoritmo genético.

Uma imagem com texto, captura de ecrã, ecrã

Descrição gerada automaticamente

Foi também implementada uma verificação que foi chamada no Algoritmo Genético para verificar se a variável LR (lista a remover), inicializada com valor 0, se já continha o plano de entrega a remover. Caso ainda não, o Algoritmo Genético continua em execução, ignorando esta validação. Caso LR seja diferente de 0, executa o predicado de “remover\_lista” e remove-a da lista da “Pop”.



Por outro lado, o predicado de inclusão de uma entrega, irá adicionar um plano de entrega extra à “Pop”. Desta forma, com o auxílio do predicado “adiciona\_lista” implementado e com a chamada da validação no Algoritmo Genético, onde LA é a lista de entrega a ser adicionada, mais uma vez definida a 0 e caso LA obtenha o valor da lista, é ativado o predicado de “adiciona\_lista”, concluindo assim, que a lista será adicionada à “Pop” com sucesso.



Uma imagem com texto

Descrição gerada automaticamente

Em suma, o predicado da alteração da dimensão do plano de entrega, foi realizado de forma simples. Em primeiro lugar, assumimos que para aumentar a dimensão do plano, teríamos de adicionar mais um armazém, pelo menos. Desta forma, ao haver mais armazéns por onde serão realizadas as entregas nesse dia, a sua dimensão também aumenta. Implementamos, também, uma validação que é chamada no Algoritmo Genético, e caso o valor de AD seja diferente de 0, usamos um predicado append “append\_aumenta\_dimensao”, onde passamos o AD (variável recebida para o aumento da dimensão dada pelo utilizador. AD é o numero extra do armazém) e onde recebemos a Mviagem (Melhor viagem) onde será adicionado o armazém extra a um determinado plano de entrega e verificar se há diferenças a nível da melhor viagem em relação ao tempo.

Uma imagem com texto

Descrição gerada automaticamente

Uma imagem com texto, laranja, captura de ecrã, definir

Descrição gerada automaticamente

Uma vez concluído, chamamos novamente o Algoritmo Genético e adaptamos de forma a considerar, a solução gerada anteriormente (Mviagem), um dos elementos da população inicial.

1. **Conclusões**

Depois de feita esta análise, é possível concluir que o Algoritmo Genético é uma das melhores opções para a solução do problema apresentado.

Obtendo ainda melhores resultados quando conjugado com as heurísticas já realizadas no Sprint anterior.

Por fim, tendo em conta o número de entregas o tempo aumenta significativamente. Outra alteração feita pelo aumento ou diminuição do número de entregas é a ordem dos armazéns que muda sempre consoante o melhor caminho calculado.

1. **Referências**

Jannicke Baalsrud Haugea Amita Singha, Magnus Wiktorssona. Trends in machine learning to solve problems in logistics. Procedia CIRP, 2021.

Erwin Rauch Manuel Woschank and Helmut Zsifkovits. A review of further directions for artificial intelligence, machine learning, and deep learning in smart logistics. Industry 4.0 for SMEs - Smart Manufacturing and Logistics for SMEs, 2020.

10 use cases of ai and machine learning in logistics and supply chain. https://addepto.com/ blog/use-cases-ai-machine-learning-logistics-supply-chain/#. Published: 2021- 04-18.

9 use cases of machine learning in logistics. https://serengetitech.com/business/ 9-use-cases-of-machine-learning-in-logistics/. Published: 2022-08-26.

How is machine learning enhancing the logistics industry? https://itechdata.ai/ machine-learning-in-logistics-industry/.

Daniel; Sardesai Saskia Wenzel, Hannah; Smit. A literature review on machine learning in supply chain management. International Conference of Logistics, 2019.

Reinforcement learning. https://www.techtarget.com/searchenterpriseai/ definition/reinforcement-learning. Published: 2021-03-01.