Relatório ALGAV

Sprint C

GRUPO 70 TURMA 3DL

Beatriz Cardoso Borges | 1201461

Hugo Henrique Almeida Carvalho | 1210813

Érica Filipa Lopes | 1201474

Guilherme Pacheco da Cunha | 1201506

Jan

Introdução

Os algoritmos genéticos são um tipo de algoritmo que se inspira na teoria da evolução darwiniana para resolver problemas. Eles são usados principalmente para otimização de funções, mas também podem ser usados para encontrar padrões em conjuntos de dados ou para fazer previsões.

Um algoritmo genético funciona da seguinte maneira:

Inicialmente, é criada uma população de soluções candidatas para o problema a ser resolvido. Cada solução é chamada de indivíduo e é representada por uma sequência de bits ou números, chamada de cromossomo.

Cada indivíduo é avaliado com base em uma função de aptidão, que mede o quão bem a solução se adequa ao problema. Os indivíduos mais aptos são selecionados com maior probabilidade para reprodução.

Os indivíduos selecionados são combinados através de operações de recombinação, como cruzamento e mutação, para gerar novos indivíduos.

O processo é repetido por várias gerações, até que uma solução satisfatória seja encontrada ou até que um critério de parada seja atingido.

Os algoritmos genéticos são úteis quando se tem um grande espaço de busca e não se tem uma solução óbvia para o problema. Eles também são robustos e podem encontrar soluções ótimas mesmo em problemas com muitas restrições e variáveis. No entanto, eles podem ser lentos e requerem muitos cálculos, especialmente para problemas de grande escala.

Explicação do algoritmo genético

Utilizou-se o método de recombinação para dar resposta à melhor solução para o problema.

O método configuracao_algoritmo_genetico(NG, DP, P1, P2, TempoExecucao, NGPrev)

começa por inicializar os parâmetros do algoritmo genético que são: NG: número de gerações, DP: dimensão da população, P1: probabilidade de cruzamento, P2: probabilidade de mutação 2, NGPrev: número de gerações prévias.

```
configuracao_algoritmo_genetico(NG, DP, P1, P2, TempoExecucao, NGPrev):-
        (retract(geracoes(_)), !; true), asserta(geracoes(NG)),
        (retract(populacao(_)), !; true), asserta(populacao(DP)),
        PC is P1/100,
        (retract(probabilidade_cruzamento(_)), !; true), asserta(probabilidade_cruzamento(PC)),
        PM is P2/100,
        (retract(probabilidade_mutacao(_));true), asserta(probabilidade_mutacao(PM)),
        (retract(maxTempoExecucao(_)), !; true), asserta(maxTempoExecucao(TempoExecucao)),
        (retract(nGPrev(_)), !; true), asserta(nGPrev(NGPrev)).
```

Fig1-Método de configuração de algoritmo genetico

Segue-se o método algoritmo_genetico(Camiao, ListaEntregas, TempoIdeal,Data).

```
algoritmo_genetico(Camiao, ListaEntregas, TempoIdeal,Data):-
        calculo_carga_total_transportar(ListaEntregas, MassaTotalTransportar),
        calculo_Capacidade_Total_Camioes([Camiao], CapacidadeTotalCamiao),
        MassaTotalTransportar =< CapacidadeTotalCamiao,
        !,
        (retract(tempoIdeal(_)), !; true), asserta(tempoIdeal(TempoIdeal)),
        (retract(dataEntrega(_)), !; true), asserta(dataEntrega(Data)),
        get_time(InitTempoExecucao),
        (retract(initTempoExecucao(_)), !; true), asserta(initTempoExecucao(InitTempoExecucao)),
        convert_entrega_armazens(ListaEntregas, ListaArmazem),
        length(ListaArmazem, NumElem),
        (retract(entregas(_)), !; true), asserta(entregas(NumElem)),
        gera_populacao(Pop, ListaArmazem),
        write('Pop: '),write(Pop),nl,
        avalia_populacao([Camiao], Pop, PopAv),
        write('PopAval: '),write(PopAv),nl,
        ordena_populacao(PopAv,PopOrd),
        geracoes(NG),
        gera_geracao([Camiao],0,NG,PopOrd,[PopOrd]).
```

Fig2-Algoritmo genético

Este método, começa por invocar os métodos calculo_carga_total_transportar(ListaEntregas, MassaTotalTransportar) e calculo_Capacidade_Total_Camioes([H|T], CapacidadeTotalCamiao). O primeiro a ser invocado irá calcular a massa total a transportar que corresponde à soma das massas de cada entrega que faz parte da listaEntregas. Já o segundo método a ser invocado irá

calcular a capacidade total do camião que corresponde à soma da capacidade de carga de cada camião dada uma lista de camiões.

```
calculo_carga_total_transportar([], 0).

calculo_carga_total_transportar([H|T], MassaTotalTransportar):-
        calculo_carga_total_transportar(T, VarTemporaria),
        entrega(H, _, Massa, _, _, _),
        MassaTotalTransportar is Massa + VarTemporaria.

calculo_Capacidade_Total_Camioes([], 0):- !.

calculo_Capacidade_Total_Camioes([H|T], CapacidadeTotalCamioes):-
        calculo_Capacidade_Total_Camioes(T, VarTemporaria),
        carateristicasCam(H, _, CapacidadeCarga, _, _, _),
        CapacidadeTotalCamioes is CapacidadeCarga + VarTemporaria.
```

Fig3-Métodos para calculo ca carga a transportar e calculo da capacidade total do Camiao.

Voltando à Fig2, a condição MassaTotalTransportar =< CapacidadeTotalCamiao implica que se a massa das entregas a serem transportadas forem superiores à capacidade total do camião o método algoritmo_genetico pára e retorna false. Se a condição for verdadeira, procede para as linhas de código seguintes.

De seguida, é feita a inicialização dos parâmetros, tempoIdeal e da data recebidos como parâmetros. Posteriormente através da função do prolog get_time vamos buscar o tempo de execução do método em questão e através do retract, esta variável é inicializada.

O método invocado seguinte, convert_entrega_armazens(ListaEntregas, ListaArmazem) tem o propósito de converter uma lista de entregas numa lista de armazéns, ou seja, para cada entrega, vamos buscar o seu armazém associado e armazenar o mesmo na ListaArmazem.

```
convert_entrega_armazens([], []).

convert_entrega_armazens([H|T1], [Armazem|T2]):-
        entrega(H, _, _, Armazem, _, _),
        convert_entrega_armazens(T1, T2).
```

Fig4-Método que converte entregas em armazéns

Seguidamente, voltando à Fig2, através do length(ListaArmazem, NumElem), iremos armazenar o tamanho da listaArmazem na variável NumElem. No passo seguinte, ao fazermos (retract(entregas(_)), !; true), asserta(entregas(NumElem)), estamos a inicializar a variável entregas com o valor NumElem.

2) Criação Inicial da população do Algoritmo Genético (AG)

De seguida, é invocado o método da Fig5, gera_populacao(Pop, ListaArmazem), este irá armazenar na variavél Pop uma lista que contem as listas de indíviduos da população. Por sua vez, o gera_populacao invoca o método gera_populacao1(TamPop,ListaArmazem,NumElem,Pop).

Este método tem como propósito criar listas de indivíduos.

Sabemos que as listas de indivíduos têm de ser geradas 2 delas com recurso a heurísticas e as restantes terão de ser random.

Sabendo que o TamPop corresponde à dimensão da população, e para termos garantias de que duas listas de indivíduos são criadas com recurso a heurísticas definimos o método gera_populacao1 de duas formas:

gera_populacao1(2,ListaArmazens,_,Lista)

gera_populacao1(TamPop,ListaArmazens,NumElem,[Head|Tail])

```
gera_populacao(Pop,ListaArmazem ):-
        populacao(TamPop),
        length(ListaArmazem, NumElem),
        gera_populacao1(TamPop,ListaArmazem,NumElem,Pop).

%gera_populacao1

gera_populacao1(2,ListaArmazens,_,Lista):-
    bestfsDistance(ListaArmazens, ListaTemp1),
    dataEntrega(Data),
    bestfsWD(ListaArmazens,Data, ListaTemp2),
    idArmazemPrincipal(Armazem),
    removeElementoLista(Armazem,ListaTemp1, ListaTemp3),
    removeElementoLista(Armazem,ListaTemp2, ListaTemp4),
    ((not(igual(ListaTemp3, ListaTemp4)),!,append([ListaTemp3], [ListaTemp4], Lista));
    reverse(ListaTemp4, ListaTemp5),append([ListaTemp4], [ListaTemp5], Lista)).


gera_populacao1(TamPop,ListaArmazens,NumElem,[Head|Tail]):-
        TamPop1 is TamPop-1,
        gera_populacao1(TamPop1,ListaArmazens,NumElem,Tail),
        gera_individuo(ListaArmazens,NumElem,Head),
        not(member(Head,Tail)).
```

Fig5-Método de gera_populacao e gera_populacao1

Como podemos ver na Fig5, se o tamPop tiver valor igual a 5, é o segundo método do gera_populacao1 que será executado. Ao longo deste método, o TamPop será decrementado a cada iteração e o valor resultante será armazenado na variável TamPop1. A menos que este valor possua valor igual a 2 (caso base), serão geradas listas de indivíduos random a cada iteração através do método gera_individuo(ListaArmazem,NumT,[G|Resto]). Quando o valor TamPop1 for igual a 2, o primeiro método do gera_populacao1 passa a ser executado e os indivíduos serão gerados através das heurísticas da distância e da massa. Ao longo deste método, e após serem executados os métodos das heurísticas que retornam o caminho mais

curto entre armazéns, é removido o armazém de Matosinhos dessa lista já que é o armazém de partida e chegada.

Se as listas de indivíduos resultantes das heurísticas forem diferentes fazemos o append dessas mesmas listas e armazenamos numa só lista, do contrário, fazemos primeiro o reverse de uma das listas e só depois fazemos o append de ambas.

4) Seleção da nova geração da população

Voltando à Fig2, após ser gerada então a população com a respetiva lista de indivíduos, é chamado o método avalia_populacao(ListaCamioes, Pop, PopAv). Este método é responsável por calcular o(s) intervalo(s) de um camião ou uma lista de camiões, através do método calculo_intervalo_avaliacao(ListaCamioes, H, 0, ListaInterval) que por sua vez invoca o método calculo_intervalo_avaliacao_camiao(H1, Trajeto, 0, InitIntervalo, FimInterval) que calcula o intervalo de um camiao. De seguida, é chamado o calculo_tempo_rota(ListaCamioes, H, ListaInterval, ListaAvaliacao) que para a lista de que intervalos calculados irá determinar o tempo da rota para camião. e colocar o resultado na variável ListaAvaliacao. De seguida, faz um sort por ordem descendete e na linha seguinte armazena em Avaliacao o primeiro elemento da lista resultante do sort, ou seja, o pior tempo encontrado. Posteriormente, invoca

recursivamente o método avalia_populacao(ListaCamioes, T1, T2).

```
avalia_populacao(_,[],[]):-!.
avalia_populacao(ListaCamioes,[H|T1],[H*Avaliacao|T2]):-
        calculo_intervalo_avaliacao(ListaCamioes, H, 0, ListaInterval),
        calculo_tempo_rota(ListaCamioes, H, ListaInterval, ListaAvaliacao),
        sort(0, @>=, ListaAvaliacao, ListaAvaliacaoOrdenada),
        nth0(0, ListaAvaliacaoOrdenada, Avaliacao),
    !,
        avalia_populacao(ListaCamioes, T1, T2).
```

Fig6-Método avalia população

O método ordena_populacao(PopAv,PopOrd) é responsável por ordenar a população por ordem crescente de avaliação, ou seja, do melhor tempo para o pior.

De seguida, obtemos o número de gerações (NG), através de geracoes(NG), que é usado no método seguinte gera_geracao(listaCamioes,0,NG,PopOrd,[PopOrd]).

6) Parametrização da condição de término do AG

O gera_geracao é responsável por gerar as gerações dada a população, um camião ou lista de camiões, este, começa por verificar se a condição de término não é verificada através do método

verificar_condicao_termino(Pop, GeracoesPrevistas, N, NumGeracoesPrevistas), neste, analisamos a condição do tempo ideal, do tempo de execução e da estabilidade. Para o tempo ideal, comparamos o tempo passado por parâmetro com o calculado (Avaliacao) de modo a verificar se encontrou a solução ótima. No caso do tempo de execução, verifica se o tempo que demora a executar é superior ao dado por parâmetro. Por fim analisa o caso da estabilidade em que verificamos se a população não se alterou durante as gerações dadas.

```
%verificar_condicao_termino
verificar_condicao_termino(Pop, GeracoesPrevistas, N, NumGeracoesPrevistas):-
        verificar_condicao_tempo_ideal(Pop);
        verificar_condicao_tempo_execucao;
        N +1>= NumGeracoesPrevistas,
        verificar_condicao_estabilidade(Pop, GeracoesPrevistas).

%verifica se o tempo Ã© o tempo ideal
verificar_condicao_tempo_ideal(Pop):-
        tempoIdeal(TempoIdeal),
        ind_tempo_ideal(Pop, TempoIdeal).

%verifica se o tempo de execucao Ã© maior que o maximo dado
verificar_condicao_tempo_execucao:-
        get_time(TempoExecucaoAtual),
        initTempoExecucao(InitTempoExecucao),
        maxTempoExecucao(MaxTempoExec),
        InitTempoExecucao - TempoExecucaoAtual >= MaxTempoExec.

%verifica condicao de estabilidade
verificar_condicao_estabilidade(Pop, GeracoesPrevistas):-
        verificar_condicao_estabilidadel(Pop, GeracoesPrevistas).

verificar_condicao_estabilidadel(_, []):-
    !.

verificar_condicao_estabilidadel(Pop, [Pop|Tail]):-
        verificar_condicao_estabilidadel(Pop, Tail).
```

Fig7-Predicados de verificação da condição de término

3) Aleatoriedade no cruzamento de indivíduos da população

Caso não sejam verificadas as condições de término, é feita a permutação da população (Pop) através do random_permutation(Pop, PopPermanente), e de seguida são chamados os predicados cruzamento(PopPermanente,PopCruzamento) e mutacao(PopCruzamento,GeracaoPop) disponibilizados pelos docentes de ALGAV, posteriormente eliminam-se os duplicados através do predicado remove_duplicados(GeracaoPop, Pop, GeracaoPopFinal), de seguida é feita a avaliação da população resultante, juntamos o resultado à população e ordenamos através dos predicados, avalia_populacao(ListaCamioes,GeracaoPopFinal,GeracaoPopAvaliacao), append(GeracaoPopAvaliacao,Pop, PopFinal), ordena_populacao(PopFinal,PopFinalOrdenada), já explicados anteriormente. Seguidamente obtemos 30% do tamanho da população para calcular os melhores indivíduos através do predicado melhores_individuos(0, NumInd, PopFinalOrdenada, ListaMelhores, RestoPop), este método retorna a ListaMelhores preenchida

com os indivíduos da PopFinalOrdenada até ao número indivíduos (NumInd). De seguida são selecionados os indivíduos do RestoPop pelo predicado selecao_individuos(RestoPop, IndSelecionados, TamPop - NumInd), estes são selecionados de forma aleatória e retornados no IndSelecionados. Posteriormente é feita a ordenação da população da junção da ListaMelhores com os IndSelecionados (ordena_populacao(NovaGeracao, NovaGeracaoOrdenada)), e caso N + 1 seja maior ou igual que o número de gerações previstas, usamos o predicado retira e juntamos a lista resultante à nova geração ordenada, de seguida apresentamos os valores e chamamos o predicado em questão recursivamente, enviando-lhe a lista resultante da junção, caso não seja maior, apenas chamamos o predicado recursivamente mas enviando-lhe as lista recebida por parâmetro (GeracoesPrevistas).

Por fim, após todas as gerações serem calculadas, são apresentados, através do rotas_geracao(Camioes, Contagem, [H*_|T]) e rota_camiao([H|T1], Trajeto, [(InitInterval, FimInterval)|T2]), os indivíduos com o(s) camião ou camiões e as suas trajetórias calculadas e explicadas anteriormente.

7) Uso do Algoritmo Genético para lidar com vários camiões

Todo o processo explicado anteriormente foi para o predicado algoritmo_genetico(Camiao, ListaEntregas, TempoIdeal,Data), este algoritmo faz o cálculo recebndo apenas um camião, no entanto para vários camiões temos o método algoritmo_genetico_varios_camioes(ListaCamioes, ListaEntregas, TempoIdeal,Data), cuja diferença é receber uma lista de camiões e também verifica se o número de camiões é suficiente para a massa total que tem a transportar, este calculo é feito pelo verifica_numero_camioes(ListaCamioes, MassaTotalTransportar). De resto este segundo algoritmo não difere em nada comparativamente com o primeiro.

```
?- configuracao_algoritmo_genetico(8, 5, 6, 7, 300, 8).
true.

?- algoritmo_genetico_varios_camioes(['eTruck01','eTruck02','eTruck03'],[443910,443811,444512,444313,445015,443911,443812,444513,444314,444915,445016,443912] ,200,20221205).
Pop: [[1,6,8,12,10,9,7,3,11,4,13,2],[11,10,13,4,9,12,8,3,6,2,1,7],[3,1,10,13,7,11,9,8,4,6,12,2],[8,3,4,13,11,7,1,12,10,2,6,9],[9,6,2,10,12,1,7,11,13,4,3,8]]
PopAval: [[1,6,8,12,10,9,7,3,11,4,13,2]*516,[11,10,13,4,9,12,8,3,6,2,1,7]*423,[3,1,10,13,7,11,9,8,4,6,12,2]*497,[8,3,4,13,11,7,1,12,10,2,6,9]*481,[9,6,2,10,12,1,7,11,13,4,3,8]*455]
Geracao 0:
[[11,10,13,4,9,12,8,3,6,2,1,7]*423,[9,6,2,10,12,1,7,11,13,4,3,8]*455,[8,3,4,13,11,7,1,12,10,2,6,9]*481,[3,1,10,13,7,11,9,8,4,6,12,2]*497,[1,6,8,12,10,9,7,3,11,4,13,2]*516]
Geracao 1:
[[11,10,13,4,9,12,8,3,6,2,1,7]*423,[9,6,2,10,12,1,7,11,13,4,3,8]*455,[9,6,2,10,12,1,7,11,13,4,3,8]*455,[1,6,8,12,10,9,7,3,11,4,13,2]*516,[1,6,8,12,10,9,7,3,11,4,13,2]*516]
Geracao 2:
[[11,10,13,4,9,12,8,3,6,2,1,7]*423,[9,6,2,10,12,1,7,11,13,4,3,8]*455,[9,6,2,10,12,1,7,11,13,4,3,8]*455,[9,6,2,10,12,1,7,11,13,4,3,8]*455,[1,6,8,12,10,9,7,3,11,4,13,2]*516]
Geracao 3:
[[11,10,13,4,9,12,8,3,6,2,1,7]*423,[9,6,2,10,12,1,7,11,13,4,3,8]*455,[9,6,2,10,12,1,7,11,13,4,3,8]*455,[11,10,13,7,9,12,8,3,6,2,1,4]*465,[1,6,8,12,10,9,7,3,11,4,13,2]*516]
Geracao 4:
[[11,10,13,4,9,12,8,3,6,2,1,7]*423,[9,6,2,10,12,1,7,11,13,4,3,8]*455,[9,6,2,10,12,1,7,11,13,4,3,8]*455,[11,10,13,7,9,12,8,3,6,2,1,4]*465,[1,6,8,12,10,9,7,3,11,4,13,2]*516]
Geracao 5:
[[11,10,13,4,9,12,8,3,6,2,1,7]*423,[9,6,2,10,12,1,7,11,13,4,3,8]*455,[9,6,2,10,12,1,7,11,13,4,3,8]*455,[11,10,13,7,9,12,8,3,6,2,1,4]*465,[1,6,8,12,10,9,7,3,11,4,13,2]*516]
Geracao 6:
[[11,10,13,4,9,12,8,3,6,2,1,7]*423,[9,6,2,10,12,1,7,11,13,4,3,8]*455,[9,6,2,10,12,1,7,11,13,4,3,8]*455,[11,10,13,7,9,12,8,3,6,2,1,4]*465,[1,6,8,12,10,9,7,3,11,4,13,2]*516]
Geracao 7:
[[11,10,13,4,9,12,8,3,6,2,1,7]*423,[11,10,13,7,9,12,8,3,6,2,1,4]*465,[11,10,13,7,9,12,8,3,6,2,1,4]*465,[1,6,8,12,10,9,7,3,11,4,13,2]*516,[1,6,8,12,10,9,7,3,11,4,13,2]*516]
Geracao 8:
[[11,10,13,4,9,12,8,3,6,2,1,7]*423,[11,10,13,7,9,12,8,3,6,2,1,4]*465,[11,10,13,7,9,12,8,3,6,2,1,4]*465,[1,6,8,12,10,9,7,3,11,4,13,2]*516,[3,6,8,12,10,9,7,1,11,4,13,2]*532]

Individuo: 1
Camiao: eTruck01
Trajeto: [5,11,10,13,4,9,5]
Camiao: eTruck02
Trajeto: [5,12,8,3,6,2,5]
Camiao: eTruck03
Trajeto: [5,1,7,5]

Individuo: 2
Camiao: eTruck01
Trajeto: [5,11,10,13,7,9,5]
Camiao: eTruck02
Trajeto: [5,12,8,3,6,2,5]
Camiao: eTruck03
Trajeto: [5,1,4,5]

Individuo: 3
Camiao: eTruck01
Trajeto: [5,11,10,13,7,9,5]
Camiao: eTruck02
Trajeto: [5,12,8,3,6,2,5]
Camiao: eTruck03
Trajeto: [5,1,4,5]

Individuo: 4
Camiao: eTruck01
Trajeto: [5,1,6,8,12,10,5]
Camiao: eTruck02
Trajeto: [5,9,7,3,11,4,5]
Camiao: eTruck03
Trajeto: [5,13,2,5]

Individuo: 5
Camiao: eTruck01
Trajeto: [5,3,6,8,12,10,5]
Camiao: eTruck02
Trajeto: [5,9,7,1,11,4,5]
Camiao: eTruck03
Trajeto: [5,13,2,5]

true .
```

5) Análise de eficácia

Nº de Armazéns de Entrega	Melhor Solução	Melhor Indivíduo	Valor médio população final	Valor médio população final base
5	401	[[4,3,6,1,2]*401	537.4	571
6	428	[7,1,6,2,3,4]*444	492	685.4
7	340	[8,3,4,2,6,1,7]*340	373.8	836
8	417	,[8,3,4,2,6,1,7,9]*417	401.3	902
9	418	[8,3,4,2,10,6,1,7,9]*418	413.3	1085

Como observamos os tempos de valor médio população final são melhores relativamente aos do algoritmo dado.

Estudo do estado da arte da visão por computador para lidar com Pesagem de Veículos sem Contacto

Introdução Teórica

Antes de começarmos a tratar de informação mais especifica sobre a Pesagem de Veículos sem Contacto através de visão por computador temos de perceber um pouco do que se trata a visão por computador.

A visão de computador é a capacidade de um sistema computacional de processar, interpretar e compreender imagens e vídeos, assim como um ser humano. Ela é uma área da computação que se concentra em criar algoritmos e tecnologias capazes de extrair informação útil de imagens e vídeos.

A visão de computador é uma área de pesquisa ativa e em constante desenvolvimento, e tem sido aplicada em uma ampla variedade de campos, incluindo robótica, reconhecimento de padrões, medicina, vigilância e análise de tráfego.

Alguns dos desafios enfrentados pela visão de computador incluem a variação de iluminação, o ruído presente nas imagens, o movimento e o ângulo da câmera. Algoritmos de visão de computador precisam ser capazes de lidar com essas variações para fornecer resultados precisos e confiáveis.

A visão de computador tem sido responsável por algumas das mais notáveis conquistas da tecnologia recente, incluindo o reconhecimento de voz, a identificação de rostos e o reconhecimento de placas de trânsito. É uma área em constante crescimento e desenvolvimento, e tem o potencial de transformar a forma como interagimos com o mundo ao nosso redor.

Pesagem de Veículos sem Contacto

A visão de computador tem ajudado na segurança rodoviária em diversos aspetos um deles é na Pesagem de Veículos sem Contacto que quando aplicada ao nosso problema é bastante útil, pois os motoristas que transportam elevadas cargas nos seus camiões carregam uma grande responsabilidade na estrada, caso ocorra algum acidente rodoviário com este, pode causar elevados danos, tanto à mercadoria que transporta como a possíveis estragos ao ambiente que o rodeia, que pode incluir vidas humanas. Estas técnicas são importantes para preservar as infraestruturas como pontes, estradas, pavimentos, pois muitas delas a partir de um certo peso começam a ser danificadas. Por fim é útil para estimar o peso do camião para fins de cálculo de peso máximo que este pode transportar. [4]

A sobrecarga de veículos é um fenómeno muito comum. A sobrecarga dos veículos não só causa danos severos nas infraestruturas rodoviárias e encurtam a sua vida útil, mas também podem levar a acidentes de viação. Por conseguinte, a identificação da carga do veículo é de grande importância para a gestão de sobrecarregar os veículos e pode também fornecer informações valiosas para a conceção e manutenção de infraestruturas de transporte. [3]

Os métodos de pesagem de veículos normalmente utilizados, incluindo a pesagem estática, pesagem em movimento de pavimento (PWIM), e pesagem em movimento de ponte (BWIM) estão em contacto tecnologias com fraquezas tais como instalação incómoda, má durabilidade, curta vida útil, e elevado custo de manutenção. É desejável ter uma identificação

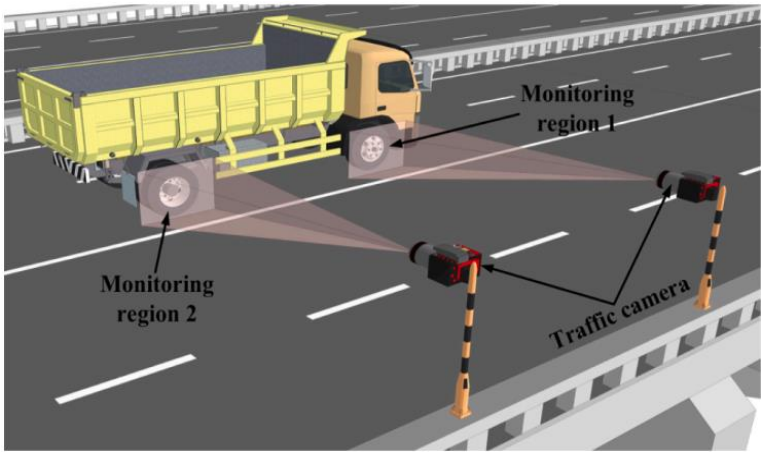

Figura 1 – Captura de imagens a partir de sensores

do peso do veículo sem contacto método sem qualquer sensor ou balança na/sobre a estrada e ponte. Existe um método que não implica contacto com o camião, este é contacto baseado no modelo de contacto pneu-estrada e visão de computador, baseia-se no contacto *Hertz* e a relação entre a força vertical do pneu e a deformação do pneu é obtida. [3] [2]

Após isto, as técnicas de visão por computador, tais como a segmentação da imagem e o reconhecimento de caracteres, são adotadas para a identificação da deformação do pneu e da pressão de inflação que são combinadas com o modelo teórico para determinar a força vertical do pneu e depois o peso do veículo. [1]

O cálculo do peso de um camião usando a Visão por Computador, pode ser dividida em 3 grandes áreas: [5]

- Aquisição de imagem do camião;
- Parâmetros da Roda (Deflação vertical, pressão e zona de contacto);
- Cálculo efetivo do peso.

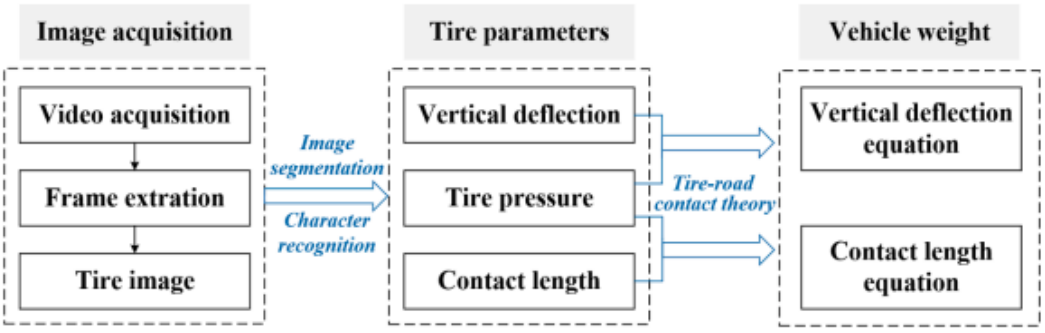

Figura 2 – Etapas para o cálculo do peso de um veículo

Como funciona a Pesagem de Veículos sem Contacto com uso de Inteligência Artificial?

Apesar da complexidade de um veículo, estimar o peso de um veículo requer fundamentalmente apenas de dois parâmetros do veículo, área de contacto pneu-estrada e pressão de contacto. A área de contacto de um pneu do veículo, também referido como a pegada do pneu, é a região do pneu em contacto com a estrada, o que é ilustrado na Fig. 3. A força dos pneus exercido na superfície da pista é igual ao contacto pneu-estra da pressão multiplicada pela área correspondente de contacto do pneu. O peso do veículo é a soma de todas as forças dos pneus como: [1] [2]

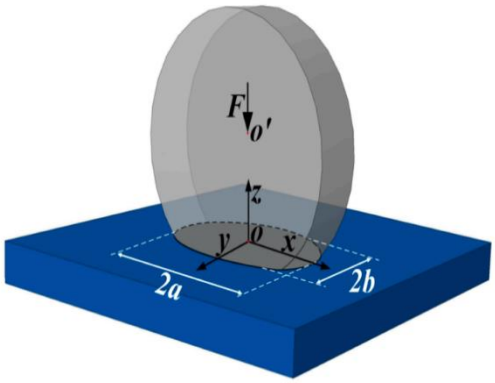

Figura 3 – Deflexão causada pelo peso do veículo

$$W = \sum_{i=1}^{N} P_i \cdot A_i$$

onde W é o peso do veículo, N é o número de pneus do veículo, P_i é a pressão de contacto pneu-estrada e A_i a área de contacto pneu-estrada do i-ésimo pneu.

O cálculo do contacto pneu-estrada é baseado na deflexão vertical do pneu δ (a diferença em diâmetro não deformado da borda do pneu e do deformado diâmetro no sentido vertical, devido ao peso do veículo) calculado da seguinte forma:

$$A = (\pi/2) \cdot C_3 \sqrt{(D_p\delta - \delta^2)(D_c\delta - \delta^2)}$$

Onde D_c é o diâmetro total do pneu. C3 e D_p são coeficientes constantes calculados empiricamente usando técnicas de cálculo fornecidos pelo fabricante do pneu, que os parâmetros técnicos fundamentais do pneu são ilustrados na Fig. 4. Nesta área de contacto pneu-estrada é estimada através da medição os dois parâmetros essenciais de deformação dos pneus utilizando o computador visão. As imagens dos sensores fornecem informações relevantes para o cálculo do peso como Modelo de pneu. (d) Marca do pneu. (e) Tamanho do pneu (uso do pneu, largura da secção, relação de aspeto, construção radial, e diâmetro da jante). [3]

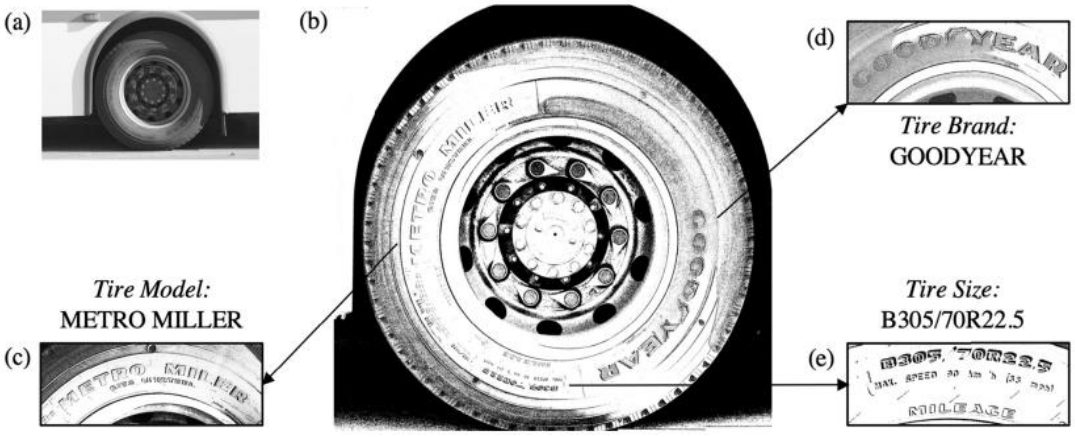

Figura 4 – Parâmetros da Roda de um veículo

Conclusões

Concluímos, desta forma, que à medida que o número de entregas aumenta, o tempo também aumentará, notamos também que a melhor viagem se altera consoante o número de entregas a fazer pelos vários armazéns. Também foi possível destacar que o tempo de execução cresce exponencialmente com o número de entregas. Por fim, como já abordado acima, os dois melhores indivíduos são sempre escolhidos para passar para a geração seguinte e a seleção dos restantes é feita de forma aleatória.

Referências

[1] M. Q. Feng, R. Y. Leung, and C. M. Eckersley, "Non-Contact vehicle Weigh-in-Motion using computer vision," *Measurement*, vol. 153, p. 107415, Mar. 2020, doi: 10.1016/J.MEASUREMENT.2019.107415.

[2] M. Q. Feng and R. Y. Leung, "Application of computer vision for estimation of moving vehicle weight," *IEEE Sens. J.*, vol. 21, no. 10, pp. 11588–11597, May 2021, doi: 10.1109/JSEN.2020.3038186.

[3] X. Kong, J. Zhang, T. Wang, L. Deng, and C. S. Cai, "Non-contact vehicle weighing method based on tire-road contact model and computer vision techniques," *Mech. Syst. Signal Process.*, vol. 174, p. 109093, Jul. 2022, doi: 10.1016/J.YMSSP.2022.109093.

[4] S. K. Leming and H. L. Stalford, "Bridge Weigh-in-Motion System Development Using Superposition of Dynamic Truck/Static Bridge Interaction," *Proc. Am. Control Conf.*, vol. 1, pp. 815–820, 2003, doi: 10.1109/ACC.2003.1239122.

[5] S. K. Upadhyaya, D. Wulfsohn, Relationship between tire deflection characteristics and soil-tire contact area, American Society of Agricultural Engineers (Microfiche collection) (USA), no. 80–1005. St. Joseph, MI, 1988.

[6] S. K. Clark. Mechanics of Pneumatic Tires, vol. 1. University of Michigan, Washington, DC 20590, 1981.