

Relatório do Sprint B de Algoritmia Avançada

**Turma: 3DD Grupo 20**

**Membros do grupo:**

1181255 Nuno Teixeira

1181498 Rafael Barbarroxa

1181500 Vítor Crista

1181529 Diogo Sousa

**Docente/Orientador:**

António Silva (ASS)

**Unidade Curricular:**

Algoritmia Avançada



Índice:

[1. Introdução 3](#_Toc58187781)

[2. Representação do conhecimento do domínio 3](#_Toc58187782)

[3. Estudo da viabilidade e complexidade dos geradores de todas as soluções (com e sem findall) – minimização do nº de mudanças de linha 4](#_Toc58187783)

[4. Adaptação do gerador de todas as soluções (sem findall) à minimização do horário de chegada ao destino pretendido 6](#_Toc58187784)

[5. Adaptação do A\* para minimização do horário de chegada ao destino pretendido 7](#_Toc58187785)

[6. Adaptação do Best First com heurística de minimização da distância ao destino e considerando os horários sobre a solução encontrada 8](#_Toc58187786)

[7. Estudo de viabilidade e complexidade e da melhor solução obtida considerando as implementações (Gerador de todas as soluções, A\* e Best First) – minimização do tempo de chegada ao destino pretendido 9](#_Toc58187787)

[8. Conclusão 11](#_Toc58187788)

[Referências 12](#_Toc58187789)

# Introdução

Durante a realização do sprint foram construídos e analisados vários algoritmos de pesquisa, tendo sido retiradas conclusões sobre a sua complexidade e viabilidade comparativamente uns com os outros.

Após uma análise sucinta dos elementos do domínio e da sua representação como factos da base de conhecimento, foi feita uma apreciação detalhada dos métodos geradores de soluções com e sem findall que nos foram fornecidos, tendo sido averiguado que o sem findall realiza o mesmo processo de uma maneira mais eficiente e com uso de menos memória.

Seguidamente, foi desenvolvido um algoritmo com base no gerador de todas as soluções sem findall para ser adaptado aos horários de autocarros. Isto foi feito no intuito de após a finalização dos algoritmos de pesquisa A\* e Best First, também adaptados a horários, comprovar através de uma outra análise de complexidade que ambos são mais eficientes que o gerador de todas as soluções. Comparativamente um com o outro o Best First é mais rápido, mas pode não gerar uma solução correta devido à heurística utilizada.

# Representação do conhecimento do domínio

O domínio do projeto de ALGAV não corresponde na totalidade ao domínio de LAPR5, sendo apenas representado por 3 factos:

* **Nó**: ponto da rede de transportes.

Representados pelo facto no/6, correspondente a nó (nome da paragem, abreviatura da paragem, flag ponto de rendição, flag estação de recolha, longitude, latitude).

Durante a realização do projeto a navegação entre nós era feita através da abreviatura da paragem, sendo essa utilizada para a representação de nós em outros factos. As flags de ponto de rendição e estação de recolha são essenciais, porque são estas que permitem a criação de ligações.

* **Linha**: Corresponde a um conjunto de percursos que são trajetos coincidentes com a sequências de nós. Contrariamente aos nós, as linhas não são equivalentes nos domínios de LAPR5 e ALGAV, consistindo numa junção de linhas e percursos.

Assim sendo, linha/5 tem como parâmetros linha (nome da linha, número do percurso, lista das abreviaturas das paragens, tempo minutos, distância metros).

* **Horário**: definição horária de um percurso, tendo a viagem como equivalência a LAPR5.

Em relação ao facto **horario/2**, este é constituído por horário (número do percurso, lista de horários), em que a lista de horários corresponde à lista das abreviaturas das paragens da linha cujo número do percurso é igual. Isto permite saber a que horas um autocarro chega a um determinado nó.

Para além dos factos que têm relação com o domínio de LAPR5, no âmbito de resolver os algoritmos propostos foi criado um facto **liga/3**, constituído por liga (abreviatura da paragem inicial, abreviatura da paragem final, número do percurso), de modo a criar todas as ligações possíveis entre pontos de rendição/estações de recolha.

# Estudo da viabilidade e complexidade dos geradores de todas as soluções (com e sem findall) – minimização do nº de mudanças de linha

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nº pontos de rendição e estações de recolha | Caminho Testado (Noi -> Nof) | Nº soluções | Tempo de resposta da solução com findall (s) | Tempo de resposta da solução sem findall (s) |
| 6 | ESTPA -> ESTLO | 2 | 0 | 0 |
| 12 | ESTPA -> ESTLO | 30 | 0.002 | 0.001 |
| 13 | ESTPA -> ESTLO | 198 | 0.023 | 0.016 |
| 14 | ESTPA -> ESTLO | 879 | 0.105 | 0.093 |
| 15 | ESTPA -> ESTLO | 1175 | 0.161 | 0.136 |
| 16 | ESTPA -> ESTLO | 3308 | 0.368 | 0.339 |
| 17 | ESTPA -> ESTLO | 13745 | 1.886 | 1.783 |
| 18 (adição de 1 nó e 2 linhas) | ESTPA -> ESTLO | 77460 | 14.472 | 13.763 |
| 19(adição de 1 nó e 2 linhas) | ESTPA -> ESTLO | 87284 | 16.638 | 15.638 |
| 20(adição de 1 nó e 1 linha) | ESTPA -> ESTLO | 235797 | 34.605 | 29.026 |
| 20(adição de 1 linha) | ESTPA -> ESTLO | 544179 | 98.316 | 75.839 |
| 20(adição de 1 linha) | ESTPA -> ESTLO | - | ERROR: Stack limit (1.0Gb) exceeded | 98.7501 |
| 21(adição de 1 nó e 1 linha) | ESTPA -> ESTLO | - | ERROR: Stack limit (1.0Gb) exceeded | 299.212 |
| 21(adição de 1 linha) | ESTPA -> ESTLO | - | ERROR: Stack limit (1.0Gb) exceeded | 356.439 |

Figura 1-Tabela da viabilidade e complexidade dos geradores de todas as soluções

Figura 2- Gráfico da viabilidade e complexidade dos geradores de soluções

No sentido de testar a viabilidade e complexidade dos geradores de soluções, foi-se aumentado gradualmente o número de pontos de rendição e estações de recolha. Como já era de esperar, com o aumento dos mesmos registou-se um acréscimo no número de soluções, bem como, o tempo de resposta das soluções com e sem findall.

Apesar disso, é de assinalar que o aumento não é constante para todos os nós, isto é, dependendo das linhas, a mudança de um determinado nó para um ponto de rendição pode ter mais ou menos impacto na diversidade de soluções. Assim sendo, podemos afirmar que apesar de o número de nós ser um fator no acréscimo das soluções, as linhas têm mais impacto, já que, a adição de linhas com alta complexidade (que passam por nós que estão presentes em várias linhas) estão associadas ao maior crescimento do número de soluções e tempo de resposta.

Num determinado ponto, o incremento do número de soluções gerou um problema no algoritmo que utilizava o findall, já que, o mesmo gera todas as soluções e guarda-as numa lista para depois averiguar qual é a menor. No caso de se tratar de dimensões reduzidas não é problemático, mas quando o número de soluções ultrapassa um determinado número (Stack Limit, neste caso 1Gb) ocorre um erro.

# Adaptação do gerador de todas as soluções (sem findall) à minimização do horário de chegada ao destino pretendido

Este algoritmo foi baseado no gerador de todas as soluções (sem findall) anteriormente analisado, por isso o algoritmo mantém a mesma forma de pesquisa, derivando apenas quando tem de ser feita a escolha entre diferentes caminhos. No caso anterior era feito através do tamanho do mesmo (length), enquanto que agora é através do tempo.

Assim sendo, foi desenvolvido um algoritmo que calcula o horário para um determinado caminho, recebendo para além do caminho, a hora de partida (valor introduzido pelo utilizador).

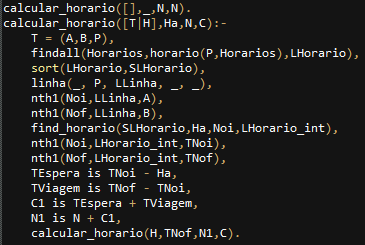


Figura 3- Algoritmo calcular\_horario

Isso é possível utilizando um findall seguido de um sort, de maneira a encontrar todos os horários para o caminho e, a partir daí, procurar o próximo horário numa dada paragem (recolha ou rendição) com horário superior ao instance de tempo.

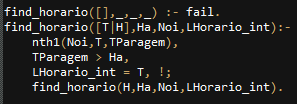


Figura 4- Algoritmo find\_horario

Desta forma é possível percorrer o caminho, incrementando o tempo gasto tanto em viagens como em esperas, de maneira a compará-lo com o anteriormente guardado, gerando assim a melhor solução.

# Adaptação do A\* para minimização do horário de chegada ao destino pretendido

Tal como o algoritmo anterior, o A\* tem como base o disponibilizado, onde se modificou a maneira como o mesmo escolhe o próximo nó no grafo. Neste caso, existiram dois valores que influenciaram a decisão sendo eles, o **horário**, que é calculado com o algoritmo anteriormente explicado; e uma **estimativa do tempo gasto no caminho** dividindo a distância entre nós, calculada através das coordenadas, e a velocidade máxima das linhas.

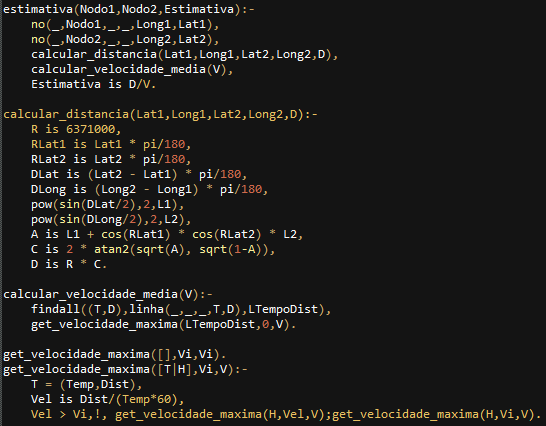


Figura 5- Algoritmo estimativa

A **distância** foi calculada através da fórmula de Haversine adaptada a SWI-PROLOG, enquanto que a **velocidade máxima das linhas** foi feita comparando o valor resultante da divisão entre a distância em metros e o tempo em minutos.

# Adaptação do Best First com heurística de minimização da distância ao destino e considerando os horários sobre a solução encontrada

Com base no algoritmo fornecido, a única mudança feita no Best First foi no cálculo da estimativa, que é usada para influenciar a escolha do nó seguinte. Como o objetivo é utilizar uma heurística de minimização da distância, foi usado o algoritmo **calcular\_distancia/5**, anteriormente desenvolvido para o A\*. Após gerado o caminho, é calculado o horário do mesmo através do algoritmo **calcular\_horario/4**.



Figura 6- Algoritmo estimativa com heurística de distância

Como estamos a lidar com uma heurística de minimização da distância, lidando apenas com horários numa visão final, quando o caminho é criado, não há maneira de garantir que o caminho desenvolvido é possível tendo em conta a hora de início, bem como os horários dos autocarros. Com isso em mente, uma simples alteração dentro do findall do Best First permite validar o caminho criado com os horários.

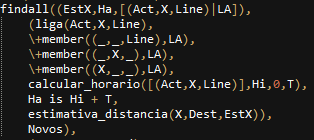


Figura 7- Findall com validação horária

# Estudo de viabilidade e complexidade e da melhor solução obtida considerando as implementações (Gerador de todas as soluções, A\* e Best First) – minimização do tempo de chegada ao destino pretendido

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nº pontos de rendição e estações de recolha | Caminho Testado (Noi -> Nof) | Nº soluções | Tempo de resposta do gerador de todas as soluções | Tempo de resposta do A\* |
| 6 | ESTPA -> ESTLO | 2 | 0.009 | 0 |
| 8(adição de 5 horários) | ESTPA -> ESTLO | 29 | 0.011 | 0.007 |
| 12 | ESTPA -> ESTLO | 46 | 0.039 | 0.011 |
| 14 | ESTPA -> ESTLO | 71 | 0.173 | 0.024 |
| 17 | ESTPA -> ESTLO | 91 | 1.015 | 0.047 |
| 17(adição de 1 horário) | ESTPA -> ESTLO | 135 | 1.433 | 0.078 |
| 17(adição de 4 horários) | ESTPA -> ESTLO | 135 | 1.527 | 0.142 |
| 19(adição de 2 linhas e 2 horários) | ESTPA -> ESTLO | 204 | 1.914 | 0.193 |
| 19(adição de 2 linhas e 2 horários) | ESTPA -> ESTLO | 204 | 12.716 | 0.211 |
| 19(adição de 2 horários) | ESTPA -> ESTLO | 325 | 19.273 | 0.485 |
| 19(adição de 3 horários) | ESTPA -> ESTLO | 691 | 23.542 | 1.781 |
| 20(adição de 3 linhas e 3 horários) | ESTPA -> ESTLO | 996 | 25.329 | 3.621 |

Figura 8- Tabela da viabilidade e complexidade do gerador de todas as soluções e A\*

Figura 9- Gráfico da viabilidade e complexidade do gerador de todas as soluções e A\*

Como era de esperar, tendo em consideração a análise feita anteriormente (ponto 3), o número de pontos de rendição, só por si, não aumenta o número de soluções. Este aumenta somente se os pontos de rendição estiverem associados a linhas e por sua vez a horários, de maneira a influenciar o cálculo do tempo.

A nível comparativo, podemos observar que o **algoritmo A\*** é muito mais eficiente do que o **gerador de soluções**, já que consegue obter o mesmo resultado, para o mesmo número de soluções, num tempo muito reduzido. Esta disparidade temporal deve-se à maneira como os algoritmos obtêm os resultados, já que, o A\* utiliza não só uma função que estima a distância à solução, como também utiliza o uso de custos acumulados conhecidos. Além disso, tem a possibilidade de comutar, de um ponto para outro na árvore de pesquisa, sem que o novo ponto seja um descendente do primeiro.

Relativamente ao **Best First** verificámos uma disparidade ainda maior, já que a heurística de distância torna o algoritmo muito mais eficiente que o A\*, no entanto os números limitados de testes não foram suficientes para o Best First demorar mais que 0s. Apesar disso, é necessário reforçar que o Best First apesar de mais eficiente, pode dar como resultado caminhos que não são possíveis para os horários conhecidos e para a hora inicial dada pelo utilizador, algo que não aconteceria no A\* e gerador de todas as soluções.

Além do mais, ainda que o nº de soluções só seja aplicável para o gerador de soluções, foi colocado como um indicador da complexidade das linhas e horários que influenciam o tempo de resposta do A\* e do Best First.

# Conclusão

Concluindo, após a realização de todos os algoritmos e da sua posterior análise incessante ao longo do sprint, é clara a distinção entre algoritmos que geram todas as soluções e algoritmos de pesquisa que encontram unicamente a melhor solução para o seu critério. Numa perspetiva de complexidade o gerador de todas as soluções não é ideal porque são criados caminhos que de imediato podem ser descartados, visto que já se conhecem melhores, é só uma perda de tempo.

Apesar disso, entre os geradores de todas as soluções, é de assinalar que o uso de findall de modo a gerar todos os caminhos, de forma que depois esses sejam comparados para se descobrir qual é o mais pequeno, é facilmente simplificado ao avaliar se o caminho é mais pequeno do que o atualmente mais pequeno. Isto é feito à medida que se vão gerando caminhos novos, como foi realizado, evitando-se assim iterar a lista de caminhos duas vezes.

No caso dos algoritmos de pesquisa, o Best First é muito mais eficiente que o A\*, graças ao uso da heurística de distância que visa simplificar o problema. No entanto, por causa do uso dessa heurística, o Best First pode não gerar a melhor solução, ou até, na pior das hipóteses, não gerar uma solução viável para os horários conhecidos. Por isso o A\* é a melhor opção no caso de ter de se garantir a melhor solução possível e o Best First no caso da dimensão do problema ser muito elevada, já que o A\* pode demorar muito tempo ou até não ser capaz de gerar uma solução.

**Distribuição de tarefas**:

Diogo Sousa: Adaptador do gerador de todas a soluções (sem findall)

Nuno Teixeira: Algoritmo A\* e análise de complexidade e viabilidade e da melhor solução obtida considerando as implementações (Gerador de todas as soluções, A\* e Best First).

Rafael Barbarroxa: Algoritmo Best First

Vítor Crista: Estudo da viabilidade e complexidade dos geradores de todas as soluções (com e sem findall) e representação do conhecimento do domínio.

# Referências

Curricular, U., & Avançada, A. (s.d.). *Apoio ao Sprint B do Trabalho Prático de ALGAV Planeamento da Mudança de Motoristas/Tripulações de Autocarros.*

(s.d.). *Heurísticas e Best First Sugestões para adaptação ao planeamento de mudanças de motoristas/tripulações.*

Melhor, P., & Ramos, C. (s.d.). *5-Métodos de Pesquisa.*

(s.d.). *O método de pesquisa A\* e sugestões para adaptação ao planeamento de mudanças de motoristas/tripulações.*