

Universidade do Minho  
Departamento de Informática

Green Distribution  
Inteligência Artificial  
Grupo 24

5 de janeiro, 2022



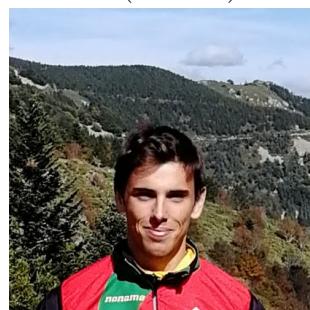
Beatriz Rodrigues  
(a93230)



Francisco Neves  
(a93202)



Henrique Costa  
(a93325)



José Pedro  
(a93163)

# Índice

<b>1</b>	<b>Introdução</b>	<b>3</b>
<b>2</b>	<b>Formulação do problema</b>	<b>4</b>
<b>3</b>	<b>Criação de circuitos</b>	<b>6</b>
3.1	Contexto . . . . .	6
3.2	Forma de criação dos circuitos . . . . .	7
3.3	Especificação das estratégias de procura . . . . .	8
3.3.1	Procura não informada . . . . .	8
3.3.2	Procura informada . . . . .	9
<b>4</b>	<b>Comparações</b>	<b>11</b>
4.1	Contexto . . . . .	11
4.2	Comparação por Distância . . . . .	11
4.3	Comparação por Tempo . . . . .	11
4.4	Minimização de circuitos por Distância . . . . .	12
4.5	Minimização de circuitos por Tempo . . . . .	12
<b>5</b>	<b>Recursos implementados</b>	<b>13</b>
<b>6</b>	<b>Resultados</b>	<b>15</b>
<b>7</b>	<b>Comentários finais e conclusão</b>	<b>16</b>

# 1. Introdução

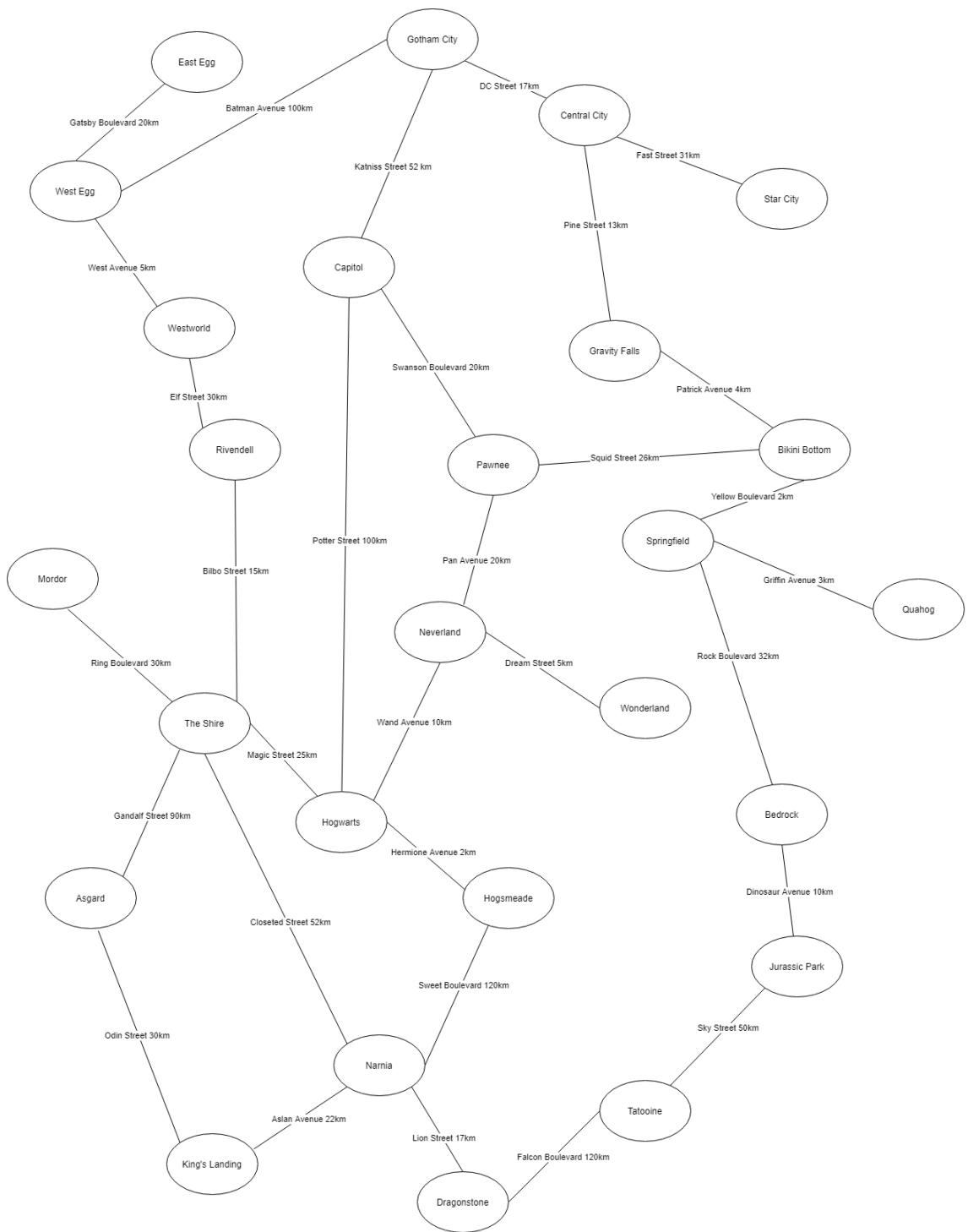
Este trabalho foi realizado no âmbito da unidade curricular de Inteligência Artificial, do terceiro ano, da Licenciatura em Engenharia Informática da Universidade do Minho, e teve como objectivo a implementação de métodos de resolução de problemas e de procura utilizando a base de conhecimento definida na fase anterior para a empresa fictícia *Green Distribution*. Neste trabalho foram implementadas estratégias para a resolução de problemas com o uso de algoritmos de procura, partindo da formulação do problema em questão e desenvolvendo os mecanismos de raciocínio considerados adequados para esta problemática.

Nesta fase foi dada continuidade tanto ao **menu** como à **interface gráfica**. No que respeita à implementação, foram expandidos novos conhecimentos no ficheiro *knowledgeBase.pl* e criadas consultas dos circuitos no ficheiro *graph\_queries*. Porém, para aproveitar todas as funcionalidades da aplicação, basta continuar a utilizar o ficheiro *queries.pl*, através do *menu* citado anteriormente.

## 2. Formulação do problema

Tendo em conta que o problema surge num ambiente determinístico e completamente observável, onde é possível o agente conhecer de forma exata o estado em que estará e as soluções pretendidas são sequências, pode-se afirmar que este é um problema de estado único. Além disso, a sua formulação pode ser efetuada da seguinte forma:

- **Representação do estado:** Grafo não orientado, em que cada nodo representa uma cidade e cada aresta representa o trajeto entre cidades.
- **Estado inicial:** Neverland, representa o local a partir de onde se vão iniciar todas os circuitos de entrega.
- **Estado/teste objetivo:** Neverland, após terem sido efetuadas as entregas desejadas para esse circuito.
- **Operadores:** Deslocação entre cidades.
- **Solução:** Um circuito (sequência de cidades até retornar à cidade inicial) que poderá ser o circuito mais rápido ou o circuito mais ecológico.
- **Custo solução:** Poderá ser de acordo com o tempo, distância ou trânsito.



Grafo representativo do mapa

# 3. Criação de circuitos

## 3.1 Contexto

A criação dos circuitos é realizada na fase de inicialização da aplicação. Optamos por gerar todos os circuitos, que cobrem todos os locais do mapa, de modo a poder catalogá-los individualmente com um identificador único - o **código do circuito** - e disponibilizá-los para a fase de associação de uma encomenda a um circuito. Isto porque, para as encomendas serem associadas a circuitos, estes precisavam já estarem pré-definidos (para poderem ser analisados) e possuirem uma forma de identificá-los (para poderem ser devidamente escolhidos). O fluxo desde o registo até atribuição de um circuito à uma encomenda será abordado mais detalhadamente no capítulo 6.

Tendo em conta a vertente que se decidiu seguir, teríamos então de garantir uma boa variedade de opções de circuitos. Assim, utilizamos **5 estratégias de procura** para os gerar:

- Profundidade
- Largura
- Busca Iterativa Limitada em Profundidade
- Gulosa
- A estrela

As estratégias **Gulosa** e **A estrela** foram implementadas com duas variações, de acordo com a heurística da **distância** e do **trânsito**. Optamos por estas heurísticas dado que a estimativa da *distância em linha reta ao destino* e a estimativa do *trânsito médio ao destino* são dois factores relevantes para uma empresa de distribuições de entregas. Com isso, o sistema criado totaliza **7 formas distintas de gerar circuitos**.

## 3.2 Forma de criação dos circuitos

A escolha tomada para a criação dos circuitos baseia-se em *criar uma grande variedade de opções de circuitos*, para depois ser possível ter uma gama de circuitos a serem comparados. Numa fase inicial, o mecanismo de criação de circuitos baseou-se na seguinte ideia: *gerar os caminhos, segundo uma estratégia, da origem para todos os nodos existentes*. Isto seria replicado para cada uma das 7 estratégias referidas, totalizando a criação de **7 x (número de destinos possíveis)** circuitos (na verdade este número é menor, dado alguns circuitos poderem ser iguais).

Porém, notou-se que podiam ser agrupados caminhos que eram sub-conjuntos de outros caminhos maiores. Isto é, se tivéssemos o caminho **Origem - A - B**, poderíamos descartar o caminho **Origem - A**. Com isto em mente, para cada estratégia utilizada, foi criado o seguinte algoritmo:

- **Passo 1:** gerar os caminhos, segundo uma estratégia, da origem para todos os nodos existentes.
- **Passo 2:** otimizar os caminhos, descartando aqueles que são sub-conjuntos de outros

Cada uma das estratégias de pesquisa implementam esta criação otimizada de circuitos, da origem para todos os destinos, através das regras intituladas como **gerar{NomeEstratégia}(Circuitos)**. Podendo assim, gerar os circuitos para todos os destinos individualmente para cada tipo de estratégia.

Este mecanismo de criação de circuitos, ainda numa fase prematura, implica que o caminho de volta seja o mesmo do caminho da ida, sendo a ideia de **circuito** traduzida por *alcançar o destino e voltar a origem pelo caminho inverso do utilizado*. Logo vimos que, para conseguir manter os aspetos característicos de cada estratégia na criação de circuitos e para conseguir traçar circuitos mais complexos e melhores, poderíamos evoluir o mecanismo. Assim, foi criada uma segunda forma de gerar circuitos, na qual, para além de ser utilizada a estratégia de pesquisa para alcançar todos os destinos possíveis, recorre-se à mesma estratégia para, a partir do destino, voltar a origem.

Outra diferença também seria a remoção do passo de otimização dos caminhos. A razão disto se encontra no contexto de nosso problema. Isto porque a atribuição de encomendas a circuitos pode variar se por exemplo, ser transportado mais de uma encomenda de uma vez. Assim, é útil para uma encomenda que tenha destino **A**, ter em conta tanto o caminho **Origem - A** como o caminho **Origem - A - B**. Remover a primeira opção só por ser sub-conjunto de outro caminho maior, seria errado. Isto porque pode existir um determinado dia que não tenha encomendas para se entregar ao destino **B**, e o estafeta que levar uma encomenda ao destino **A** não precisa ir até o ponto **B**. Em contrapartida, caso existir uma encomenda para o destino **B** no mesmo dia que uma encomenda esteja registada para o destino **A**, é útil também poder ter disponível um circuito que passe pelos 2 pontos de uma vez. Este segundo mecanismo de criação de caminhos é intitulado, para cada estratégia, como: **gerar{NomeEstratégia}2(Circuitos)**.

### 3.3 Especificação das estratégias de procura

#### 3.3.1 Procura não informada

##### Profundidade

Nessa estratégia de procura é expandido sempre um dos nodos mais profundos da árvore. Tal estratégia é boa de se aplicar quando há muitas soluções. O lado mau desta estratégia é a possibilidade de ficar presa em ramos errados para árvores demasiadas profundas. Porém pode-se dizer que em nosso contexto tal estratégia nunca encontra esta possibilidade dado a árvore ter profundidade finita e também ser tomado o cuidado de verificar se o próximo nodo a expandir já não faz parte dos nodos já percorridos (evitando assim ciclos).

Quanto à complexidade, esta estratégia apresenta as seguintes propriedades:

- **Temporal:**  $\mathcal{O}(b^m)$
- **Espacial:**  $\mathcal{O}(b * m)$

Onde  $b$  representa o fator máximo de ramificação da árvore de pesquisa e  $m$  a profundidade máxima do espaço de estados.

##### Largura

Difere da estratégia *primeiro em profundidade* na escolha do nodo a expandir: todos os nodos de menor profundidade são expandidos primeiro. É boa de se aplicar para problemas de pequena dimensão, justamente por demorar muito tempo e ocupar muito espaço. Como nosso mapa criado não possui muita complexidade, tal estratégia é exequível em nosso contexto. Em contrapartida não encontra a solução ótima (em nosso contexto) dado o custo de cada passo não ser igual.

Quanto à complexidade, esta estratégia apresenta as seguintes propriedades:

- **Temporal:**  $\mathcal{O}(b^d)$
- **Espacial:**  $\mathcal{O}(b^d)$

Onde  $b$  representa o fator máximo de ramificação da árvore de pesquisa e  $d$  a profundidade da melhor solução.

##### Busca Iterativa Limitada em Profundidade

Utilizada essencialmente quando a profundidade da solução não é conhecida. Assim, é utilizada a pesquisa *primeiro em profundidade* como sub-rotina das iterações feitas, mediante um nível máximo da profundidade para encontrar a solução em cada iteração. Em nosso contexto, tal estratégia apresentou resultados similares a pesquisa em *largura*.

Quanto à complexidade, esta estratégia apresenta as seguintes propriedades:

- **Temporal:**  $\mathcal{O}(b^d)$
- **Espacial:**  $\mathcal{O}(b * d)$

Onde  $b$  representa o fator máximo de ramificação da árvore de pesquisa e  $d$  a profundidade da melhor solução.

### 3.3.2 Procura informada

Esta secção apresenta as estratégias de procura que possuem *informações adicionais para serem consideradas para além do enunciado do problema*. Isto permite um critério de escolha especial para a expansão dos nodos.

#### Gulosa

A ordem de expansão dos nodos é escolhida segundo uma heurística. É escolhido o nodo que ”parece estar mais próximo da solução”. Em nosso contexto as heurísticas utilizadas estimam a ”distância em linha reta ao destino” e o ”percentual de trânsito” ao destino. Como já foi dito, achamos útil esses dados como heurística a ser utilizada num sistema em que a distância e trânsito são factores cruciais do desempenho. Como tal estratégia não têm em conta o custo do caminho já percorrido, não é encontrada a solução ótima.

É interessante mencionar que, por mais que criássemos essas *estimativas* para a heurística, teríamos de contornar o problema de não possuirmos todas as estimativas ( dado serem muitas as combinações possíveis a estimar de um nodo a qualquer outro segundo distância e trânsito). Assim, para a pesquisa Gulosa, foi implementado um mecanismo que, na ausência de informação da estimativa por parte de um dos nodos a se comparar, é escolhido aquele em que a estimativa é conhecida. Na ausência de ambos é escolhido o primeiro nodo da comparação, convergindo assim tal pesquisa para a estratégia em *Largura*.

Quanto à complexidade, esta estratégia apresenta as seguintes propriedades:

- **Temporal:**
  1. No melhor caso:  $\mathcal{O}(b * d)$
  2. No pior caso:  $\mathcal{O}(b^m)$
- **Espacial:**
  1. No melhor caso:  $\mathcal{O}(b * d)$
  2. No pior caso:  $\mathcal{O}(b^m)$

Onde  $b$  representa o fator máximo de ramificação da árvore de pesquisa,  $m$  a profundidade máxima da árvore e  $d$  a profundidade da solução.

## A\*

Difere da estratégia Gulosa no momento de avaliar o nodo a ser expandido. Desta vez é tido em conta também o custo do caminho já percorrido e por isso tal estratégia é completa. Utilizamos as heurísticas já mencionadas e, na ausência de informação das estimativas, considera-se o nodo a expandir aquele que possuir menor custo do caminho percorrido até ele, convergindo assim para a pesquisa do *custo uniforme*.

Quanto à complexidade, esta estratégia apresenta as seguintes propriedades:

- **Temporal:** Dependente do número de nodos com  $g(n) + h(n) \leq C^*$
- **Espacial:** Dependente do número de nodos com  $g(n) + h(n) \leq C^*$

Onde  $g(n)$  representa o custo do percurso até ao momento e  $h(n)$  o custo estimado para chegar ao objetivo.

## 4. Comparações

### 4.1 Contexto

Um aspeto importante do trabalho é a comparação de circuitos com base em elementos de produtividade, nomeadamente a distância e o tempo.

Para comparar dois circuitos com base na distância apenas é necessário a informação do grafo e porque ordem e quais os nodos os circuitos passam. No entanto, para comparar o tempo de dois circuitos é necessário ter também a informação das encomendas que se encontram associadas a esses circuitos. Isto porque para viabilizar uma entrega é necessário que a soma dos pesos das encomendas transportadas num circuito seja menor à permitida pelo transporte e que todas as encomendas cheguem a tempo. Desta forma o transporte escolhido para calcular a comparação fica condicionado a estes dois aspetos.

### 4.2 Comparação por Distância

Para a comparação em termos de distância obtém-se a distância de cada circuito e o resultado da comparação é a diferença de distâncias

### 4.3 Comparação por Tempo

Para a comparação em termos do tempo de entrega é necessário estabelecer um mínimo prazo máximo e obter a soma total dos pesos das encomendas a serem transportadas. Este mínimo prazo máximo é obtido percorrendo todas as encomendas associadas ao circuito e escolher a que tem uma data de entrega mais próxima de uma certa data de comparação. Se uma pessoa quisesse comparar dois circuitos no instante atual essa pessoa inseria a data do instante atual como argumento. A partir deste prazo de entrega calculado é possível saber qual a máxima duração que um circuito poderá ter.

Com uma máxima duração estabelecida e um total de carga é possível saber quais os transportes capazes de fazer o circuito. Este processo é feito em duas etapas. Numa primeira fase filtra-se os transportes que consigam suportar o peso das encomendas. Na segunda, faz-se outra filtragem por cima da anterior. A partir dos transportes mais ecológicos para os menos ecológicos é calculado o tempo que esse transporte demoraria para o circuito dado. O predicado usado para este cálculo é `tempo/4` que calcula o tempo de um percurso dado o circuito, peso e o transporte. Este predicado tem em atenção as restrições mencionadas no enunciado da 2<sup>a</sup> fase. Com o tempo calculado é necessário comparar se este tempo é menor que a

duração máxima. Se não for menor é calcula-se o tempo do próximo transporte mais ecológico, até se encontrar um tempo que cumpra com o requisito. Se não forem encontrados transportes que cumpram com este requisito é porque este circuito com estas encomendas não é execuível.

## 4.4 Minimização de circuitos por Distância

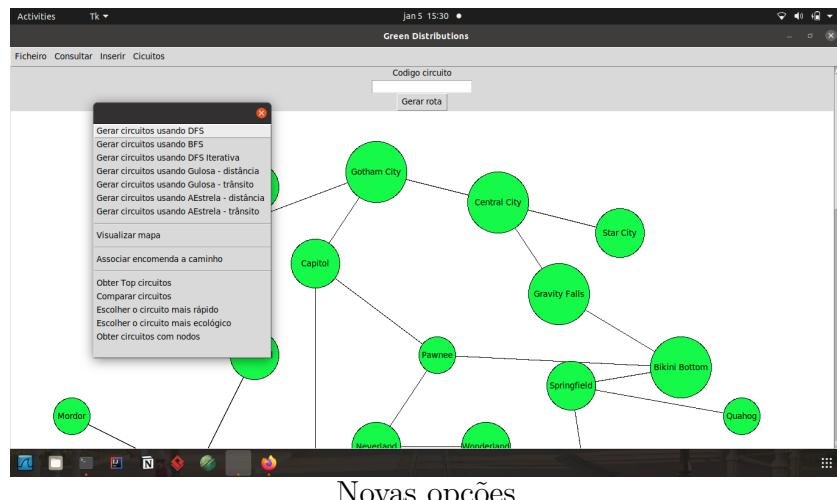
Calcula-se a distância para todos os circuitos dados por argumento e o que tiver a menor distância é retornado.

## 4.5 Minimização de circuitos por Tempo

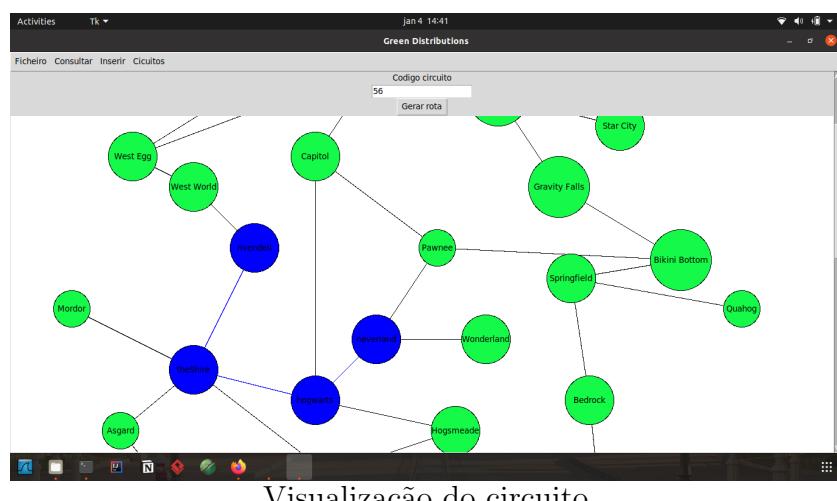
Tal como explicado na secção anterior, o tempo de um circuito é calculado com base no transporte mais ecológico que permita a execução do mesmo. Assim é calculado o tempo para cada circuito e é devolvido o que demora menos tempo.

## 5. Recursos implementados

Dando continuidade a interface gráfica mostrada na primeira fase do trabalho, foram extendidas suas funcionalidades. Com a criação de uma nova secção destinada aos **circuitos**, é possível então visualizar todos os circuitos gerados para cada tipo de estratégia de pesquisa. Para além disso é também possível visualizar graficamente o circuito, através de uma representação gráfica do mapa, realçando o caminho do circuito escolhido. É possível também comparar circuitos segundo critérios de distância e tempo, assim como obter os melhores circuitos segundo tais critérios. Um outro recurso adicional é a obtenção de circuitos que passam por determinados nodos escolhidos.



Novas opções



Visualização do circuito

The screenshot shows a desktop environment with several windows open. The main window is titled "Green Distributions" and contains four separate tables, each representing a step (Etapa) and location (Local) for a circuit. The tables show various locations such as neverland, hogwarts, theShire, rivendell, mordor, and pawnee. Below this main window, there are other smaller windows and icons visible in the background.

Etapa	Local
0	neverland
1	hogwarts
2	theShire
3	rivendell
4	westworld
5	rivendell
6	theShire
7	hogwarts
8	neverland

Etapa	Local
0	neverland
1	hogwarts
2	theShire
3	rivendell
4	theShire
5	hogwarts
6	neverland

Etapa	Local
0	neverland
1	hogwarts
2	theShire
3	rivendell
4	theShire
5	hogwarts
6	neverland

Etapa	Local
0	neverland
1	hogwarts
2	theShire
3	neverland

Etapa	Local
0	neverland
1	pawnee
2	bikiniBottom
3	gravityfalls
4	bikiniBottom
5	pawnee
6	neverland

Etapa	Local
0	neverland
1	pawnee
2	bikiniBottom
3	springfield
4	bikiniBottom
5	pawnee
6	neverland

Etapa	Local
0	neverland
1	hogwarts
2	neverland
3	hogwarts

Etapa	Local
0	neverland
1	hogwarts
2	neverland
3	hogsmade
4	hogwarts

Visualizações dos circuitos

Para além disto foram implementados dois novos **invariantes**. O primeiro deles garante que uma encomenda não estará associada a mais de um circuito (para o estafeta saber exatamente o circuito tendo em conta que este foi a melhor opção escolhida). O segundo invariante garante que não há mais de um circuito com o mesmo identificador (tornando assim o seu código único).

Com a expansão da interface com as novas funcionalidades da aplicação, podemos então conceber o ciclo de vida completo de utilização do programa, ou seja, o ciclo de vida de uma encomenda. Primeiramente é inserido um registo de uma nova encomenda, indicando todos os dados descritos na primeira fase do trabalho. Nessa altura, tal encomenda ainda não foi associada a nenhum circuito. Cabe ao utilizador da aplicação analisar a melhor opção de circuito para a encomenda e depois associá-la a tal circuito. A escolha do circuito pode ser auxiliada com os mecanismos de comparação sugeridos. Para além disto, caso seja notado que existem uma gama de encomendas no mesmo período de tempo, podem ser filtrados os circuitos que englobam os destinos pretendidos e depois compará-los. Feita escolha, resta associar a encomenda ao circuito e a partir dai os estafetas já possuem toda a informação necessária para realizar a entrega. Optamos por esse processo *manual* de análise de circuito para realçar todos os mecanismos implementados.

## 6. Resultados

Abaixo apresenta-se uma tabela que permite a realização de uma análise comparativa entre as diversas estratégias implementadas.

Estratégia	Tempo (seg)	Espaço (MB)	Indicador/Custo	Encontrou a melhor solução?
DFS	0.003	$\mathcal{O}(b * m)$	Distância	Não
BFS	0.001	$\mathcal{O}(b^d)$	Distância	Não
Iterativa em profundidade	0.003	$\mathcal{O}(b * d)$	Distância	Não
Gulosa	0.035	$\mathcal{O}(b^m)$	Trânsito	Não
Gulosa	0.037	$\mathcal{O}(b^m)$	Distância	Não
A*	0.050	$g(n) + h(n) \leq C^*$	Trânsito	Sim
A*	0.057	$g(n) + h(n) \leq C^*$	Distância	Sim

A máquina utilizada para a obtenção dos resultados possui as seguintes especificações:

```
OS: Manjaro Linux x86_64
Host: 81NC Lenovo IdeaPad S340-15API
Kernel: 5.13.19-2-MANJARO
Uptime: 6 hours, 28 mins
Packages: 1383 (pacman), 16 (snap)
Shell: bash 5.1.8
Resolution: 1920x1080
DE: Xfce 4.16
WM: Xfwm4
WM Theme: Matcha-sea
Theme: Matcha-dark-azul [GTK2/3]
Icons: Papirus-Maia [GTK2], Adwaita [GTK3]
Terminal: xfce4-terminal
Terminal Font: Monospace 12
CPU: AMD Ryzen 5 3500U with Radeon Vega Mobile G
GPU: AMD ATI 04:00.0 Picasso
Memory: 3088MiB / 9893MiB
```

Especificações da máquina utilizada

## 7. Comentários finais e conclusão

A partir do trabalho desenvolvido, foi possível diferenciar as estratégias relativamente ao seu desempenho e eficácia. Notou-se que as pesquisas informadas necessitam de um maior tempo de execução do que as pesquisas não informadas (como previsto, visto que o uso das heurísticas confere uma maior complexidade).

Ficou ainda evidente que se o objetivo pretendido fosse encontrar a melhor solução para o problema, a procura informada A estrela seria a única capaz de o fazer consistentemente. A pesquisa gulosa, apesar de também ser uma pesquisa informada, falha ao não ter em consideração o custo do caminho já percorrido. Por outro lado, a pesquisa em largura tem o problema de não ter em consideração que o custo de cada passo não é igual e a pesquisa em profundidade pode acabar a desperdiçar tempo a expandir ramos errados (algo que no contexto deste projeto não acontece, devido às dimensões do mapa desenvolvido). Por fim, a busca iterativa limitada em profundidade acabou por convergir para busca em largura devido, novamente, ao contexto em que foi desenvolvida.

Tendo tudo isto em conta, consideramos que podemos concluir que os desafios do projeto foram superados com êxito. Além destes, foram ainda implementadas diversas funcionalidades que, tal como referido anteriormente, foram consideradas pertinentes para a resolução do problema em causa.