U.PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Conceção e Análise de Algoritmos

Sistema de Evacuação

2MIEIC02 – Grupo 3 – Tema 3

(11 de abril de 2018)

Patrícia Janeiro **up201605946**@fe.up.pt

Luís Borges **up201605859**@fe.up.pt

Sandro Campos **up201605947**@fe.up.pt

Índice

1. Introdução

Na unidade curricular de Conceção e Análise de Algoritmos foi-nos proposta a implementação de um sistema de gestão de redireccionamento de veículos para a rede de autoestradas de Portugal, para casos de emergências que impossibilitem, temporariamente, o trânsito em algumas dessas vias.

Procedeu-se assim ao desenvolvimento de um sistema que determina os percursos alternativos que cada automóvel deve realizar caso exista um acidente grave, tendo esta solução em conta o cálculo do percurso mais rápido de um ponto inicial para um final, evitando os troços de autoestradas cortados.

2. Descrição do Problema

2.1 Input

Construção de um grafo G = (V, E), em que:

- Vértices (V) – representam os vários pontos num mapa de estradas.
- Arestas (E) – representam as ligações entre os vários pontos do mapa assim como a capacidade máxima e o tráfego atual da via.
- Nós de início e de destino.

2.2 Dados de entrada

Ficheiros com o tráfego a percorrer as estradas.

2.3 Output

O percurso que deve ser realizado de um ponto inicial para um ponto final, ou todos os percursos possíveis a partir de um ponto tendo em conta a existência de algum troço fechado, apresentado o percurso realizado no menor tempo possível.

2.4 Objetivo

Apresentar o percurso que deve ser realizado para evitar um troço fechado e chegar ao destino em segurança, no menor tempo possível, processando todo o tráfego. Neste caso, na inexistência de informação relativa à velocidade média de cada troço da autoestrada, assume-se que o caminho mais rápido a ser percorrido é aquele que corresponde ao caminho de menor distância.

2.5 Restrições

Evitar um troço fechado e impedir que os veículos não se dirijam para troços lotados das autoestradas, uma vez que cada um destes possui uma capacidade limitada, isto é, um número máximo de automóveis que suporta.

3. *Algoritmos e estruturas de dados*

A implementação do projeto para evacuação de veículos em caso de emergências tem como base a instanciação de uma classe genérica representativa de um **dígrafo**, isto é, um **grafo acíclico dirigido**. Este é, por isso, constituído por um conjunto de nós e arestas que os interligam representando uma rede viária.

Com o objetivo de encontrar o caminho mais rápido entre 2 pontos da rede viária tomaram-se em consideração, e implementaram-se, dois importantes algoritmos criados para este mesmo efeito: o algoritmo de **Dijkstra** e o **A***.

O Algoritmo de Dijkstra, pelo facto de encontrar o percurso mais curto num grafo dirigido ou não dirigido pesado em tempo **O((arestas+vértices]) * log (vértices))** e pelo facto de o **grafo** ser **esparso** (**|E| ~ |V|**), foi considerado como uma das possíveis soluções para este problema. É, no entanto, necessário ter em conta que este algoritmo só pode ser usado em grafos cujas arestas possuam pesos não negativos, caso que se verifica neste contexto.

Uma estrutura de dados necessária à sua implementação é a ***priority queue alterável***, ou semelhante, como um ***heap* binário** (*heap* de mínimos), que nos permita manter um conjunto de vértices ordenados por distância ao vértice origem, ao longo do cálculo do caminho mais curto. Assim, certifica-se que o vértice em processamento a cada iteração do algoritmo possui uma distância mínima em relação ao nó origem do trajeto. O algoritmo de Dijkstra procura, portanto, maximizar o ganho imediato a cada e diz-se que é por isso **ganancioso**.

Com o uso do heap binário conseguimos obter um desempenho geral bastante bom, da ordem **logarítmica**, uma vez que cada um dos vértices nele inseridos guarda a sua posição no vetor. Aquando da inserção ou da remoção de vértices do heap há, no entanto, necessidade de atualizar os seus índices.

No caso de grafos mais **densos**, em que o número de arestas por nó é elevado, o algoritmo **A*** apresenta uma eficiência melhorada dada a heurística que este incorpora, apesar de em geral não garantir a solução ótima. Este efetua, ao processar cada nó, uma estimativa da distância mínima ao vértice de destino e tem em consideração o

somatório dessa distância com a distância à origem de forma a dar vantagem aos vértices mais próximos do vértice destino. Assim, ganha-se algum **speedup** no processo de encontro do caminho mais curto.

```
DIJKSTRA(G, s):  // G=(V,E), s ∈ V
1.    for each v ∈ V do
2.       dist(v) ← ∞
3.       path(v) ← nil
4.    dist(s) ← 0
5.    Q ← ∅   // min-priority queue
6.    INSERT(Q, (s, 0)) // inserts s with key 0
7.    while Q ≠ ∅ do
8.       v ← EXTRACT-MIN(Q) // greedy
9.       for each w ∈ Adj(v) do
10.         if dist(w) > dist(v) + weight(v,w) then
11.            dist(w) ← dist(v)+ weight(v,w)
12.            path(w) ← v
13.            if w ∉ Q then  // old dist(w) was ∞
14.               INSERT(Q, (w, dist(w)))
15.            else
16.               DECREASE-KEY(Q, (w, dist(w)))
```

Pseudocódigo do algoritmo de Dijkstra

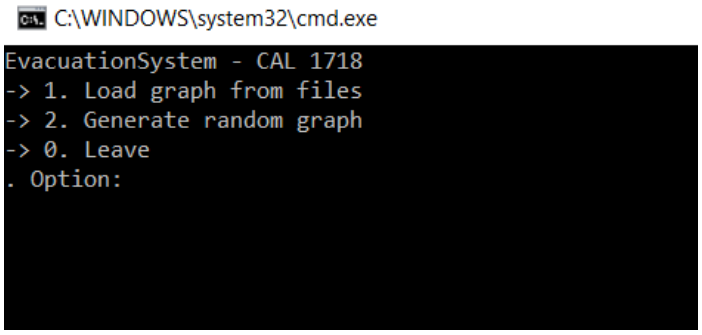
```
1  if goal(start) = true then return makePath(start)
2
3  open ← start
4  closed ← ∅
5  while open ≠ ∅ do
6      sort(open)
7      n ← open.pop()
8      kids ← expand(n)
9      forall the kid ∈ kids do
10         kid.f ← (n.g + 1) + h(kid)
11         if goal(kid) = true then return makePath(kid)
12         if kid ∩ closed = ∅ then open ← kid
13     closed ← n
14 return ∅
```

Pseudocódigo do algoritmo A*

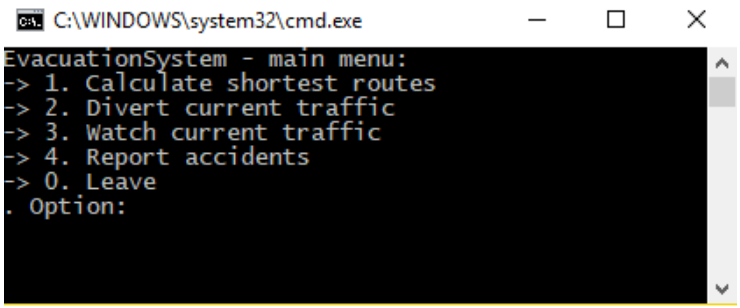
4. Casos de Utilização

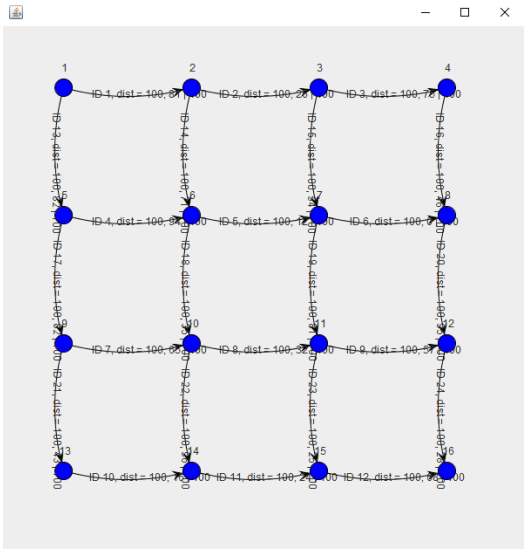
A interface do programa foi desenvolvida para a linha de comandos, sendo que todas as instruções são inseridas através desta. Todas os menus, incluindo inputs e outputs estão definidos na classe “Utils.cpp”.

Inicialmente opta-se por carregar um grafo já existente ou gerar um aleatoriamente. Este grafo, por mera simplificação, mas também para tornar o exemplo apresentado o máximo realista possível, possui no máximo 15 nós e cada nó pode possuir até um máximo de 2 arestas adjacentes, representando interseções e as autoestradas correspondentes, respetivamente.

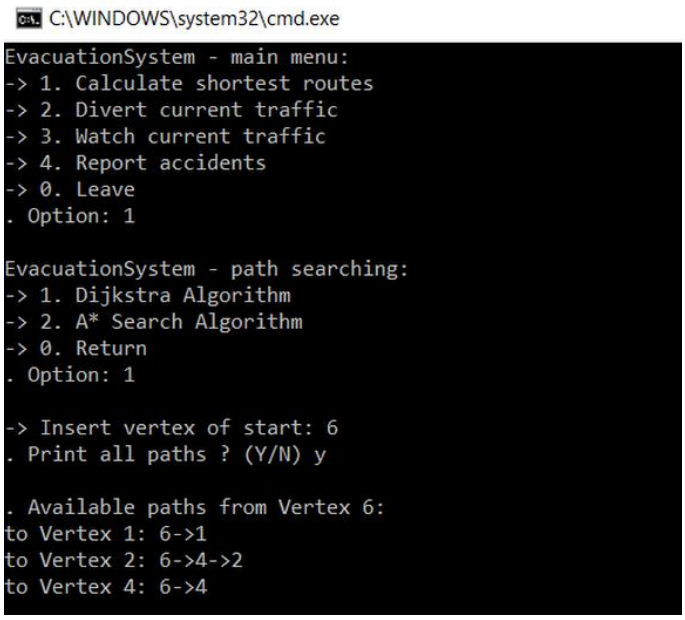


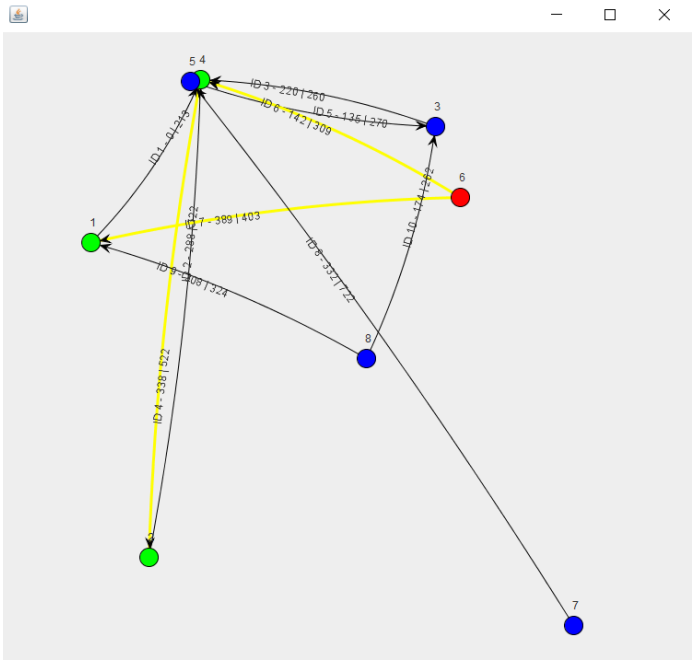
De seguida é possível selecionar uma das seguintes opções: calcular o caminho mais curto, divergir e observar o tráfego atual e reportar um novo acidente. É também apresentado o esquema do grafo, com os respetivos IDs de cada nó e aresta, assim como a distância entre nós, o número de veículos atuais num troço e a sua capacidade máxima.



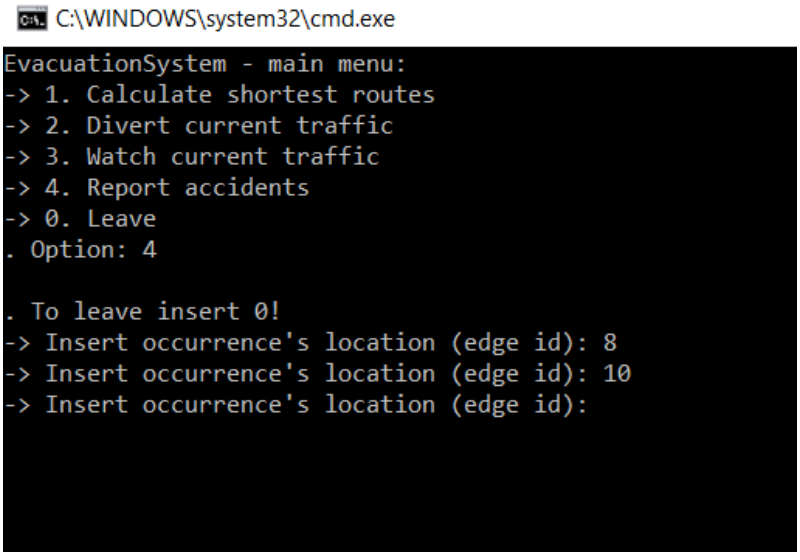


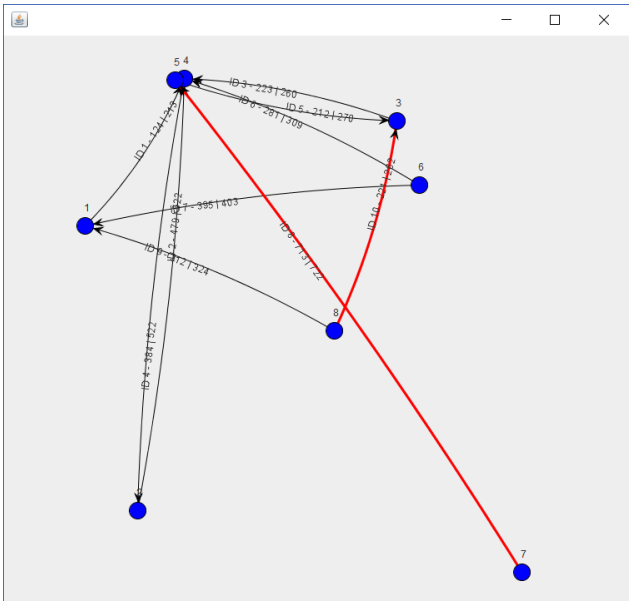
Na primeira opção é possível optar pelo algoritmo de Dijkstra e de A*, seleciona-se o ID do nó inicial e opta-se por exibir todo os percursos possíveis a partir desse nó ou a inserção de um nó de destino específico. Graficamente, os percursos ficam representados na cor amarela, o nó de partida a vermelho e o(s) nó(s) de chegada a verde.



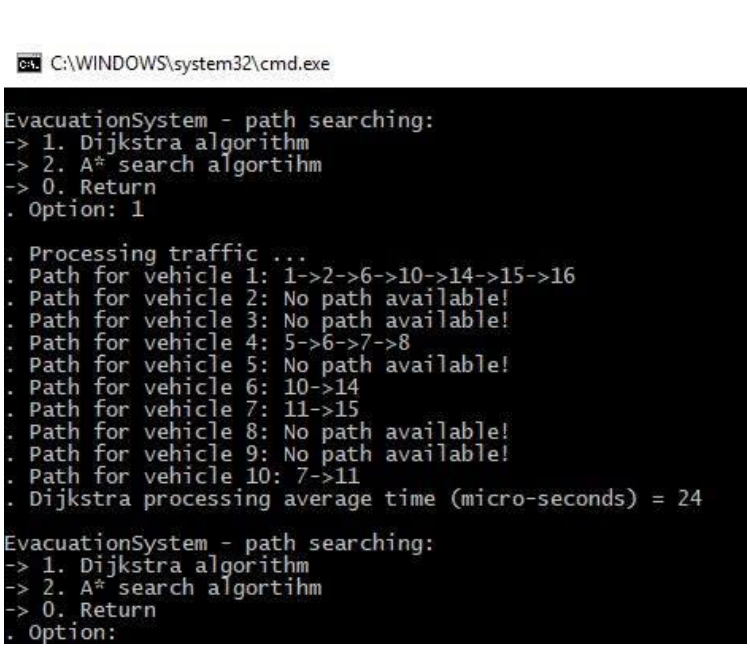


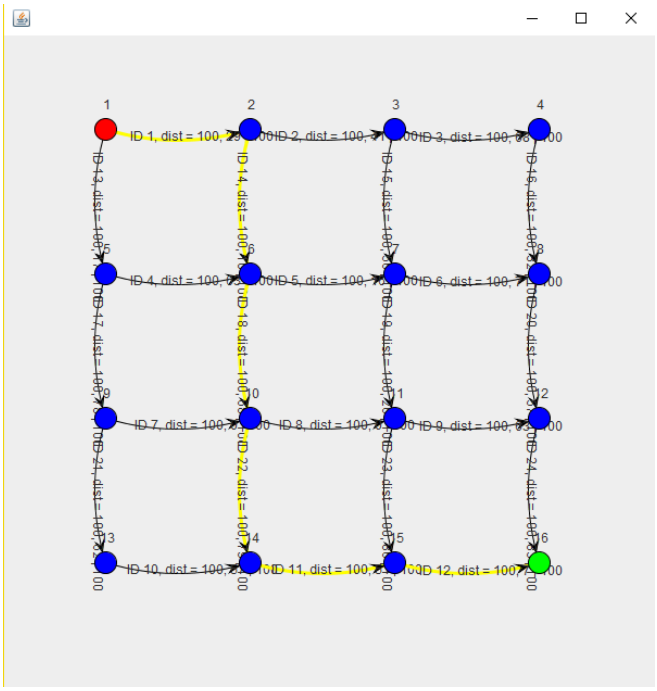
Na opção de reportar acidentes é possível marcar uma aresta, através do seu ID como inacessível, ficando esta marcada a vermelho.



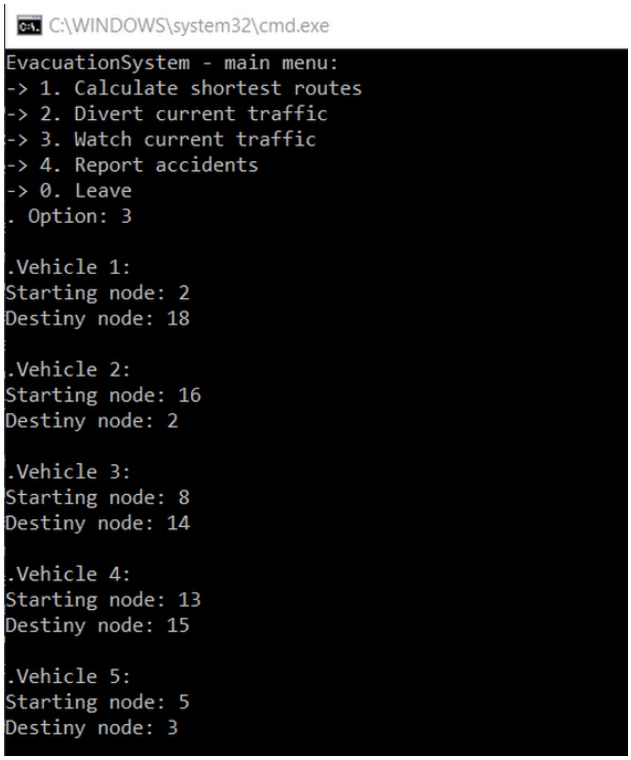


Na opção 2 é possível realizar o desvio do trânsito, tendo em conta a existência ou não de desvios. É atualizado também, ao mesmo tempo, o número de veículos existentes em cada estrada.





Na opção 3 é possível ver todo o tráfego não processado, sendo que este pode ser gerado aleatoriamente ou lido de um ficheiro “traffic.txt”.



5. *Diagrama de Classes*

6. *Principais Dificuldades*

Na realização deste projeto consideramos que a maior dificuldade encontrada foi na divisão de tarefas. Além disso consideramos também que o enunciado fornecido era pouco específico em relação às especificações do trabalho a realizar.

7. *Contribuição no Projeto*

Consideramos que a divisão de tarefas não foi a mais justa, como se apresenta a seguir: Patrícia Janeiro - 10%, Luís Borges - 20% e Sandro Campos - 70%.

8. Conclusão

Com a realização deste projeto foi possível um maior contacto com algoritmos de pesquisa do caminho mais curto em grafos, conceitos indispensáveis no âmbito do curso de Engenharia Informática. Tentámos usar mapas reais na realização do trabalho, tal não foi possível, mas esperamos incorporar o já elaborado para esse efeito na próxima entrega do trabalho.