Análise e Síntese de Algoritmos

Projeto 2

Grupo 21

João Martinho, 86454

Miguel Valério, 86483

Introdução:

O presente relatório procura apresentar uma solução ao segundo projeto proposto para a cadeira de Análise e Síntese de Algoritmos de 2º semestre do ano letivo de 2017/18.

O projeto baseia-se numa empresa de distribuição de mercadorias que tem investido na investigação de desenvolvimento de veículos sem motorista. Para tal é necessário desenvolver algoritmos eficientes para a segmentação das imagens captadas pela câmara desses veículos.

Cada imagem é dada por um retângulo de pixéis. O problema é então segmentar os pixéis, isto é, segmentá-los como sendo de 1º plano ou de cenário.

Descrição da solução:

Na resolução do problema, optámos pela linguagem de programação C.

A representação de uma imagem captada assenta na transformação do Input, constituído pelas dimensões da imagem (nº de pixéis por coluna e por linha), pelos valores de cada pixel pertencer ao 1º plano ou ao cenário e ainda pelos valores das relações de vizinhança entre pixéis (cada pixel é vizinho do pixel acima, do pixel abaixo, do pixel à esquerda e do pixel à direita), num grafo não dirigido (por cada aresta que ligue dois vértices existe uma que os liga no sentido oposto), onde os vértices são os diversos pixéis e as arestas representam as relações de vizinhança entre eles. Este grafo servirá como rede de fluxos para tal acrescentou-se ainda dois vértices: um a que chamamos P, que representa o 1º plano e que é a origem da rede, e outro que é fictício a que chamamos C, isto é, o vértice em si não existe, mas existem ligações para ele, este vértice representa o cenário. Todos os vértices ligam a P com o valor de pertencerem ao 1º plano e a C com o valor de pertencerem ao cenário.

O grafo foi representado sob a forma de uma matriz de adjacências. O conteúdo desta matriz é o peso das várias ligações entre vértices. Visto que cada pixel só é vizinho dos pixéis acima, abaixo, à esquerda e à direita, então cada linha da matriz só possui 6 elementos (1 para cada seu vizinho e ainda um para P e para C). No entanto a linha do vértice P tem um elemento para cada vértice existente na rede de fluxo, com exceção de si mesmo e do vértice C (visto que P nunca liga a nenhum destes dois).

A segmentação dos pixéis foi obtida através da aplicação de uma variante do algoritmo de *Edmonds-Karp*.

O nosso programa realiza os seguintes passos:

1. Leitura do *input* e construção do grafo;
2. Aplicação do algoritmo adaptado de *Edmonds-Karp*:
    a. Visto que, normalmente, todos os vértices ligam tanto à origem P como ao destino C, então percorre-se os V vértices e aplica-se o fluxo logo aos caminhos de aumento origem -> vértice -> destino, esgotando pelo menos uma das arestas destes caminhos. Esta variante é uma boa prática visto que sabemos *a priori* que a maior parte dos vértices possuem estas ligações. Estes caminhos de aumento seriam os V primeiros caminhos retornados pela BFS.
    b. Aplica-se o algoritmo *Edmonds-Karp* normal sobre a rede de fluxos resultante.
    c. O corte mínimo é obtido quando não existem mais caminhos de aumento. Através das estruturas auxiliares à BFS, é feita a divisão dos vértices: os vértices descobertos pertencem ao lado da origem e os restantes ao lado do destino.
3. Cálculo do peso total da segmentação: Uma vez que a origem é o vértice que representa o 1º plano e o destino é o vértice que representa o cenário, então os vértices que pertencem ao conjunto da origem não contribuem para as contas associadas ao 1º plano, o mesmo acontece para os restantes vértices, mas para as contas associadas ao cenário. Assim sendo, os vértices do conjunto da origem contribuem com o seu peso de cenário (representando os pixéis de cenário) enquanto os vértices do conjunto do destino contribuem com o seu peso de 1º plano (representando os pixéis de 1º plano). Para contribuição do corte (ligações entre os dois conjuntos) é feita a verificação de quais os vértices que ligam a vértices do conjunto oposto, considerando o peso dessas.
4. Impressão do *output*: apresenta-se peso total da segmentação seguido da representação do grafo em forma de matriz, onde cada vértice é representado por um “P” se for de 1º plano (se contribuir para as contas de 1º plano) ou com um “C” se for de cenário (se contribuir para as contas de cenário), apresentando assim a segmentação da imagem lida como *input*.

Análise Teórica

A complexidade temporal do programa desenvolvido, por se basear no algoritmo de *Edmonds-Karp*, estima-se que seja $O(V*E^2)$, onde V representa o número de vértices e E o número de arestas. No entanto, devido à representação feita e ao facto de cada vértice possuir 6 arestas (exceto a origem que possui cerca de V arestas), o número de arestas E é dado por 7*V, podendo estimar a complexidade temporal em $O(V^3)$.

Quanto à complexidade espacial estima-se que tenha um limite assintótico de O(V), visto que para a representação do grafo (imagem captada) é guardado uma matriz de dimensões V linhas por 6 colunas (com exceção da linha para o vértice P, que tem V colunas) e visto ainda que as outras estruturas auxiliares também são vetores de V elementos.

Análise Experimental

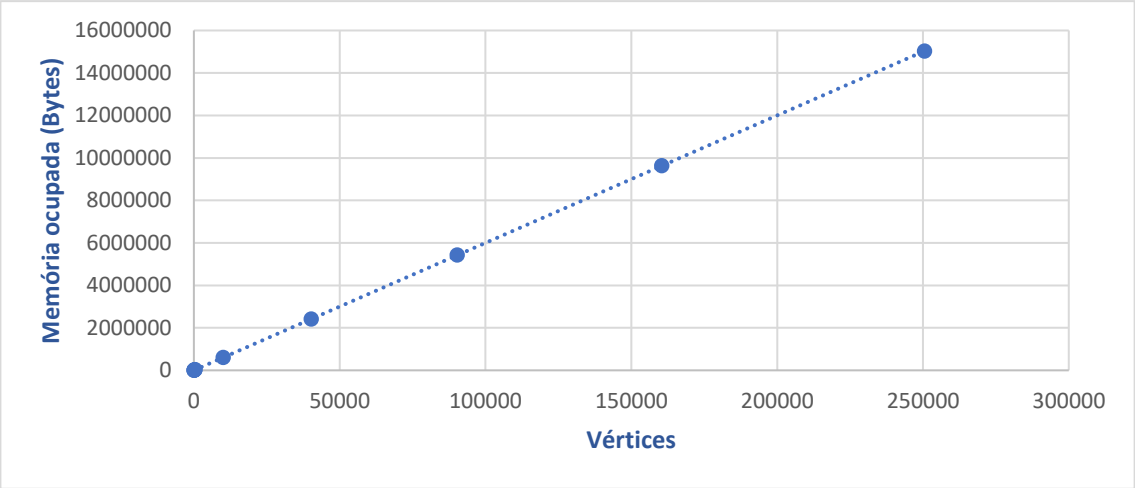
Para a análise experimental foram corridos 10 testes (10 grafos diferentes dados como input). Os testes foram corridos numa máquina com processador Intel® Core™ i7-7700HQ CPU @ 2.80GHz 2.81GHz e com 16GB de memória RAM, através de um subsistema (bash Ubuntu para Windows 10) com sistema operativo Ubuntu (distribuição Linux).

Os testes obtiveram os seguintes resultados experimentais:

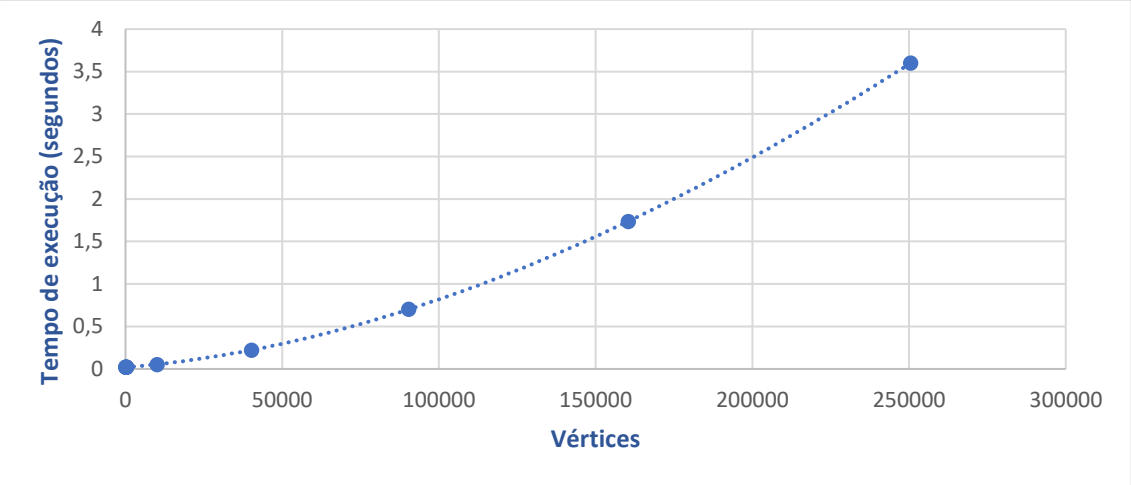
V	Tempo (s)	Espaço (B)
20	0.0203	2 248
72	0.0205	5 368
156	0.0202	10 408
272	0.0202	17 368
420	0.0207	26 248
10 100	0.0507	607 048
40 200	0.2211	2 413 048
90 300	0.7024	5 419 048
160 400	1.7353	9 625 048
250 500	3.5985	15 031 048

Nota: Os resultados obtidos são uma média de 10 experiências executadas para cada teste.

Com os resultados apresentados anteriormente foi possível desenhar os seguintes gráficos:



Tal como previsto na análise teórica, o espaço de memória reservado durante a execução dos testes cresce linearmente com o número de vértices dos grafos. Portanto, a complexidade espacial é O(V).



Tal como previsto na análise teórica, o tempo a execução dos testes cresce cubicamente com o número de vértices dos grafos. Portanto, a complexidade temporal é $O(V^3)$.

Referência:

A referência consultada para a realização deste projeto foi:

- **Introduction to Algorithms, Third Edition:** Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford SteinSeptember 2009 ISBN-10: 0-262-53305-7; ISBN-13: 978-0-262-53305-8.