**Análise e Síntese de Algoritmos**

2º Projeto - 3 de maio de 2018

*77906 António Sarmento 81947 Marta Simões*

**Introdução**

O exemplo do Sr. João Caracol com a sua empresa de distribuição de mercadorias e o seu desejo de reduzir os custos de operação da sua empresa serve para evidenciar um problema relacionado com fluxos. O algoritmo usado foi o Edmons-Karp .O objetivo é segmentar os píxeis, ou seja classificá-los como sendo de primeiro plano ou cenário.

**Análise Teórica**

Para representar uma grelha de Píxeis , temos os seguintes dados:

* Nº Vértices — V
* Nº Arcos — E

Também temos um grafo representado por um vetor de arcos que recorre a três vetores:

1. Tamanho: V; Índice: Número do Vértice; Conteúdo: Índice do Arco;
2. Tamanho: E; Índice: Número do Arco; Conteúdo: Vértice Final;
3. Tamanho: E; Índice: Número do Arco; Conteúdo: Arco Adjacente;

Dado uma grelha de Píxeis com dois pesos, mais um entre pares de Píxeis, um grafo é criado a partir da ligação de mais dois vértices entre todos os outros, um fonte *s* e um destino *t*.

Exemplo da representação de uma grelha de Píxeis 2 x 2 (esquerda), junto à sua transformação numa rede fluxos, representada como um Grafo dirigido (direita):

1

9

7

8

8

8

1

2

2

1

6

2

8

8

1

2

A nossa estrutura de dados tem a seguinte eficiência para cada operação:

* Espaço: O(V+E)
* Inicialização: O(V+E)
* Inserir Arco: O(1). Foram usados truques para obter esta complexidade.
* Encontrar Arco: Depende do par de vértices:
  + O(E) para par (s, u) no pior caso
  + O(1) para qualquer outro par

Para o cumprimento do nosso objetivo, poderíamos, em teoria, usar três algoritmos: **Ford-Fulkerson**, **Edmans-Karp** e **Push-Relabel**.

Após alguma análise, dado que teremos no pior caso um número de arcos , chegámos à conclusão que **Ford-Fulkerson** não poderá ser utilizado devido à sua complexidade O().

**Edmonds-Karp** pode ser usado para ser completo em O() graças ao uso de Procura em Largura Primeiro (BFS), embora demore bastante tempo para alguns casos.

**Push-Relabel** será o melhor caso para este tipo de problemas graças à sua complexidade de O() (ou até O()dependendo da implementação).

**Implementação e Resultados**

Implementámos o nosso programa em C pela familiaridade com a linguagem. No sistema Mooshak passámos a 11 em 16, falhando os restantes com Time Limit Exceeded por falta de optimização no algoritmo.

Escolhemos o **Edmonds-Karp** como algoritmo para o nosso projeto devido à sua facilidade de implementação (em comparação com o Push-Relabel). Tendo como complexidade teórica de O(), a nossa implementação segue a seguinte análise teórica:

* O() : para o aumento de fluxos
* O() : para a BFS

Durante a inicialização do Grafo, em particular na ligação entre os arcos, de forma a evitar uma ligação entre todos os vértices e demorar muito tempo entre Procuras em Largura Primeiro, decidimos ligar fluxos da fonte *s* para o destino *t* com a diferença entre os pesos l e c, criando apenas o arco com a diferença e somando ao valor de fluxo final o menor peso.

Por exemplo:

7

6

6

8

8

1

2

5

1

9

7

8

8

8

1

2

2

1

6

2

torna-se

Apresentamos de seguida 6 testes públicos aplicados ao nosso programa (testados num portátil com i5 de última geração a 2.3GHz, processados com a função clock(), com a flag de compilação -O3):

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Ficheiro** | **m** | **n** | **V** | **E (pior caso)** | **T(Criação)** | **T(Algoritmo)** | **T(Total)** |
| t100.in | 201 | 200 | 40 200 | 160 399 | 0.02 | 0.09 | 0.11 |
| t300.in | 601 | 600 | 360 600 | 1 441 199 | 0.13 | 10.44 | 10.57 |
| t500.in | 1 001 | 1 000 | 1 001 000 | 4 001 999 | 0.36 | 56.85 | 57.21 |
| ts100.in | 201 | 200 | 40 200 | 160 399 | 0.01 | 0.00 | 0.01 |
| ts300.in | 601 | 600 | 360 600 | 1 441 199 | 0.13 | 0.00 | 0.13 |
| ts500.in | 1 001 | 1 000 | 1 001 000 | 4 001 199 | 0.36 | 0.02 | 0.38 |

Devido à falta de dados suficientes para criar um gráfico correto, não conseguimos concluir se a execução do nosso programa chega a seguir as complexidades indicadas previamente.

**Referências**

* Introduction to Algorithms (3rd ed.), MIT Press and McGraw-Hill, ISBN 0-262-03293-7.
* Grafos: Slides Introdução Algoritmos e Estruturas de Dados, Profº Francisco Santos IST