Relatório 2º Projeto ASA 2020/2021

Grupo: al024
Alunos: António Coelho (ist195535) e Gustavo Aguiar (ist195587)

1 Descrição do Problema e da Solução

Pelo enunciado, temos que o pretendido é $min\{\sum_{i \in P_x} \mathrm{X}_i + \sum_{i \in P_y} \mathrm{Y}_i + \sum_{(i,j) \in P_x \times P_y} \mathrm{c}_{ij}\}$. Assim, por definição, este problema trata-se de um corte de capacidade mínima, pelo que com base no Teorema de Fluxo Máximo Corte Mínimo, modelámo-lo como um de rede de fluxo, onde a fonte e o sumidouro correspondem aos processadores X e Y, respetivamente, e os restantes vértices aos n processos do programa. Para além disso, sabendo que $(\sum_{i \in P_x} X_i + \sum_{i \in P_y} Y_i) \in O(n)$, temos que $|f^*| \in O(|V|)$. Justifica-se assim usar um algoritmo de caminhos de aumento, baseado no método de *Ford-Fulkerson*, sendo esses determinados por uma *BFS* - **Algoritmo de *Edmonds-Karp***.

Assim, como auxílio de resolução dos problemas encontrados utilizaram-se as seguintes referências:

- Análise teórica do algoritmo de *Edmonds-Karp*
- Implementação do algoritmo de *Edmonds-Karp* para determinar o fluxo máximo

2 Análise Teórica

A representação da rede residual em memória foi feita com recurso a uma matriz de capacidades residuais e a uma lista de adjacências - complexidade de espaço $\Theta(V^2)$ e $\Theta(V + E)$ - justificada pelo acesso O(1) às capacidades dos arcos na rede residual no decorrer do algoritmo e por um custo O(E) na *BFS*, respetivamente.

A solução visada para a resolução dos problemas apresentados consiste em 2 etapas: ler a rede de fluxo de *input* (1) e computar o fluxo máximo com recurso ao algoritmo de *Edmonds-Karp*(2).

Para (1) - leitura de *input* - usando `scanf`, ler os dados de entrada dentro de dois ciclos (n e k) a dependerem linearmente e quadraticamente de V para os custos entre os processos e os processadores e entre processos, respetivamente. Mais concretamente, $\Theta(V-2) = \mathrm{O}(V)$ para o primeiro ciclo e $\mathrm{O}(\frac{(V-2)(V-3)}{2}) = \mathrm{O}(V^2)$ para o segundo[1].

Para (2) – aplicar o algoritmo de *Edmonds-Karp* na rede residual – enquanto houver caminhos de aumento aplica-se a *BFS*[2], que corre em O(E). É sabido que o número de iterações de *Edmonds-Karp*, i.e., o número total de aumentos de fluxo é O(VE) -

[1] Numa matriz N × N existem $\frac{N(N-1)}{2}$ entradas na matriz triangular superior (sem contar com a diagonal).

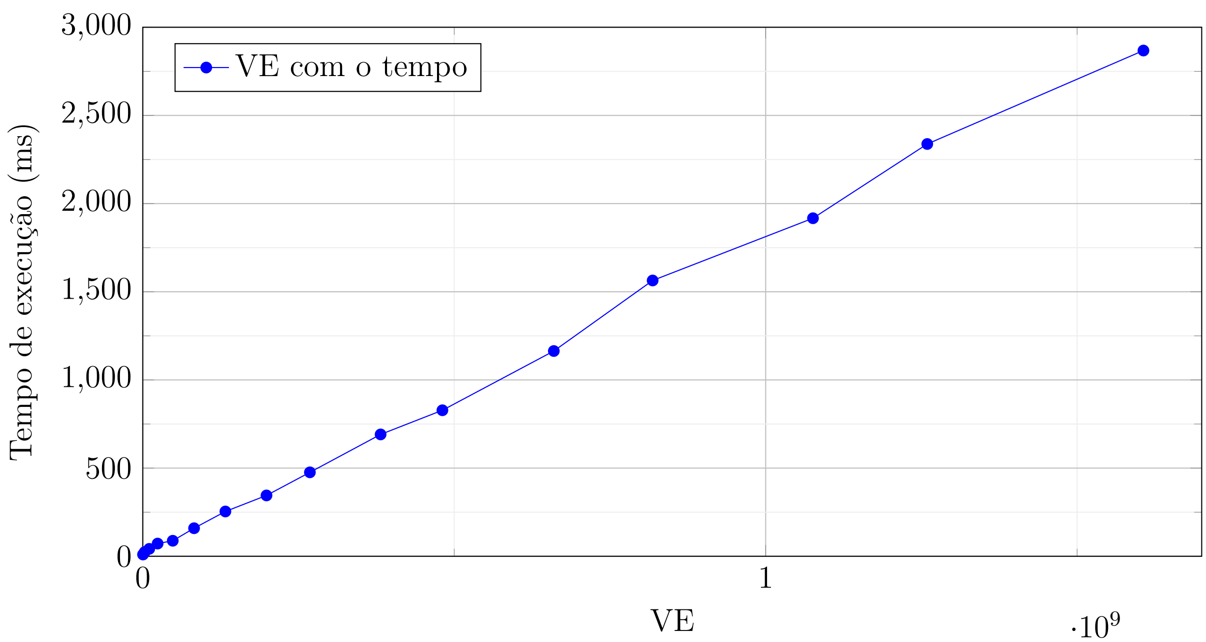
[2] Na rede de fluxo do contexto do problema, $|E| \gg |V|$.

existem O(E) pares de vértices que, para um arco (u, v), ficam críticos O(V) vezes, o que perfaz uma complexidade total de O(E)×O(EV) = O(E^2V). Contudo, mediante uma análise mais detalhada, notamos que o algoritmo de *Edmonds-Karp* $\in O(|f^*|\, E)$ (e $\in O(E^2V)$), e de acordo com o enunciado $|f^*| \in O(V)$, pelo que conseguimos apertar o limite assimptótico de (2) para $O(|f^*|\, E) \in O(VE)$. Concluímos assim que a solução geral dos problemas apresentados $\in$ **O(VE)**.

3 Avaliação Experimental dos Resultados

Para a devida análise do algoritmo implementado, utilizou-se um computador com processador *AMD Ryzen 5 5600X 6-Core* a 3.7 GHz, 16 GB de memória RAM e sistema operativo *Windows 10*.

Utilizou-se a ferramenta `gen2procs` fornecida pelo corpo docente para gerar redes de fluxo com capacidade por arco até 15 – mostra-se irrelevante utilizar capacidades superiores uma vez que a solução apresentada tem complexidade independente desse valor - e número de vértices entre 102 e 1602 aumentando de 100 em 100, perfazendo um tamanho de grafo[3] até à ordem de grandeza 10^6. Com o intuito de cronometrar o desempenho do algoritmo implementado utilizou-se a chamada de sistema `time` sobre o programa a correr nas redes geradas.



Por fim, registaram-se os valores obtidos da testagem e ajustou-se uma regressão linear com VE no eixo das abcissas, x, e o tempo (em ms) no eixo das ordenadas, y, que demonstra, experimentalmente, a veracidade da complexidade do algoritmo esperada teoricamente - **O(VE)**.

[3]Tamanho do grafo é dado por $|V| + |E|$.