# Relatório 2º Projeto ASA 2020/2021

**Grupo:** al034

**Aluno(s):** David Belchior (95550) e Diogo Santos (95562)

---

**Descrição do Problema e da Solução**

Considerámos o conjunto de processadores e processos, além das comunicações entre os últimos, uma **rede de fluxo**. Neste, os **vértices** correspondem aos **processadores** (**X** e **Y**) e aos n **processos** ($p_i$, i = 1, …, n), as **arestas** são compostas pelas **ligações de X aos processos e dos processos a Y** e por k **ligações entre processos que comunicam entre si** (arestas antiparalelas, simulando arestas não-dirigidas), com capacidades iguais a:

- $\mathbf{X_i}$ (Custo de execução de $p_i$ no processador X) nas arestas de X para $p_i$;
- $\mathbf{Y_i}$ (Custo de execução de $p_i$ no processador Y) nas arestas de $p_i$ para Y;
- $\mathbf{c_{ij}}$ (Custo de comunicação entre $p_i$ e $p_j$) nas arestas entre $p_i$ e $p_j$.

De modo a obter o custo mínimo da execução do programa, concluímos que tal seria possível tratando-o como um problema de corte mínimo, em que a resposta ao problema corresponderia, precisamente, à capacidade do corte mínimo.

Tendo em conta que **|V|** = n + 2 = **Θ(n)**, **|E|** = 2n + 2k = O(n + k) e **|f*|** ≤ n/2 = O(n) = O(|V|) (ver nota 1), optámos por uma implementação do **algoritmo de Edmonds-Karp** (baseado em caminhos de aumento), cuja complexidade pode ser apertada pela do método de Ford-Fulkerson: O(|f*| |E|) = O(|V| |E|) = O(n (n + k)), que é sempre menor ou igual do que a complexidade do algoritmo Relabel-to-Front (baseado em pré-fluxos): $O(|V|^3) = O(n^3)$. Tal deve-se ao facto de que |E| = O(|V|) = O(n), para grafos esparsos, e $|E| = O(|V|^2) = O(n^2)$ para grafos densos, e, portanto, no pior dos casos, o algoritmo de Edmonds-Karp tem complexidade $O(|V|^3) = O(n^3)$, tal como o algoritmo Relabel-to-Front.

Para efeitos de melhorias na eficiência do problema, recorremos apenas à rede residual, com a capacidade do corte mínimo = fluxo máximo da rede a ser obtida somando o fluxo de cada caminho de aumento obtido pela BFS feita em cada iteração do algoritmo de Edmonds-Karp.

Para permitir um tratamento eficiente do refluxo, colapsamos, na rede residual, a aresta de refluxo de $(p_i,p_j)$ e a aresta $(p_j,p_i)$ numa só, sendo o fluxo correspondente a essa aresta a soma das arestas colapsadas. Além disso, quando diminuímos o fluxo residual numa aresta, aumentamos o mesmo valor na aresta de sentido contrário.

Nota 1: $(\Sigma_{i \in p_X} X_i + \Sigma_{i \in p_Y} Y_i) \in O(n)$, pelo que o fluxo máximo corresponderá, no máximo, a metade desse valor (quando a soma dos custos de ligação a cada processador é igual) $\in$ O(n).

# Relatório 2º Projeto ASA 2020/2021

**Grupo:** al034

**Aluno(s):** David Belchior (95550) e Diogo Santos (95562)

---

### Análise Teórica

O nosso programa efetua os seguintes passos:

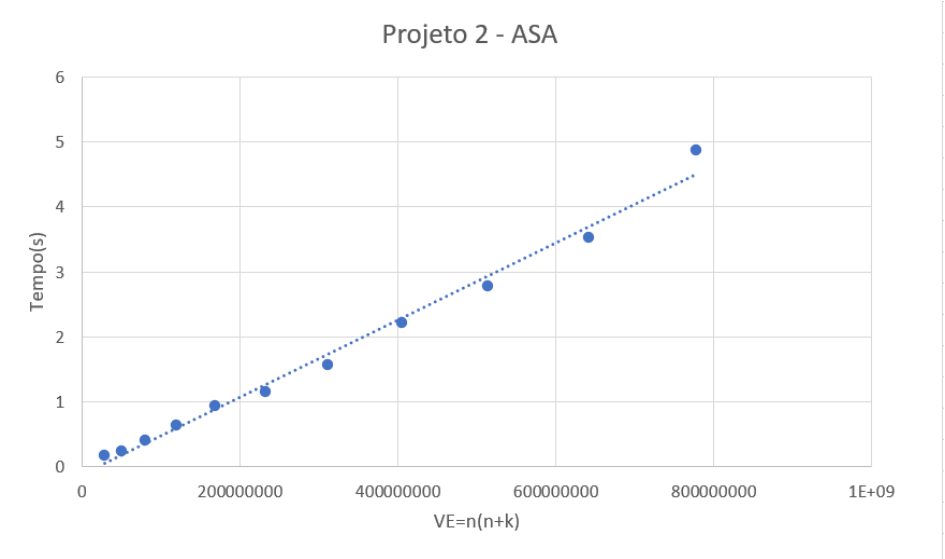
- **Leitura dos dados de entrada**: leitura do input e criação da rede residual por via de uma lista de adjacências. Este processo depende apenas do número de arestas do grafo, logo a sua complexidade é **O(|E|) = O(n + k)**.
- **Aplicação do algoritmo Edmonds-Karp: O(|V| |E|) = O(n (n + k))**.
- **Apresentação dos dados**: **O(1)**.

**Complexidade global da solução** (em função de n e k): O(n+k) + O(n(n+k)) + O(1) = **O(n(n+k))**.

### Avaliação Experimental dos Resultados

Usando o programa gen2procs.cpp fornecido pelos docentes, geramos um total de 11 grafos com 500 processos iniciais e um tamanho incremental de 100 até 1500 processos, utilizando um valor fixo para o valor máximo de execução de qualquer processo de 50.

Após correr o projeto sobre estes 11 grafos obtivemos o seguinte gráfico:




Com recurso a uma aproximação linear, concluímos que o tempo de execução confirma as nossas previsões teóricas, possuindo uma complexidade de **O(|V| |E|) = O(n (n + k))**.

Especificações do computador em que foram corridos os testes: CPU Intel i7- 8565U e 16 GB de RAM a 2400 MHz com Ubuntu (WSL1).