Universidade do Minho

Braga, Portugal

TRABALHO PRÁTICO 2 - RELATÓRIO

Problema de Fluxo Máximo numa Rede

Investigação Operacional

Departamento de Informática

Engenharia Informática 2024/25

Equipa de Trabalho:

A106932 - Luís António Peixoto Soares

A106856 - Tiago Silva Figueiredo

A104365 - Fábio Magalhães

A106936 - Duarte Escairo Brandão Reis Silva

A104704 - Inês Ferreira Ribeiro

30 Abril 2025

Índice

1. Introdução

No âmbito da unidade curricular de Investigação Operacional, foi-nos proposto este trabalho prático, que tem como objetivo a resolução de um problema de maximização do fluxo numa rede.

Com este trabalho pretendemos conhecer e perceber o modelo em questão, de modo a que consigamos aplica-lo na resolução do problema proposto.

Neste relatório será apresentado de forma detalhada o modelo utilizado, bem como a sua aplicação no nosso caso concreto, juntamente com uma formulação do problema e respetiva resolução. Por fim iremos validar o resultado obtido.

2. Problema

2.1. Questão 0 - Dados

De acordo com o enunciado, os dados deste problema são determinados em função do maior número de inscrição entre os elementos do grupo, que corresponde ao **106936**. Assim sendo, os vértices de origem e destino são calculados em função de k, k esse que é determinado através do resto da divisão de DE por 7, i.e. $k = DE \bmod(7)$.

De acordo com o nosso número, DE tem valor 36, sendo o nosso valor de k igual a 1. Desta forma os vértices de origem e destinos a serem utilizados são: $O = 1$ e $D = 5$. Para o cálculo da capacidade de cada vértice, foi mais uma vez utilizado o número de inscrição, sendo os valores das capacidades dos vértices dados pela seguinte tabela:

Vértice	**Capacidade**
1	*ilimitada*
2	100
3	100
4	40
5	*ilimitada*
6	100

Tabela 1: Capacidade dos vértices

O grafo resultante da aplicação das capacidades dos vértices é o seguinte:

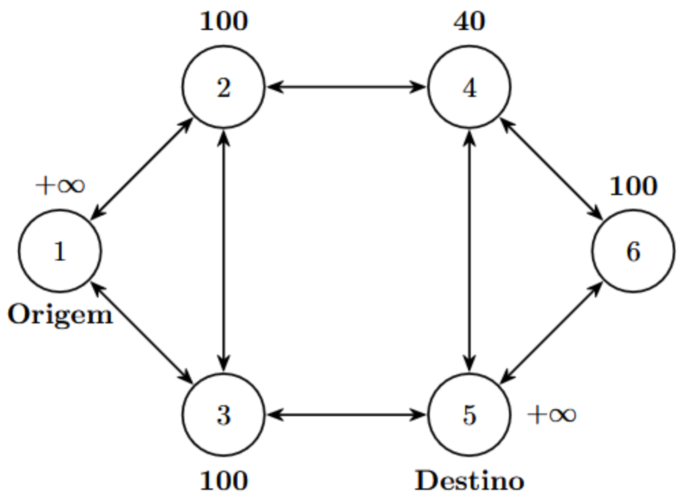

Figura 1: Grafo com as respetivas capacidades identificadas

2.2. Questão 1 - Formulação do Problema

2.2.1. Explicação do problema e alguns conceitos

Para uma melhor compreensão da resolução do problema apresentado, vamos propor um exemplo mais prático que torne o problema menos abstrato, e vamos também abordar e explicar alguns conceitos que consideramos relevantes para a compreensão do trabalho.

Suponhamos que o grafo do problema representa uma rede de transporte desde os campos de petróleo, localizados na origem (O), até à refinaria, localizada no destino (D). Neste exemplo, os vértices representam pontos na rede, e os arcos representam os troços do oleoduto. Para este caso em concreto, o fluxo irá ser o petróleo transportado entre os diversos pontos da rede.

A capacidade de um vértice representa o fluxo máximo que pode entrar nele mesmo, ou seja, no nosso exemplo representa a quantidade máxima de petróleo que pode entrar num determinado ponto, por exemplo no ponto da rede que representa o vértice 2 do grafo, só podem entrar 100 unidades de petróleo.

Neste caso vamos considerar a capacidade dos arcos como infinita, tal como é dito no enunciado, e vamos ignorar os valores relativos à oferta/procura em cada um dos vértices, já que estes não são referidos.

Assim sendo, neste exemplo, ao resolver este problema do fluxo máximo na rede, vamos descobrir qual é a quantidade máxima de petróleo que pode ser envidada deste os campos até à refinaria, independentemente do caminho seguido pelo petróleo no oleoduto.

Alguns conceitos importantes já referidos, e que continuarão a ser referidos são os conceitos de custo, capacidade e oferta/procura.

- **Custo**: o custo unitário de um arco refere-se ao valor gasto no transporte de uma unidade de fluxo por esse mesmo arco. No enunciado não são dados os valores dos custos nos arcos, por isso assume-se que são nulos;
- **Capacidade**:
  - **Arco**: a capacidade de um arco representa a quantidade máxima de fluxo que o mesmo pode transportar. Tal como referido no enunciado, considera-se a capacidade dos arcos do problema como sendo infinita;
  - **Vértice**: a capacidade de um vértice representa a quantidade máxima de fluxo que pode entrar num determinado vértice. Os valores das capacidades dos vértices do problema foram calculado na secção anterior;
- **Oferta/Procura**: a oferta/procura de um vértice é o fluxo que é produzido, e que será transportado ao longo da rede (oferta), ou o fluxo que será consumido quando recebido do resto da rede (procura). Mais uma vez não são apresentados valores de oferta/procura para os vértices do grafo do problema, por isso consideramos que o valor fosse nulo.

2.2.2. Formulação matemática

Antes da resolução concreta do problema, vamos apresentar a definição formal que envolve um problema de fluxos em rede.

Seja $G = (V, A)$ um grafo orientado de uma rede, que descreve um problema de fluxos em rede, onde V é o conjunto dos vértices do grafo, e $A \subseteq V \times V$ é o conjunto dos arcos do grafo. Para cada vértice $j \in V$, pode estar associada uma oferta/procura, b_j, que é um valor positivo caso se trate de uma oferta, ou um valor negativo caso se trate de uma procura. Para cada arco $(i, j) \in A$, pode estar associado um custo unitário de transporte, c_{ij}, e ainda uma capacidade, u_{ij}.

Para a resolução de problemas de minimização do custo, é utilizado o modelo geral, sendo este um modelo de programação linear:

$$\min : \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{1}$$

Suj. a

$$-\sum_{(i,j)\in A} x_{ij} + \sum_{(j,i)\in A} x_{ji} = b_j, \forall j \in V \tag{2}$$

$$0 \leq x_{ij} \leq u_{ij}, \forall (i, j) \in A \tag{3}$$

Variáveis de decisão:

x_{ij} : fluxo de um único tipo de entidades no arco orientado (i, j)

A restrição apresentada em (2) designa-se por restrição de conservação de fluxo, e mostra que, para cada vértice, a diferença entre os fluxos de saída e entrada deve ser igual à oferta/procura, nesse arco.

A restrição apresentada em (3) designa-se por restrição de capacidade, e diz-nos que o fluxo em todos os arcos não pode ser negativo e que não pode ultrapassar o valor da capacidade de cada arco.

Contudo, para a resolução de problemas de maximização de fluxo (como é o caso do nosso problema), é utilizado outro modelo. Neste modelo, o objetivo é, logicamente, maximizar o fluxo entre os vértices de origem (O) e destino (D), e esse fluxo é representado pela variável f.

Neste modelo são novamente utilizadas as restrições de conservação de fluxo e de capacidade já apresentadas, que são representadas pelas equações (1) e (2) respetivamente.

$$\max : f \tag{4}$$

Suj. a

$$-\sum_{(i,j)\in A} x_{ij} + \sum_{(j,i)\in A} x_{ji} = \begin{cases} f \text{ se } i = O \\ 0 \text{ se } i \in V \setminus \{O, D\} \\ -f \text{ se } i = D \end{cases} \tag{5}$$

$$0 \leq x_{ij} \leq u_{ij}, \forall (i,j) \in A \tag{6}$$

Neste caso, a restrição de conservação de fluxo em (5), foi modificada em relação à original, pois é necessário garantir que o balanço de fluxo na origem é igual a uma oferta de f unidades ($b_O = f$), e que no destino é igual a uma procura de f unidades ($b_D = -f$).

2.2.3. Construção do modelo

O problema consiste em determinar o fluxo máximo entre os vértices origem $O = 1$ e destino $D = 5$, num grafo onde os vértices possuem capacidades, exceto O e D. Como os modelos de fluxo máximo trabalham com capacidades nos arcos (e não nos vértices), é necessário transformar o grafo original numa rede direcionada com capacidades apenas nos arcos. A abordagem é a seguinte:

1. **Transformação da capacidades dos vértices em arcos:**

Cada vértice com capacidade finita (2, 3, 4, 6) é dividido em dois nós: um de entrada (v_{in}) e um de saída (v_{out}). Um arco direcionado é adicionado entre v_{in} e v_{out}, com capacidade igual à do vértice original.

Exemplo: Vértice 2 (capacidade 100) → Arco $2_{\text{in}} \to 2_{\text{out}}$ com capacidade 100.

2. **Tratamento dos arcos bidirecionais:**

Os arcos do grafo original permitem fluxo em ambos os sentidos e têm capacidade infinita. Cada arco (u, v) é substituída por dois arcos direcionados: $u \to v$ e $v \to u$, ambos com capacidade $+\infty$ (representada por um valor numérico elevado, e.g. 1000).

3. **Redirecionamento de conexões:**

Todas os arcos que chegam a um vértice v no grafo original agora chegam a v_{in} e todas os arcos que saem de v no grafo original agora saem de v_{out}.

Modelo de Fluxos em Rede (Grafo Transformado):

- **Vértices:**
  - 1 (origem) e 5 (destino) permanecem inalterados.
  - Vértices $2, 3, 4, 6$ são substituídos por pares 2_{in}; 2_{out};3_{in};3_{out}; etc.

- **Arcos e Capacidades:**
  - Arcos internos (de capacidade finita): ($2_{in} \to 2_{out}$: 100); ($3_{in} \to 3_{out}$: 100); ($4_{in} \to 4_{out}$: 40); ($6_{in} \to 6_{out}$: 100)
  - Arcos originais (capacidade infinita):
    - Para cada arco (u, v) original, adicionar dois arcos: $u \to v$: 1000 e $v \to u$: 1000.
    - ***Exemplo***: Se há uma arco entre 2 e 3 no grafo original, no modelo transformado teremos: $2_{out} \to 3_{in}$ de valor 1000 e $3_{out} \to 2_{in}$ de valor 1000.

Para além disto, considera-se que o custo em cada arco é 0 (no grafo o valor estará omitido), com a exceção do arco $(8, 1)$ (que representam o destino e a origem como veremos mais à frente), que terá valor -1. Este passo é necessário porque o algoritmo utilizado pelo RELAX4 é de minimização de custos, e desta forma ele tentará maximizar o fluxo entre a origem e o destino, resolvendo o nosso problema.

Estas transformações resultam num novo grafo:

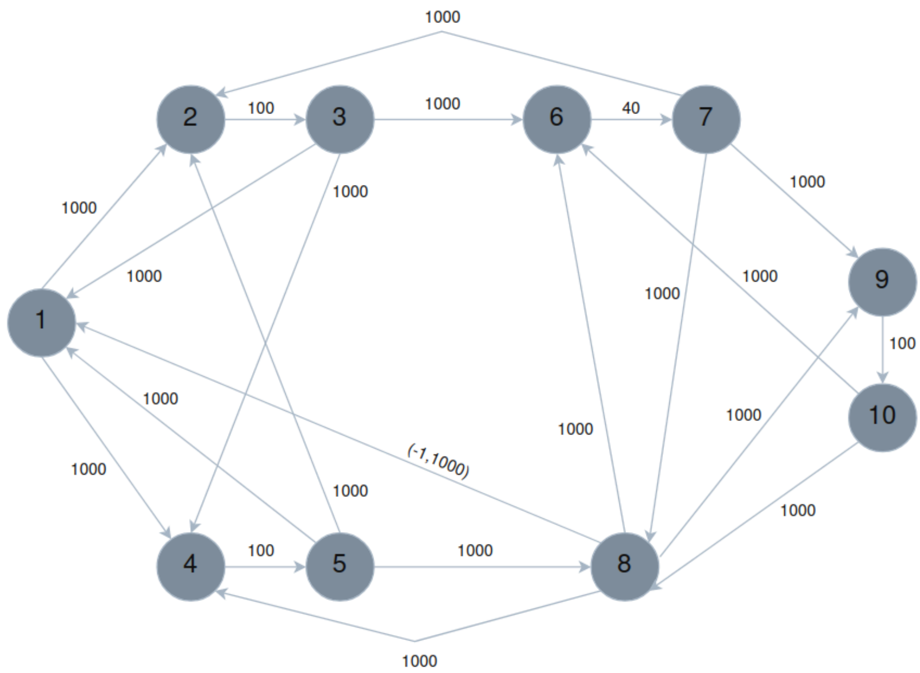

Figura 2: Grafo original após aplicadas as transformações

Este modelo permite utilizar um solver de fluxo máximo padrão, como o RELAX4, pois todas as restrições estão convertidas em capacidades de arcos. O fluxo máximo será limitado pelas capacidades dos vértices, agora representadas como arcos internos.

2.3. Questão 2 - Ficheiro de Input do Relax4

O ficheiro de input utilizado no RELAX4 foi elaborado de acordo com o grafo transformado do problema. Nas duas primeiras linhas encontram-se o número de vértices e o número de arcos respetivamente, de seguida uma linha para cada arco com a informação da sua origem, destino, custo e capacidade. Por fim, uma linha para cada um dos vértices contendo a informação relativa à oferta/procura.

```
10
21
 1  2 0  1000
 1  4 0  1000
 2  3 0  100
 3  1 0  1000
 3  4 0  1000
 3  6 0  1000
 4  5 0  100
 5  1 0  1000
 5  2 0  1000
 5  8 0  1000
 6  7 0  40
 7  2 0  1000
 7  8 0  1000
 7  9 0  1000
 8  1 -1 1000
 8  4 0  1000
 8  6 9  1000
 8  9 0  1000
 9 10 0  100
10  6 0  1000
10  8 0  1000
0
0
0
0
0
0
0
0
0
0
```

2.4. Questão 3 - Ficheiro de Output do Relax4

O resultado do RELAX4 para o input utilizado.

```
s -140.
f 1 2 100
f 1 4 100
f 2 3 100
f 3 1 60
f 3 4 0
f 3 6 40
f 4 5 100
f 5 1 0
f 5 2 0
f 5 8 100
f 6 7 40
f 7 2 0
f 7 8 40
f 7 9 0
```

```
f 8 1 140
f 8 4 0
f 8 6 0
f 8 9 0
f 9 10 0
f 10 6 0
f 10 8 0
```

2.5. Questão 4 - Solução Ótima

A solução dada pelo RELAX4, dá-nos não apenas o fluxo máximo entre a origem e o destino, como também o fluxo em cada um dos arcos.

Da primeira linha da solução ($s - 140.$), podemos concluir que o fluxo máximo entre os vértices 1 e 5 (origem e destino respetivamente), é de 140 unidades. Este valor negativo é explicado a partir do valor de custo -1 atribuído anteriormente entre o arco que liga o destino à origem, sendo o único arco que afeta a função objetivo.

Através da solução ótima obtida é possível obter um grafo que mostra o fluxo máximo entre a origem e destino, bem como o fluxo em cada vértice.

Por exemplo a linha **f 3 6 40**, pode ser interpretada como sendo um arco do vértice 3 para o vértice 6 $(3, 6)$, com um fluxo de 40 unidades. Aplicando este raciocínio para todas as linhas é possível construir o seguinte grafo.

Nota: As linhas cujo fluxo é nulo (0) não serão desenhadas.

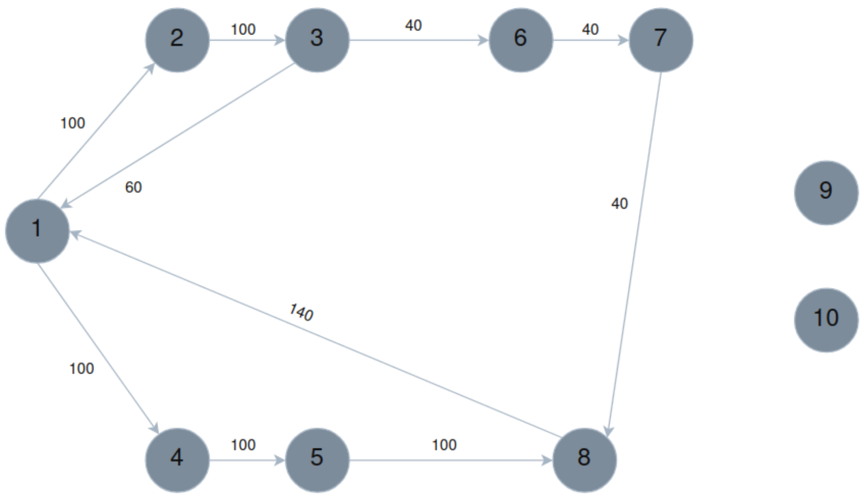

Figura 3: Grafo obtido a partir da solução ótima

Apesar deste grafo conter a solução ótima, este não representa o grafo original do problema, mas antes o grafo transformado a partir deste. Assim sendo é necessário desfazer as transformações previamente feitas de forma a podermos analisar a solução no grafo original.

Será necessário juntar o vértices separados em vértices de entrada e vértice de saída $(v_{\text{in}}, v_{\text{out}})$, e ainda retirar o arco que liga o destino à origem, colocando o fluxo que entra e sai na origem e no destino respetivamente. Estas transformações reversas dão origem ao seguinte grafo:

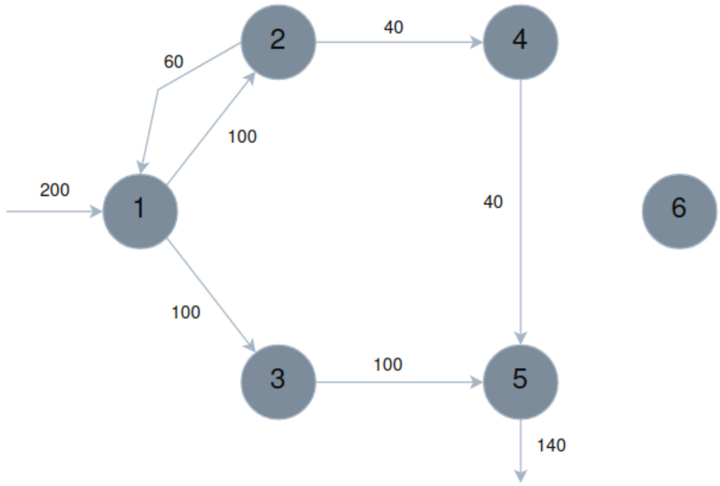

Figura 4: Grafo após a sua normalização

Para simplificar um pouco mais o grafo, podemos ainda retirar o arco $(2, 1)$ com fluxo de 60 unidades, pois este devolve o excesso de fluxo que não pode ser transmitido, e para isso basta enviar um fluxo de 40 unidades no arco $(1, 2)$ já que o vértice 4 tem capacidade de apenas 40 unidades. Finalmente, o grafo final que contém a solução ótima é o seguinte:

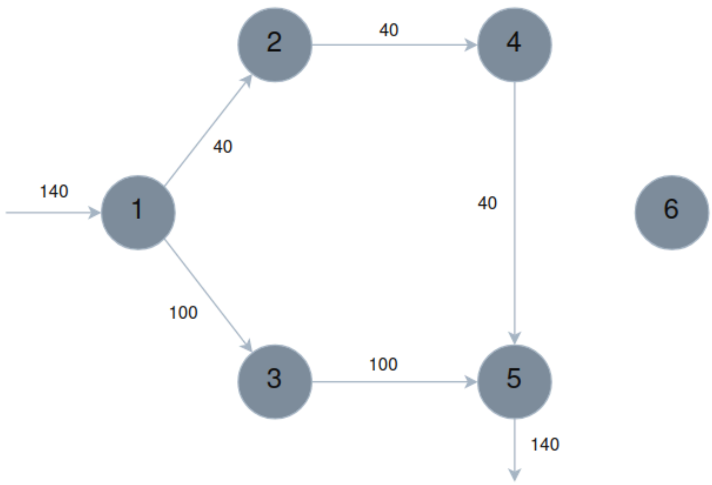

Figura 5: Grafo que representa a solução ótima

Agora estamos em condições de interpretar e analisar a solução ótima obtida. Concluímos então que o fluxo máximo entre os vértices 1 e 5 é de 140 unidades. 100 dessas unidades percorrem o caminho $1 \rightarrow 3 \rightarrow 5$ e as restantes 40 unidades percorrem o caminho $1 \rightarrow 2 \rightarrow 4 \rightarrow 5$.

2.6. Questão 5 - Identificação do Corte Mínimo

O corte mínimo corresponde ao conjunto de arestas cuja remoção isola o vértice de origem (1) do vértice de destino (5). Neste caso, existe mais do que um conjunto, sendo um deles constituído pelos vértices $\{(1 \rightarrow 3), (1 \rightarrow 2)\}$, com um custo total de 140.

2.7. Questão 6 - Validação do Modelo

Após termos calculado o valor da solução ótima, procedemos à validação do modelo. Começamos por verificar se a solução obtida fazia sentido, e não violava nenhum pressuposto que pudesse ter sido esquecido quando desenvolvemos o ficheiro de input para o RELAX4. Verificou-se que:

1. Em todos os vértices, o fluxo que entra nunca ultrapassa a sua capacidade. No vértice 2, que tem capacidade 100, entram 40 unidades, no vértice 4 que tem capacidade 40, entram 40 unidades e por fim, o vértice 3 que tem capacidade 100 entram 100 unidades;

2. Não existe desperdício em nenhum vértice, ou seja, todo o fluxo que entra num determinado vértice sai por completo.

Para complementar a validação, o Teorema do Fluxo Máximo-Corte Mínimo afirma que numa rede, o valor máximo do fluxo desde um ponto até outro é igual ao corte mínimo, que corresponde ao conjunto de arestas que quando removidas impossibilitam o fluxo entre a origem e o destino. Este teorema valida o nosso modelo pois o fluxo máximo e o corte mínimo têm ambos valor 140.

Para além disto, aplicamos o Algoritmo de Ford-Fulkerson escrito em python, que calcula o fluxo máximo de um grafo.

```python
class Graph:
    def __init__(self, size):
        self.adj_matrix = [[0] * size for _ in range(size)]
        self.size = size
        self.vertex_data = [''] * size

    def add_edge(self, u, v, c):
        self.adj_matrix[u][v] = c

    def add_vertex_data(self, vertex, data):
        if 0 <= vertex < self.size:
            self.vertex_data[vertex] = data

    def bfs(self, s, t, parent):
        visited = [False] * self.size
        queue = []
        queue.append(s)
        visited[s] = True

        while queue:
            u = queue.pop(0)
            for ind, val in enumerate(self.adj_matrix[u]):
                if not visited[ind] and val > 0:
                    queue.append(ind)
```

```
                    visited[ind] = True
                    parent[ind] = u
                    if ind == t:
                        return True
        return False

    def fordFulkerson(self, source, sink):
        parent = [-1] * self.size
        max_flow = 0

        while self.bfs(source, sink, parent):
            path_flow = float("Inf")
            s = sink
            while s != source:
                path_flow = min(path_flow, self.adj_matrix[parent[s]][s])
                s = parent[s]

            max_flow += path_flow

            v = sink
            while v != source:
                u = parent[v]
                self.adj_matrix[u][v] -= path_flow
                self.adj_matrix[v][u] += path_flow
                v = parent[v]

            path = []
            v = sink
            while v != source:
                path.append(v)
                v = parent[v]
            path.append(source)
            path.reverse()
            path_names = [self.vertex_data[node] for node in path]
            print("Augmenting path:", " -> ".join(path_names), ", Flow:",
path_flow)

        return max_flow

g = Graph(10)

vertex_labels = {
    0: '1 (O)',
    1: '2_in', 2: '2_out',
    3: '3_in', 4: '3_out',
    5: '4_in', 6: '4_out',
    7: '6_in', 8: '6_out',
    9: '5 (D)'
}
for v, label in vertex_labels.items():
    g.add_vertex_data(v, label)

INF = 1000
g.add_edge(0, 1, INF)
g.add_edge(0, 3, INF)
```

```
g.add_edge(1, 2, 100)
g.add_edge(2, 0, INF)
g.add_edge(2, 3, INF)
g.add_edge(2, 5, INF)
g.add_edge(3, 4, 100)
g.add_edge(4, 0, INF)
g.add_edge(4, 1, INF)
g.add_edge(4, 9, INF)
g.add_edge(5, 6, 40)
g.add_edge(6, 1, INF)
g.add_edge(6, 7, INF)
g.add_edge(6, 9, INF)
g.add_edge(9, 0, INF)
g.add_edge(9, 3, INF)
g.add_edge(9, 5, INF)
g.add_edge(9, 7, INF)
g.add_edge(7, 8, 100)
g.add_edge(8, 5, INF)
g.add_edge(8, 9, INF)

# Add reverse edges for residual graph
for i in range(g.size):
    for j in range(g.size):
        if g.adj_matrix[i][j] > 0 and g.adj_matrix[j][i] == 0:
            g.add_edge(j, i, 0)

source = 0  # 1 (O)
sink = 9  # 5 (D)

print("Computing maximum flow from", vertex_labels[source], "to",
vertex_labels[sink])
max_flow = g.fordFulkerson(source, sink)
print("\nThe maximum possible flow is", max_flow)
```

A solução do algoritmo para o nosso problema foi, tal como esperado, um fluxo máximo de 140 unidades.

```
Computing maximum flow from 1 (O) to 5 (D)
Augmenting path: 1 (O) -> 3_in -> 3_out -> 5 (D) , Flow: 100
Augmenting path: 1 (O) -> 2_in -> 2_out -> 4_in -> 4_out -> 5 (D) , Flow: 40

The maximum possible flow is 140
```

Desta forma, conseguimos validar a nossa solução.

3. Conclusão

Ao longo do desenvolvimento deste trabalho, não apenas conseguimos resolver o problema de forma correta, com também ficamos a compreender melhor um dos modelo de resolução de problemas de fluxo máximo numa rede. Através da utilização do software RELAX4 fomos capazes de encontrar a solução ótima e, consequentemente, construir o grafo que a representa. Por fim, com a construção do grafo e a descoberta da solução ótima, fomos capazes de identificar o corte mínimo e validar o nosso modelo.

4. Bibliografia

[1] Investigação Operacional. Relax4 - Software de Otimização de Redes, Blackboard E-learning, 2024.

[2] Algoritmo de Ford-Fulkerson. - https://www.w3schools.com/dsa/dsa_algo_graphs_fordfulkerson.php