AGÊNCIA DE VIAGENS

Desenho de Algoritmos | Grupo 122 | Entrega 2 | 03/06/2022

André Gabriel Correia Vieira - up202004159@edu.fe.up.pt
Francisca Horta Guimarães - up202004229@edu.fe.up.pt
Pedro Joaquim Alves Oliveira - up202004324@edu.fc.up.pt

PROBLEMA

Este projeto tem como objetivo implementar um sistema capaz de apoiar a gestão de pedidos para transporte de grupos de pessoas de um local de origem para um local de destino.

Para isso, as soluções algorítmicas a desenvolver devem ser o mais eficiente possível, utilizando para esse propósito, algoritmos abordados em contexto de sala de aula.

Para este projeto, pretende-se particularmente explorar os seguintes cenários:

<u>Cenário 1</u> – grupos que não se separam

- 1.1 – maximizar a dimensão do grupo e indicar um encaminhamento;
- 1.2 – maximizar a dimensão do grupo e minimizar o número de transbordos.

<u>Cenário 2</u> – grupos que podem separar-se

- 2.1 - determinar um encaminhamento para um grupo, dada a sua dimensão;
- 2.2 - corrigir um encaminhamento para que a dimensão do grupo possa aumentar;
- 2.3 - determinar a dimensão máxima do grupo e um encaminhamento;
- 2.4 - determinar quando é que o grupo se reuniria novamente no destino;
- 2.5 - indicar o tempo máximo de espera e os locais em que haveria elementos que esperam esse tempo.

CENÁRIO 1 - FORMALIZAÇÃO

Este cenário processa deslocamento de grupos que não se separam e está dividido em duas alíneas com os seguintes objetivos:

- 1.1 – maximizar a dimensão do grupo e indicar um encaminhamento;
- 1.2 – maximizar a dimensão do grupo e minimizar o número de transbordos.

➢ **1.1:**

Dados:

G = (V, A, {s, t}, c, d) – Grafo dirigido com um nó origem [s] (source) e um nó destino [t] (target) e com valores (capacidade [c] e duração [d]) nos ramos. (V – vértices [nodes]. A – adjacências [edges]).

Variáveis de Decisão:

c – capacidade máxima;

list(path) – lista com os caminhos possíveis.

Maximizar a função objetivo:

Caminho com a capacidade máxima.

CENÁRIO 1 - FORMALIZAÇÃO

Este cenário processa deslocamento de grupos que não se separam e está dividido em duas alíneas com os seguintes objetivos:

- 1.1 – maximizar a dimensão do grupo e indicar um encaminhamento;
- 1.2 – maximizar a dimensão do grupo e minimizar o número de transbordos.

➢ **1.2:**

Dados:
G = (V, A, {s, t}, c, d) – Grafo dirigido com um nó origem [s] (source) e um nó destino [t] (target) e com valores (capacidade [c] e duração [d]) nos ramos. (V – vértices [nodes]. A – adjacências [edges]).

Variáveis de Decisão:
c – capacidade máxima;
layovers – número de transbordos
list(path) – lista com os caminhos possíveis.

Maximizar a função objetivo:
Resultado pareto-ótimo maximizando a capacidade e minimizando o número de transbordos.

Sujeito às restrições:
layovers < max(layovers)

CENÁRIO 1 – ALGORITMOS RELEVANTES

Neste cenário baseamo-nos fortemente nos algoritmos de caminhos de capacidade máxima.

Utilizamos, para esse efeito, uma adaptação do Algoritmo de Dijkstra com recurso a uma Heap de Máximo, sendo a segunda alínea (1.2) uma variação da primeira (1.1), executada várias vezes diminuindo o número de transbordos a cada execução.

CENÁRIO 1 – ANÁLISE DE COMPLEXIDADE

- Complexidade Temporal:

  - 1.1:
    - ➢ T(N) = O((|V| + |A|) log2 |V|)
  - 1.2:
    - ➢ T(N) = O(R(1.1)*(|V| + |A|) log2 |V|)

  Sendo R(1.1) o resultado da 1.1, ou seja o resultado da primeira execução da função pareto.

- Complexidade Espacial:

  - 1.1:
    - ➢ S(N) = O(1)
  - 1.2:
    - ➢ S(N) = O(1)

V – Vértices

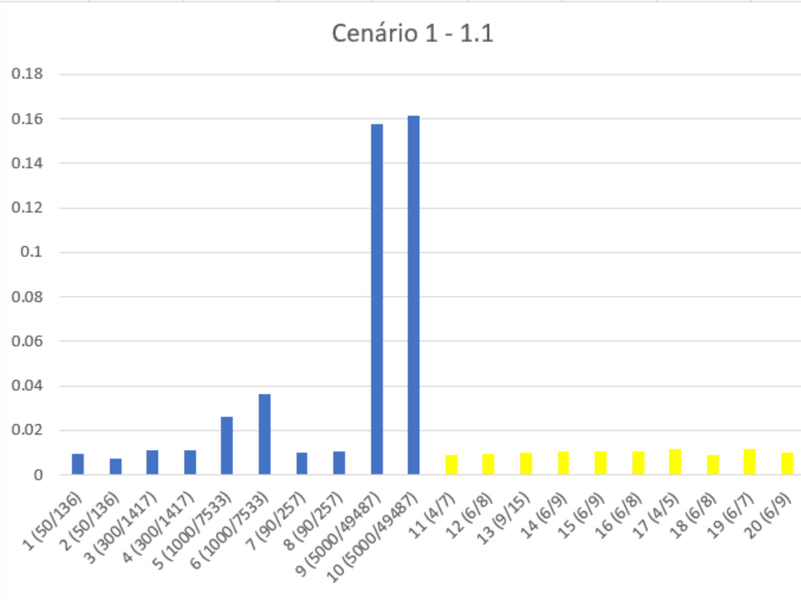
A – Adjacências (Edges)

CENÁRIO 1 - AVALIAÇÃO EMPÍRICA

- 1.1:

número de vértices (N) e o número de ramos (R)

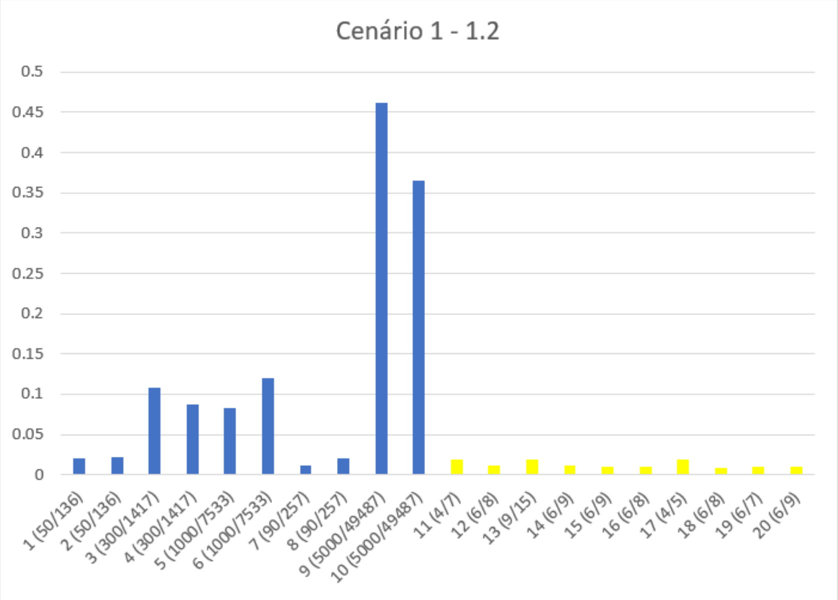
Dias (N/R)	segundos	nanossegundos
1 (50/136)	0.0094163	9416300
2 (50/136)	0.0075401	7540100
3 (300/1417)	0.0110603	11060300
4 (300/1417)	0.0110516	11051600
5 (1000/7533)	0.0261649	26164900
6 (1000/7533)	0.0360983	36098300
7 (90/257)	0.0098036	9803600
8 (90/257)	0.010315	10315000
9 (5000/49487)	0.1577312	157731200
10 (5000/49487)	0.1615296	161529600
11 (4/7)	0.008696	8696000
12 (6/8)	0.0097366	9736600
13 (9/15)	0.0101466	10146600
14 (6/9)	0.0107459	10745900
15 (6/9)	0.0104784	10478400
16 (6/8)	0.0108077	10807700
17 (4/5)	0.0114991	11499100
18 (6/8)	0.0091402	9140200
19 (6/7)	0.0113985	11398500
20 (6/9)	0.0101932	10193200



- 1.2:

número de vértices (N) e o número de ramos (R)

Dias (N/R)	segundos	nanossegundos
1 (50/136)	0.0197524	19752400
2 (50/136)	0.0221318	22131800
3 (300/1417)	0.1073923	107392300
4 (300/1417)	0.0874734	87473400
5 (1000/7533)	0.0832901	83290100
6 (1000/7533)	0.1205275	120527500
7 (90/257)	0.0107667	10766700
8 (90/257)	0.0204262	20426200
9 (5000/49487)	0.461377	461377000
10 (5000/49487)	0.3657193	365719300
11 (4/7)	0.0182048	18204800
12 (6/8)	0.0107091	10709100
13 (9/15)	0.0184846	18484600
14 (6/9)	0.0115084	11508400
15 (6/9)	0.0094201	9420100
16 (6/8)	0.0095137	9513700
17 (4/5)	0.0192903	19290300
18 (6/8)	0.0085887	8588700
19 (6/7)	0.0091989	9198900
20 (6/9)	0.0101822	10182200



Azul – Datasets fornecidos
Amarelo – Datasets adicionais

CENÁRIO 2 - FORMALIZAÇÃO

Este cenário processa grupos que se podem separar e está dividido em cinco alíneas com os seguintes objetivos:

- 2.1 – determinar um encaminhamento para um grupo, dada a sua dimensão;
- 2.2 – corrigir um encaminhamento para que a dimensão do grupo possa aumentar;
- 2.3 – determinar a dimensão máxima do grupo e um encaminhamento;
- 2.4 – determinar quando é que o grupo se reuniria novamente no destino;
- 2.5 – indicar o tempo máximo de espera e os locais em que haveria elementos que esperam esse tempo.

➢ **2.1 / 2.2 / 2.3 :**

Dados:

G = (V, A, {s, t}, c, d) – Grafo dirigido com um nó origem [s] (source) e um nó destino [t] (target) e com valores (capacidade [c] e duração [d]) nos ramos. (V – vértices [nodes]. A – adjacências [edges]).

Variáveis de Decisão:

f – fluxo na rede

Maximizar a função objetivo:

$|\mathrm{f}| = \sum_{v\,\in V} f(s,v) = \sum_{v\,\in V} f(v,t)$

Sujeito às restrições:

$f(u,v) = -f(v,u),\ \ para\, u,v\ \in V;$

$f(u,v) \leq c(u,v), para\, u,v\ \in V;$ (fluxo não excede capacidade)

$\sum_{v\in V} f(u,v) = 0, para\, u\ \in V\ \ \setminus \{s,t\};$ (conservação do fluxo)

CENÁRIO 2 - FORMALIZAÇÃO

Este cenário processa grupos que se podem separar e está dividido em cinco alíneas com os seguintes objetivos:

- 2.1 – determinar um encaminhamento para um grupo, dada a sua dimensão;
- 2.2 – corrigir um encaminhamento para que a dimensão do grupo possa aumentar;
- 2.3 – determinar a dimensão máxima do grupo e um encaminhamento;
- 2.4 – determinar quando é que o grupo se reuniria novamente no destino;
- 2.5 – indicar o tempo máximo de espera e os locais em que haveria elementos que esperam esse tempo.

➢ **2.4:**

Dados:
G = (V, A, {s, t}, c, d) – Grafo dirigido com um nó origem [s] (source) e um nó destino [t] (target) e com valores (capacidade [c] e duração [d]) nos ramos. (V – vértices [nodes]. A – adjacências [edges]).

Variáveis de Decisão:
ES[V] – Duração mínima para a reunião do grupo.

Maximizar a função objetivo:
ESij = max{ ESki + Dki | (k, i) ∈ A}.

Sujeito às restrições:
Grau[V] = 0.

CENÁRIO 2 - FORMALIZAÇÃO

Este cenário processa grupos que se podem separar e está dividido em cinco alíneas com os seguintes objetivos:

- 2.1 – determinar um encaminhamento para um grupo, dada a sua dimensão;
- 2.2 – corrigir um encaminhamento para que a dimensão do grupo possa aumentar;
- 2.3 – determinar a dimensão máxima do grupo e um encaminhamento;
- 2.4 – determinar quando é que o grupo se reuniria novamente no destino;
- 2.5 – indicar o tempo máximo de espera e os locais em que haveria elementos que esperam esse tempo.

- **2.5 :**

Dados:

G = (V, A, {s, t}, c, d) – Grafo dirigido com um nó origem [s] (source) e um nó destino [t] (target) e com valores (capacidade [c] e duração [d]) nos ramos. (V – vértices [nodes]. A – adjacências [edges]).

Variáveis de Decisão:

FL[V] – Folga livre em cada nó.

Maximizar a função objetivo:

FLij = min{ESjk | (j, k) ∈ A} − ES[ij] + dij

Sujeito às restrições:

- (ES[j]- (ES[ij] + dij)) > FL[j]
- FL[j] > 0;

CENÁRIO 2 – ALGORITMOS RELEVANTES

Neste cenário baseamo-nos fortemente nos algoritmos que resolvem problemas de fluxo máximo.

Para a realização das primeiras três alíneas deste cenário, considerámos o Algoritmo de Ford-Fulkerson com recurso ao Algoritmo Breadth-First Search (BFS) (implementação Edmonds-Karp). Usando o algoritmo BFS melhoramos a complexidade do nosso algoritmo visto que é escolhido sempre um caminho com um número mínimo de arestas.

Nas últimas duas alíneas utilizamos o método do Caminho Crítico mais propriamente usando a função "earliestStart" para a alínea 2.4.

Na alínea 2.5 reutilizamos a função "earliestStart" para, desta vez, calcular a Folga Livre.

CENÁRIO 2 – ANÁLISE DE COMPLEXIDADE

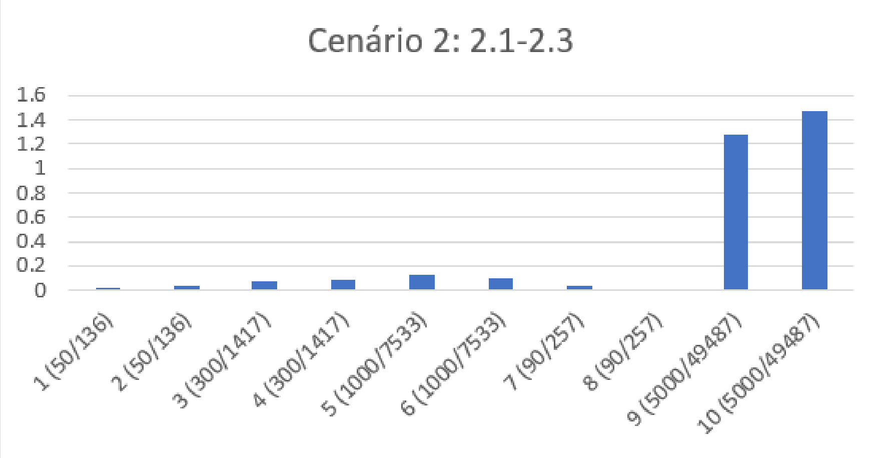
- Complexidade Temporal:
  - 2.1 / 2.2 / 2.3
    - ➢ T(N) = O ($|V| \times |A|^2$);
  - 2.4 / 2.5
    - ➢ T(N) = O($|V| \times |A|$);
- Complexidade Espacial:
  - 2.1 / 2.2 / 2.3
    - ➢ S(N) = O($|V|^2$);
  - 2.4 / 2.5
    - ➢ S(N) = O($|V|$);

V – Vértices
A – Adjacências (Edges)

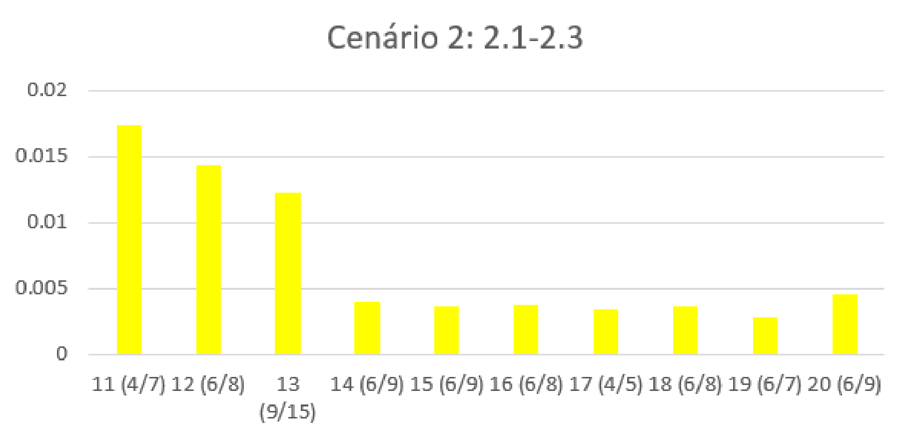
CENÁRIO 2 - AVALIAÇÃO EMPÍRICA

- 2.1/2.2/2.3:

número de vértices (N) e o número de ramos (R)		
Dias (N/R)	segundos	nanossegundos
1 (50/136)	0.0169748	16974800
2 (50/136)	0.0394224	39422400
3 (300/1417)	0.0775512	77551200
4 (300/1417)	0.0823532	82353200
5 (1000/7533)	0.1217627	121762700
6 (1000/7533)	0.0945124	94512400
7 (90/257)	0.0384314	38431400
8 (90/257)	0.0112093	11209300
9 (5000/49487)	1.2761803	1276180300
10 (5000/49487)	1.4796682	1479668200



número de vértices (N) e o número de ramos (R)		
Dias (N/R)	segundos	nanossegundos
11 (4/7)	0.0173413	17341300
12 (6/8)	0.0143643	14364300
13 (9/15)	0.0122545	12254500
14 (6/9)	0.0040171	4017100
15 (6/9)	0.0035624	3562400
16 (6/8)	0.0037259	3725900
17 (4/5)	0.0033912	3391200
18 (6/8)	0.0035812	3581200
19 (6/7)	0.0028089	2808900
20 (6/9)	0.0045066	4506600

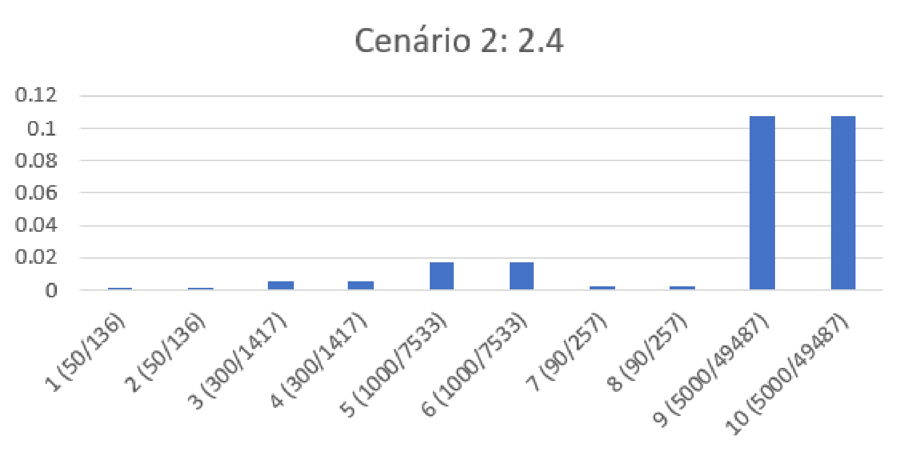


Azul – Datasets fornecidos
Amarelo – Datasets adicionais

CENÁRIO 2 - AVALIAÇÃO EMPÍRICA

- 2.4:

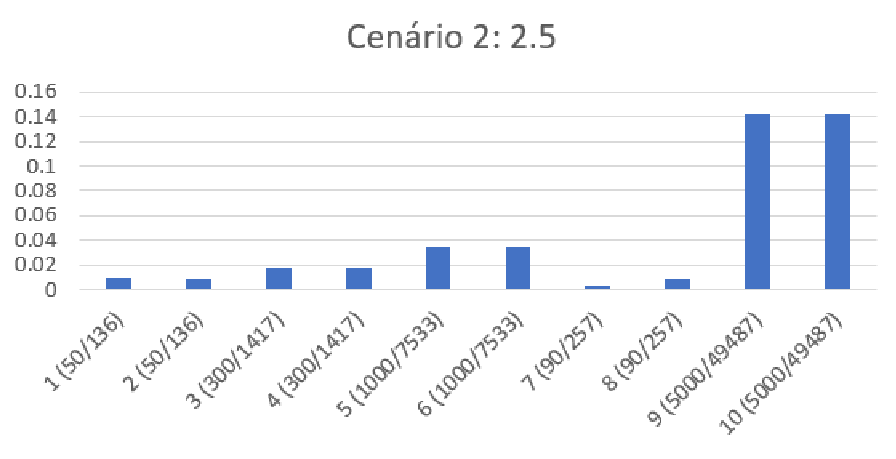
número de vértices (N) e o número de ramos (R)		
Dias (N/R)	segundos	nanossegundos
1 (50/136)	0.0020009	2000900
2 (50/136)	0.0020112	2011200
3 (300/1417)	0.00523	5230000
4 (300/1417)	0.0052875	5287500
5 (1000/7533)	0.0172393	17239300
6 (1000/7533)	0.0175172	17517200
7 (90/257)	0.0021587	2158700
8 (90/257)	0.0023738	2373800
9 (5000/49487)	0.1070063	107006300
10 (5000/49487)	0.1071891	107189100



Azul – Datasets fornecidos

- 2.5:

número de vértices (N) e o número de ramos (R)		
Dias (N/R)	segundos	nanossegundos
1 (50/136)	0.0097313	9731300
2 (50/136)	0.0088853	8885300
3 (300/1417)	0.0180378	18037800
4 (300/1417)	0.0181509	18150900
5 (1000/7533)	0.0351847	35184700
6 (1000/7533)	0.0344331	34433100
7 (90/257)	0.0028793	2879300
8 (90/257)	0.0092258	9225800
9 (5000/49487)	0.1426397	142639700
10 (5000/49487)	0.1426761	142676100



SOLUÇÃO ALGORÍTMICA A DESTACAR

Destacamos a solução algorítmica utilizada na alínea 1.2 ("pareto") porque achamos que foi uma boa e eficiente adaptação do algoritmo utilizado na alínea 1.1, reutilizando a maioria do seu código, acrescentando apenas uma contagem de transbordos para calcular um máximo que diminui em cada chamada de forma a encontrar as soluções pareto-ótimas.

Destacamos também a solução algorítmica utilizada nas alíneas 2.1-2.3 dada a sua relevância e peso na totalidade do projeto e importância no estudo de algoritmos.

CONCLUSÃO

- **Principais Dificuldades Encontradas**
  - Interpretação dos objetivos para algumas das alíneas do projeto;
  - Melhorar a forma de implementação de alguns algoritmos;
  - Utilização das formulas matemáticas para função objetivo.
- **Esforço de Cada Elemento do Grupo**
  - André Vieira: 30%
  - Francisca Guimarães: 40%
  - Pedro Oliveira: 30%