Digressão pelas enumerações de trajectos e de caminhos

.

Marta Pascoal

Departamento de Matemática da FCTUC e INESC-Coimbra

Resumo

A enumeração de trajectos entre um par de nós de uma rede é uma generalização do problema do trajecto com o menor custo em que se pretende listar trajectos por ordem não decrescente de custo.
Aqui se apresentam as principais classes de métodos para a enumeração de trajectos.
Mostra-se como estes métodos podem ser utilizados para determinar trajectos satisfazendo a restrições adicionais e como podem ser adaptados para enumerar trajectos sem nós repetidos, ou seja, caminhos.

Plano

- Notação e definições
- O trajecto com o menor custo
  - Algoritmo de rotulação
- Os K trajectos com o menor custo
  - Algoritmos de rotulação
  - Algoritmos de apagamentos
  - Algoritmos de desvios
- Passeatas *vs.* restrições adicionais
- Os K caminhos com o menor custo
  - Algoritmos de desvios
- Complexidade computacional
- Fim. . .

Notação e definições

Dados:

- Rede $(\mathcal{N}, \mathcal{A})$: conjunto de nós $\mathcal{N}$, conjunto de arcos $\mathcal{A}$.
- Custo do arco (i, j), $c_{ij} \in \mathbb{R}_0^+$.
- Nó inicial s e nó terminal t, com $s \neq t$.

Trajecto de i para j em $(\mathcal{N}, \mathcal{A})$ é $\langle i = v_1, v_2, \ldots, v_{\ell_p} = j \rangle$, onde:

- $v_k \in \mathcal{N}$, para $k \in \{1, \ldots, \ell_p\}$;
- $(v_k, v_{k+1}) \in \mathcal{A}$, para $k \in \{1, \ldots, \ell_p - 1\}$.

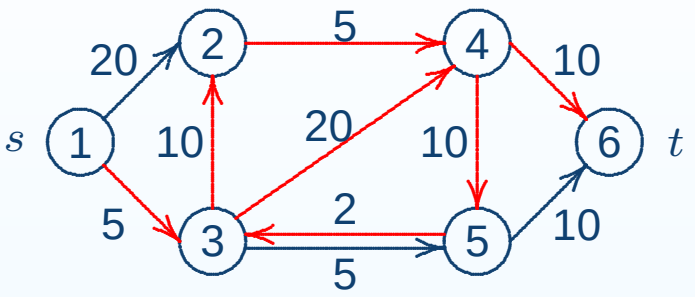
Caminho de i para j em $(\mathcal{N}, \mathcal{A})$ é um trajecto sem nós repetidos.

Conjunto de trajectos em $(\mathcal{N}, \mathcal{A})$ de i para j: $\mathcal{P}_{ij}$. ($\mathcal{P} = \mathcal{P}_{st}$.)

Custo do trajecto p:

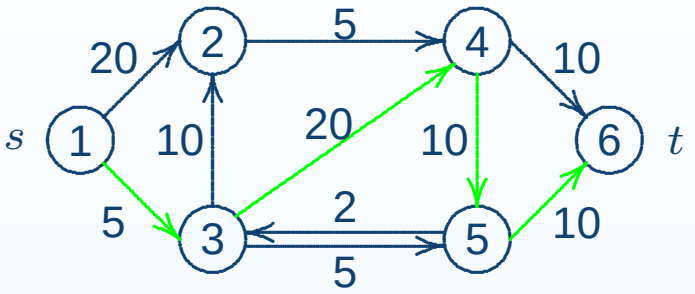
$$\begin{array}{rccc} c: & \cup_{i,j \in \mathcal{N}} \mathcal{P}_{ij} & \rightarrow & \mathbb{R} \\ & p & \mapsto & \sum_{(x,y) \in p} c_{xy} \end{array}$$

Notação e definições



Trajecto: $p = \langle 1, 3, 4, 5, 3, 2, 4, 6 \rangle$, $c(p) = 62$

Notação e definições



Trajecto: $p = \langle 1, 3, 4, 5, 3, 2, 4, 6 \rangle$, $c(p) = 62$

Caminho: $q = \langle 1, 3, 4, 5, 6 \rangle$, $c(q) = 45$

O trajecto com o menor custo

O trajecto com o menor custo

Objectivo: Determinar $p^* \in \mathcal{P}$ tal que $c(p^*) \leq c(p)$, para todo $p \in \mathcal{P}$.

Algoritmo de rotulação

$X \leftarrow \{s\}$

$p_{si} \leftarrow \emptyset$ ($c(p_{si}) \leftarrow +\infty$), para todo $i \in \mathcal{N} - \{s\}$; $c(p_{ss}) \leftarrow 0$

Enquanto $X \neq \emptyset$ **Faz**:

$i \leftarrow$ nó de X

Para $j \in \mathcal{N}$ **tal que** $(i,j) \in \mathcal{A}$ **Faz:**

Se $c(p_{si} \diamond \langle i,j \rangle) < c(p_{sj})$ **Então:**

$p_{sj} \leftarrow p_{si} \diamond \langle i,j \rangle$

Se $j \notin X$ **Então** $X \leftarrow X \cup \{j\}$

$X \leftarrow X - \{i\}$

O trajecto com o menor custo

Objectivo: Determinar $p^* \in \mathcal{P}$ tal que $c(p^*) \leq c(p)$, para todo $p \in \mathcal{P}$.

Algoritmo de rotulação

$X \leftarrow \{s\}$

$p_{si} \leftarrow \emptyset$ ($c(p_{si}) \leftarrow +\infty$), para todo $i \in \mathcal{N} - \{s\}$; $c(p_{ss}) \leftarrow 0$

Enquanto $X \neq \emptyset$ **Faz**:

$i \leftarrow$ nó de X

Para $j \in \mathcal{N}$ **tal que** $(i,j) \in \mathcal{A}$ **Faz:**

Se $c(p_{si} \diamond \langle i,j \rangle) < c(p_{sj})$ **Então:**

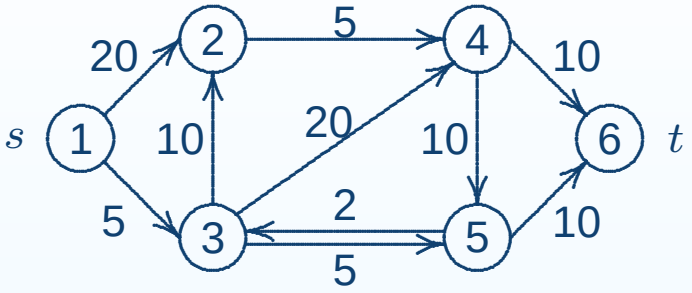
$p_{sj} \leftarrow p_{si} \diamond \langle i,j \rangle$

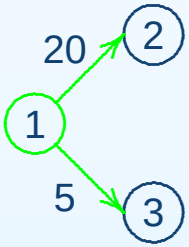
Se $j \notin X$ **Então** $X \leftarrow X \cup \{j\}$

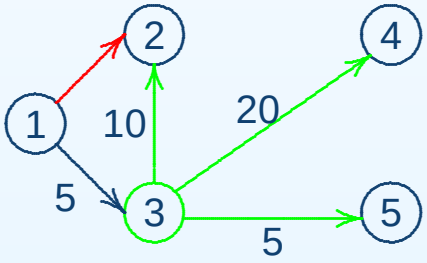
$X \leftarrow X - \{i\}$

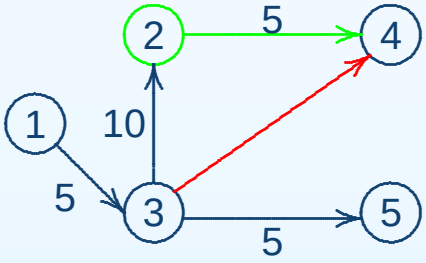
- Rótulos não definitivos: Bellman-Ford-Moore (X é um FIFO)
  Pape-Levit (X é um FIFO ou um LIFO)
  ...
- Rótulos definitivos: Dijkstra (i é escolhido em X de modo a p_{si} ser óptimo)

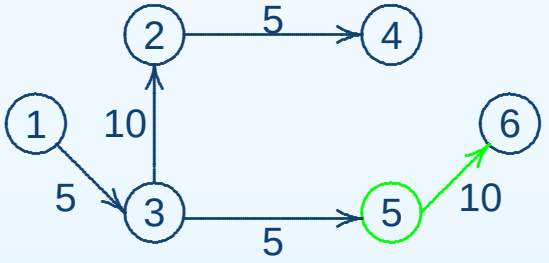
O trajecto com o menor custo











π_i	0	20	5	0	15	5	25	10	0	15	5	20	10	0	15	5	20	10	20
ξ_i	-	1	1	-	3	1	3	3	-	3	1	2	3	-	3	1	2	3	5

Os K trajectos com o menor custo

Os K trajectos com o menor custo

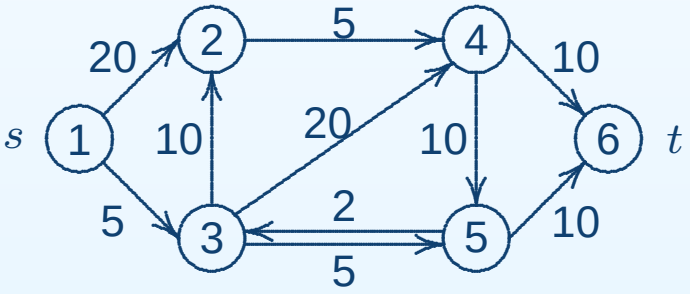
Dado $K \in \mathbb{N}$ pretende-se encontrar $\mathcal{P}_K = \{p_1, \ldots, p_K\} \subseteq \mathcal{P}$, onde:

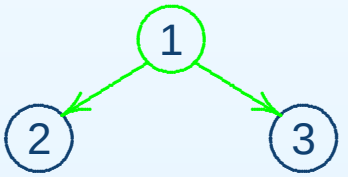
- $c(p_k) \leq c(p_{k+1})$, para todo $k \in \{1, \ldots, K-1\}$;
- $c(p_K) \leq c(p)$, para todo $p \in \mathcal{P} - \mathcal{P}_K$;
- p_k é determinado antes de p_{k+1}, para todo $k \in \{1, \ldots, K-1\}$.

Algoritmos de rotulação

Generalizações de algoritmos de rotulação para o problema do trajecto com o menor custo

- Determinar os K trajectos com o menor custo de s para $i \in \mathcal{N}$.
- Generalizar as formas do algoritmo de rotulação para $K = 1$.
- Associar a cada $i \in \mathcal{N}$ o K-rótulo, $(\pi_i^1, \ldots, \pi_i^K)$ e $(\xi_i^1, \ldots, \xi_i^K)$.



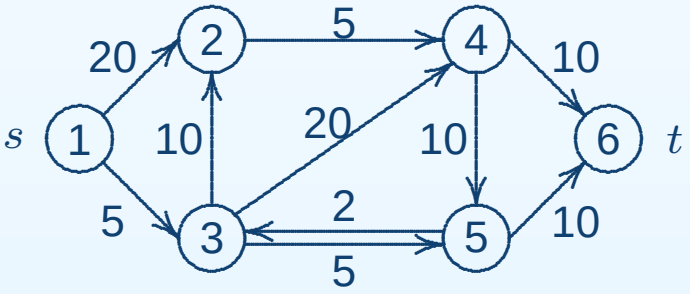


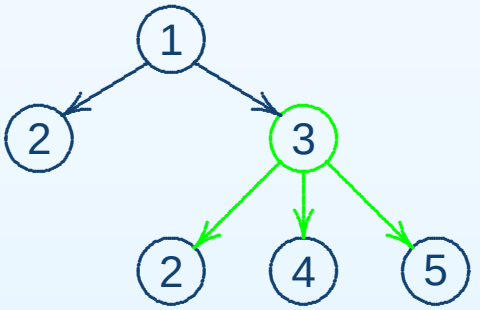
π_i		ξ_i	
0	-	-	-
20	-	1	-
5	-	1	-
-	-	-	-
-	-	-	-
-	-	-	-

Algoritmos de rotulação

Generalizações de algoritmos de rotulação para o problema do trajecto com o menor custo

- Determinar os K trajectos com o menor custo de s para $i \in \mathcal{N}$.
- Generalizar as formas do algoritmo de rotulação para $K = 1$.
- Associar a cada $i \in \mathcal{N}$ o K-rótulo, $(\pi_i^1, \ldots, \pi_i^K)$ e $(\xi_i^1, \ldots, \xi_i^K)$.



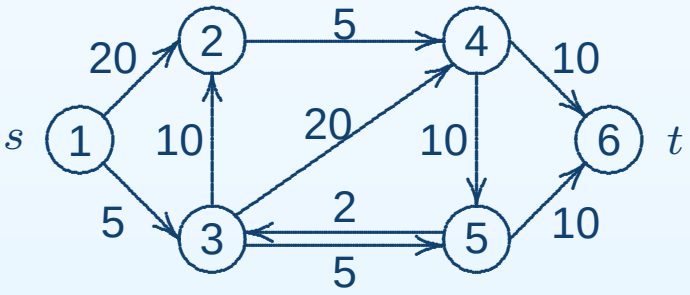


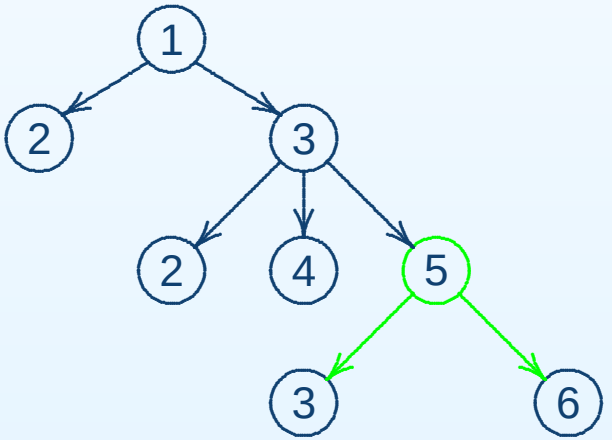
π_i		ξ_i	
0	-	-	-
20	15	1	3
5	-	1	-
25	-	3	-
10	-	3	-
-	-	-	-

Algoritmos de rotulação

Generalizações de algoritmos de rotulação para o problema do trajecto com o menor custo

- Determinar os K trajectos com o menor custo de s para $i \in \mathcal{N}$.
- Generalizar as formas do algoritmo de rotulação para $K = 1$.
- Associar a cada $i \in \mathcal{N}$ o K-rótulo, $(\pi_i^1, \ldots, \pi_i^K)$ e $(\xi_i^1, \ldots, \xi_i^K)$.



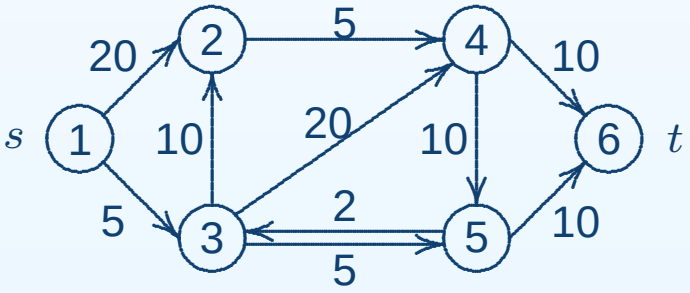


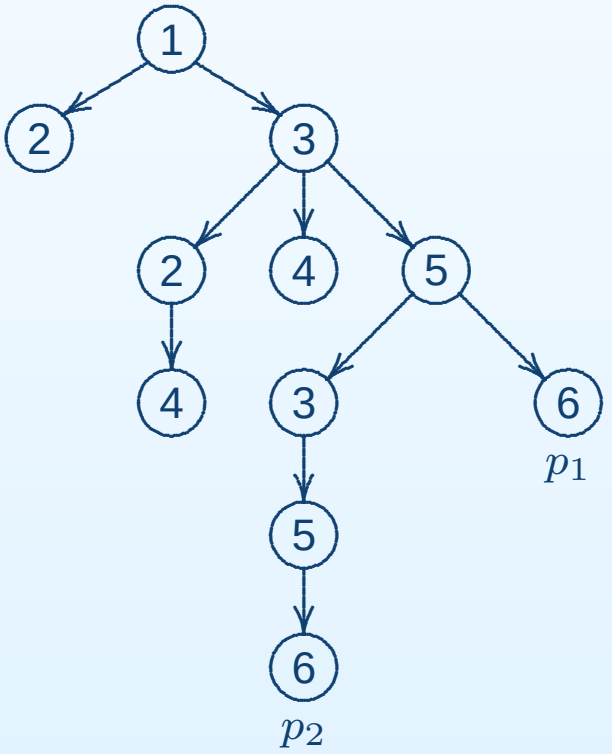
π_i		ξ_i	
0	-	-	-
20	15	1	3
5	12	1	5
25	-	3	-
10	-	3	-
20	-	5	-

Algoritmos de rotulação

Generalizações de algoritmos de rotulação para o problema do trajecto com o menor custo

- Determinar os K trajectos com o menor custo de s para $i \in \mathcal{N}$.
- Generalizar as formas do algoritmo de rotulação para $K = 1$.
- Associar a cada $i \in \mathcal{N}$ o K-rótulo, $(\pi_i^1, \ldots, \pi_i^K)$ e $(\xi_i^1, \ldots, \xi_i^K)$.





π_i		ξ_i	
0	-	-	-
20	15	1	3
5	12	1	5
25	20	3	2
10	17	3	3
20	27	5	5

Algoritmos de apagamentos

Algoritmo de apagamento successivo de trajectos em $(\mathcal{N}, \mathcal{A})$

Determinar p_1

$(\mathcal{N}_1, \mathcal{A}_1) \leftarrow (\mathcal{N}, \mathcal{A})$

Para $k \in \{2, \ldots, K\}$ **Faz:**

$(\mathcal{N}_k, \mathcal{A}_k) \leftarrow$ rede obtida da anterior por remoção de p_{k-1}

$p' \leftarrow$ trajecto com o menor custo em $(\mathcal{N}_k, \mathcal{A}_k)$

$p_k \leftarrow$ trajecto correspondente a p' em $(\mathcal{N}, \mathcal{A})$

Algoritmos de apagamentos

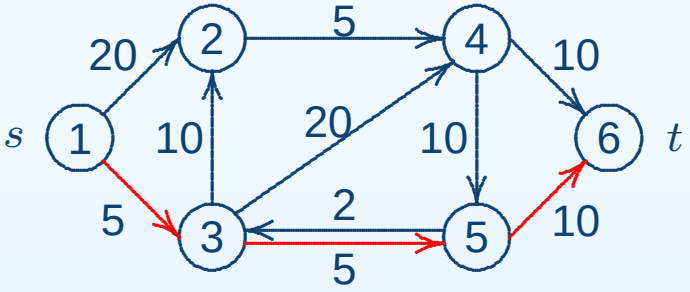
Apagamento do trajecto $p = \langle s = v_1, v_2 \ldots, t = v_{\ell_p} \rangle$

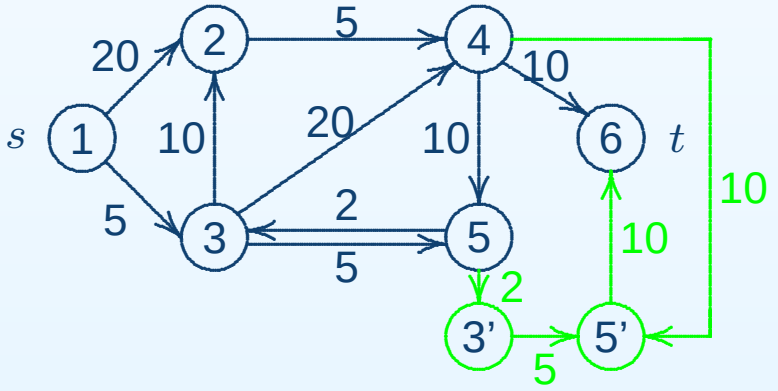
Criar cópia dos nós intermédios de p, $v'_2, \ldots, v'_{\ell_p - 1}$

Criar cópia dos arcos intermédios de p, $(v'_2, v'_3), \ldots, (v'_{\ell_p - 2}, v'_{\ell_p - 1})$

Trocar $(v_{\ell_p - 1}, t)$ por $(v'_{\ell_p - 1}, t)$

Criar cópia (j, v'_i) dos arcos $(j, v_i) \in \mathcal{A}$, para $v_i \in p$





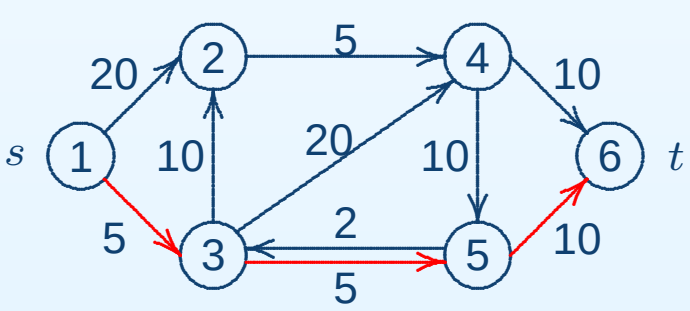
$p_1 = \langle 1, 3, 5, 6 \rangle$

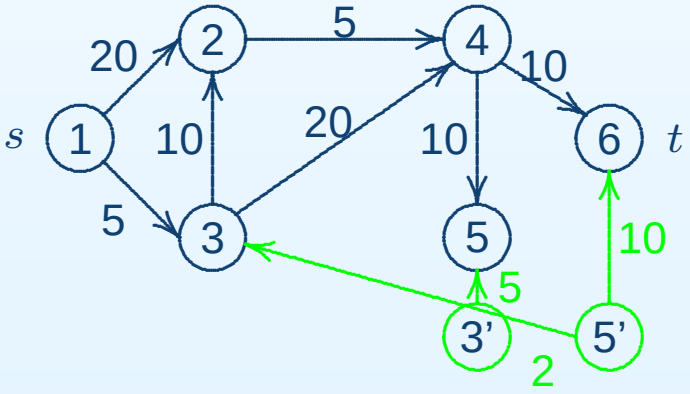
$p' \mapsto p_2 = \langle 1, 3, 5, 3, 5, 6 \rangle$

Algoritmos de apagamentos

O algoritmo original foi sujeito a sucessivos melhoramentos . . .

- Rotula nós duplicados por análise dos arcos incidentes $\Rightarrow$ Evita o aumento do número de arcos da rede e reduz o espaço de memória.
- Ordena o conjunto dos arcos incidentes nos nós por ordem não crescente de custo $\Rightarrow$ Reduz o número de operações.





$p_1 = \langle 1, 3, 5, 6 \rangle$

$p' \mapsto p_2 = \langle 1, 3, 5, 3, 5, 6 \rangle$

Algoritmos de desvios

Seja X um conjunto de trajectos candidatos a p_k.

Algoritmo de desvios

Determinar p_1

$X \leftarrow \{p_1\}$

Para $k \in \{2, \ldots, K\}$ **Faz:**

$p_k \leftarrow$ trajecto com o menor custo em X

$X \leftarrow X - \{p_k\}$

Analisar nós de p_k e gerar novos trajectos (desvios) que se armazenam em X (candidatos a p_j, com $j > k$)

Desvio de um trajecto

Trajecto com o menor custo que desvia de $p = \langle v_1, \ldots, v_{\ell_p} \rangle$ no nó v_i:

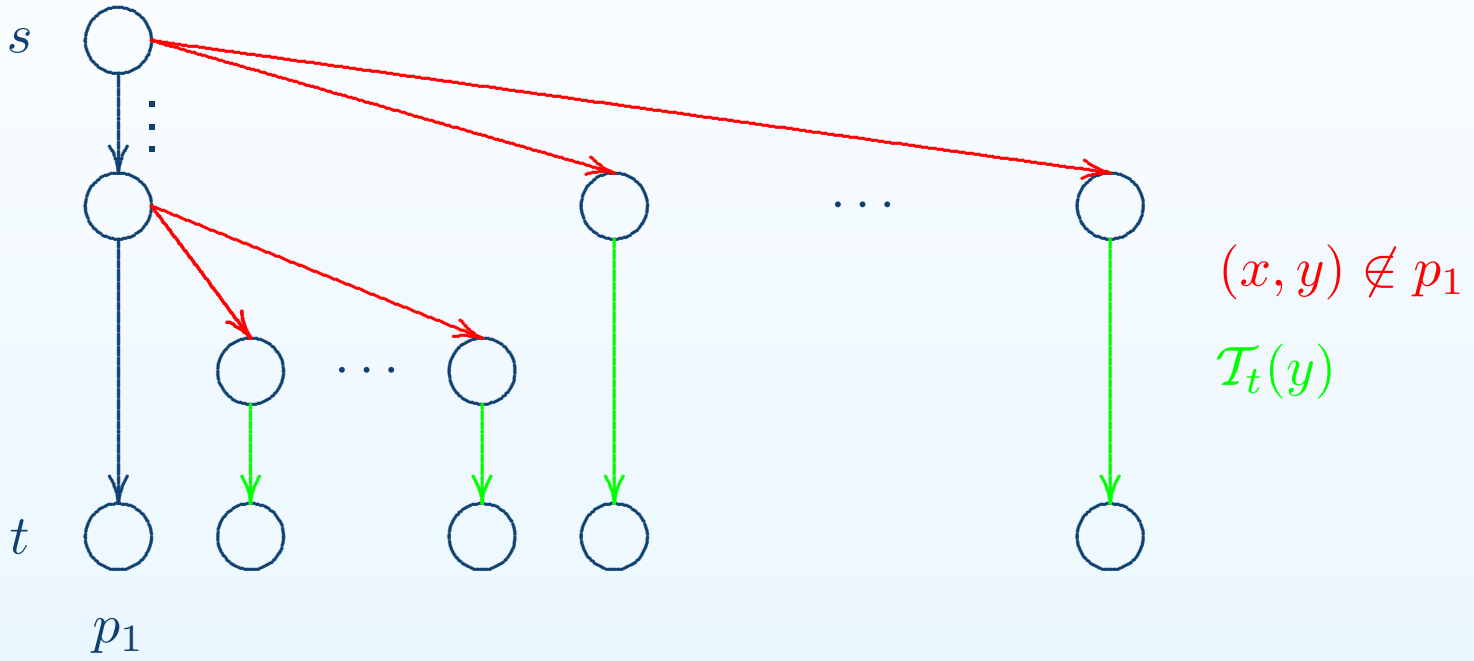
- $\text{sub}_p(s, v_i) \diamond q$,
  onde q é o trajecto com o menor custo de v_i para t, depois de apagado o arco (v_i, v_{i+1}),

  ou

- $\text{sub}_p(s, v_i) \diamond \langle v_i, v \rangle \diamond \mathcal{T}_t(v)$,
  onde $(v_i, v) \neq (v_i, v_{i+1})$, $\mathcal{T}_t$ denota a árvore dos caminhos com o menor custo de todos os nós para t e $\mathcal{T}_t(v)$ o seu subtrajecto de v para t.

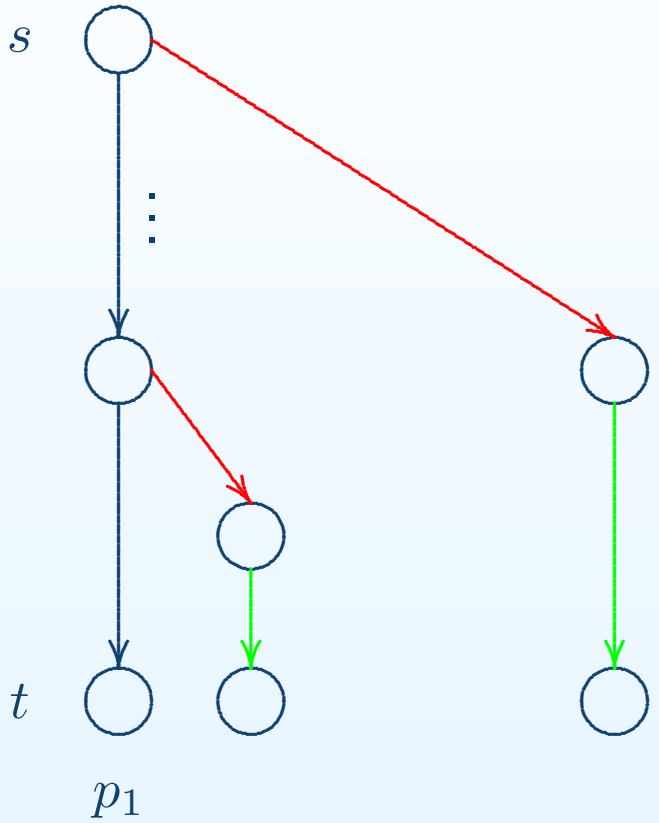
Algoritmos de desvios

Determinar **"quase todos" os desvios possíveis** para cada nó analisado.



Algoritmos de desvios

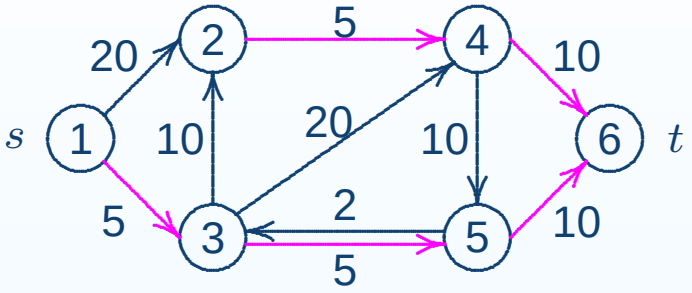
Determinar **apenas o desvio com o menor custo** para cada nó analisado.

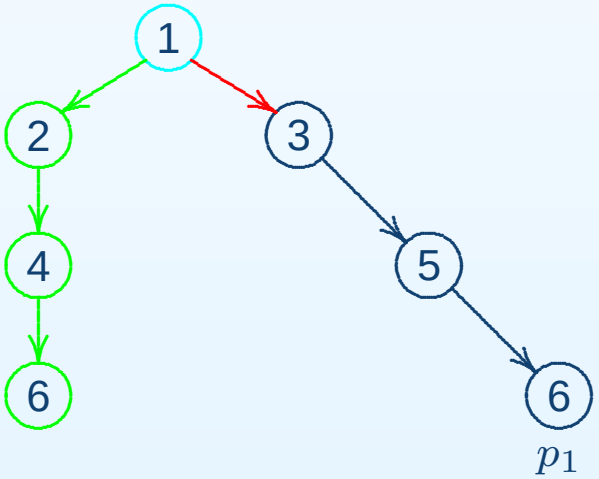


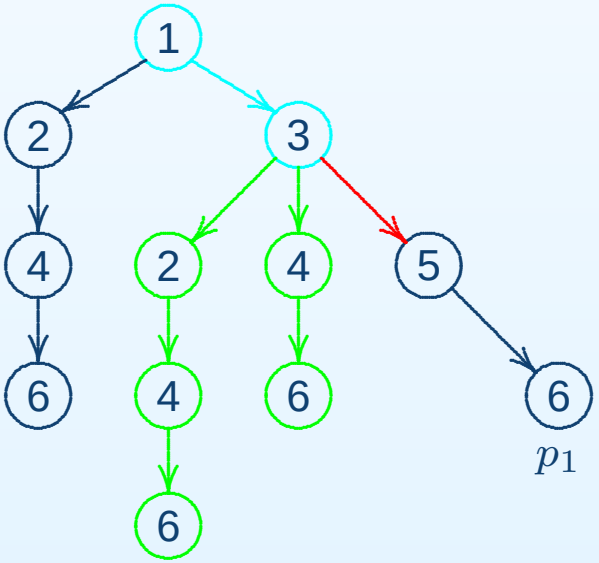
$(x, y) \notin p_1$ e $c_{xy} + c(\mathcal{T}_t(y))$ é mínimo

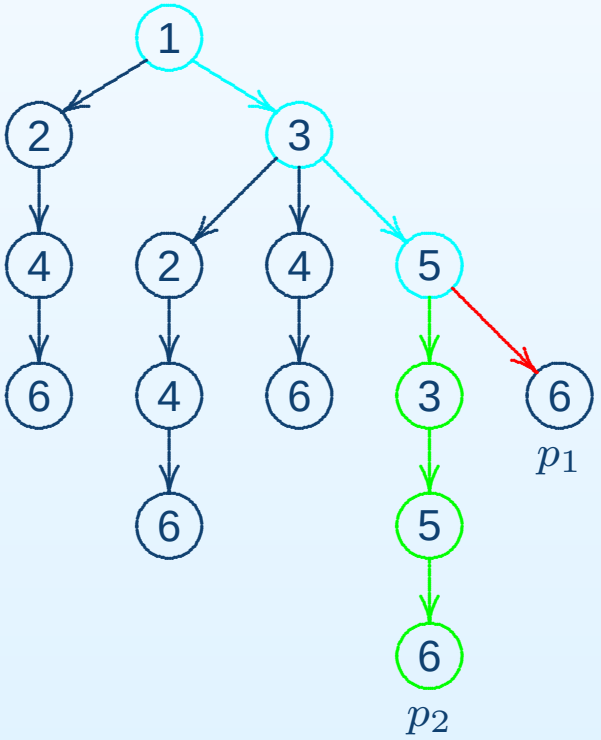
$\mathcal{T}_t(y)$

Algoritmos de desvios (todos os desvios)

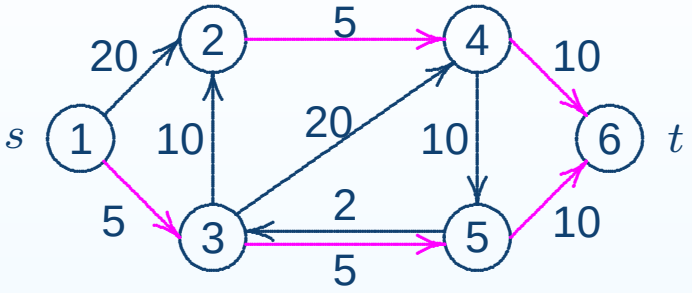


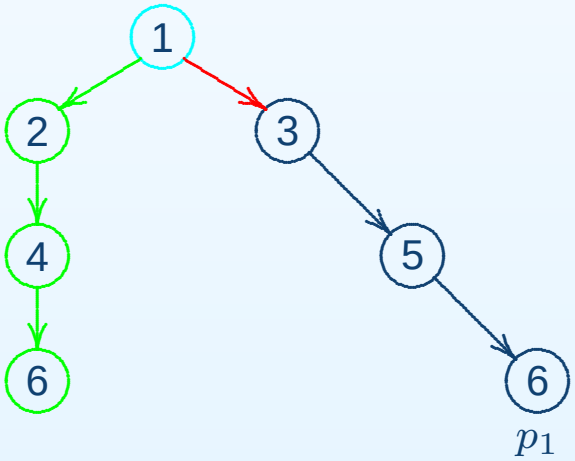


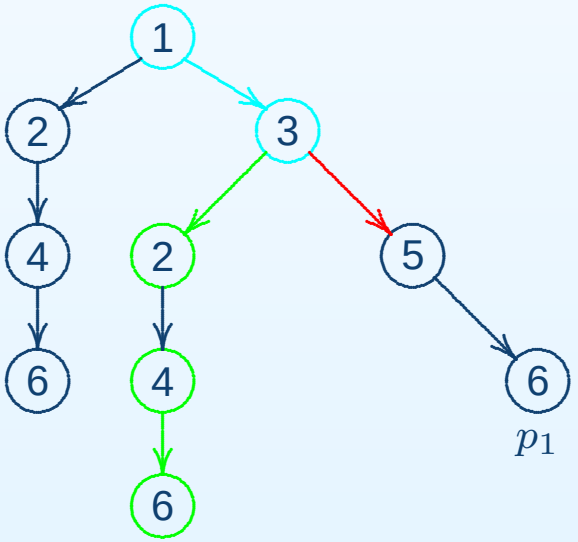


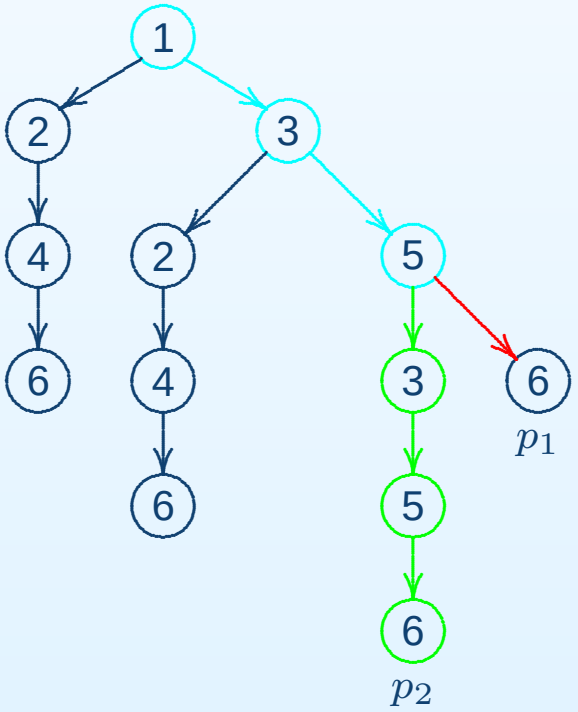


Algoritmos de desvios (apenas um desvio)









Algoritmos de desvios

Custo reduzido de $(i,j) \in \mathcal{A}$ associado a $\mathcal{T}_t \longrightarrow \bar{c}_{ij} = \pi_j - \pi_i + c_{ij}$.
Custo reduzido de $p \in \mathcal{P} \longrightarrow \bar{c}(p) = \sum_p \bar{c}_{ij}$.

Propriedades:

- $c(p) \leq c(q)$ se e só se $\bar{c}(p) \leq \bar{c}(q)$.
- $\bar{c}_{ij} = 0$, para todo $(i,j) \in \mathcal{T}_t$.

Substituir c_{ij} por $\bar{c}_{ij}$ reduz o número de operações executadas:

$$q \in \mathcal{T}_t \Rightarrow \bar{c}(q) = 0,$$

e portanto $\bar{c}(p \diamond q) = \bar{c}(p)$, para todo o trajecto p.

Algoritmos de desvios

Custo reduzido de $(i,j) \in \mathcal{A}$ associado a $\mathcal{T}_t \quad \longrightarrow \bar{c}_{ij} = \pi_j - \pi_i + c_{ij}$.
Custo reduzido de $p \in \mathcal{P} \quad \longrightarrow \quad \bar{c}(p) = \sum_p \bar{c}_{ij}$.

Propriedades:

- $c(p) \leq c(q)$ se e só se $\bar{c}(p) \leq \bar{c}(q)$.
- $\bar{c}_{ij} = 0$, para todo $(i,j) \in \mathcal{T}_t$.

Substituir c_{ij} por $\bar{c}_{ij}$ reduz o número de operações executadas:

$$q \in \mathcal{T}_t \Rightarrow \bar{c}(q) = 0,$$

e portanto $\bar{c}(p \diamond q) = \bar{c}(p)$, para todo o trajecto p.

Arcos com custo reduzido baixo devem ser usados cedo!

Ordenar o conjunto dos arcos emergentes dos nós por ordem não decrescente de custo reduzido.

Utilizar enumeração: Para quê?

Para quê:

- Obter soluções alternativa à óptima.
- Problemas com restrições.
- Problemas multiobjectivo.
- Análise de sensibilidade, incerteza de custos,. . .

Utilizar enumeração: Para quê?

Para quê:

- Obter soluções alternativa à óptima.
- Problemas com restrições.
- Problemas multiobjectivo.
- Análise de sensibilidade, incerteza de custos,. . .

- Utilização em vários problemas e com várias função objectivo.
- A eficiência dos métodos de enumeração é crucial no que respeita a aplicações.

Passeatas *vs.* restrições adicionais

Problemas de trajectos com restrições:

- Trajectos sem nós repetidos (caminhos).
- Trajectos com um número limitado de arcos.
- Trajectos disjuntos.
- . . .

Passeatas *vs.* restrições adicionais

Problemas de trajectos com restrições:

- Trajectos sem nós repetidos (caminhos).
- Trajectos com um número limitado de arcos.
- Trajectos disjuntos.
- . . .

Solução:
Enumerar trajectos ordenamente, até encontrar o 1º que satisfaça as restrições.

Passeatas *vs.* restrições adicionais

Problemas de trajectos com restrições:

- Trajectos sem nós repetidos (caminhos).
- Trajectos com um número limitado de arcos.
- Trajectos disjuntos.
- . . .

Solução:
Enumerar trajectos ordenamente, até encontrar o 1º que satisfaça as restrições.

Contra:
Não sabemos quantos trajectos é necessário listar até encontrar um admissível...

Os K caminhos com o menor custo

Os K caminhos com o menor custo

Dado $K \in \mathbb{N}$ pretende-se encontrar $\mathcal{P}_K = \{p_1, \ldots, p_K\} \subseteq \mathcal{P}$, onde:

- $c(p_k) \leq c(p_{k+1})$, para todo $k \in \{1, \ldots, K-1\}$;
- $c(p_K) \leq c(p)$, para todo o caminho $p \in \mathcal{P} - \mathcal{P}_K$;
- p_k é determinado antes de p_{k+1}, para todo $k \in \{1, \ldots, K-1\}$;
- p_k é um caminho, para todo $k \in \{1, \ldots, K\}$.

Os K caminhos com o menor custo

Dado $K \in \mathbb{N}$ pretende-se encontrar $\mathcal{P}_K = \{p_1, \ldots, p_K\} \subseteq \mathcal{P}$, onde:

- $c(p_k) \leq c(p_{k+1})$, para todo $k \in \{1, \ldots, K-1\}$;
- $c(p_K) \leq c(p)$, para todo o caminho $p \in \mathcal{P} - \mathcal{P}_K$;
- p_k é determinado antes de p_{k+1}, para todo $k \in \{1, \ldots, K-1\}$;
- p_k é um caminho, para todo $k \in \{1, \ldots, K\}$.

Quando $K = 1$ o caminho com o menor custo é também um trajecto com o menor custo (admitindo que a rede não contém ciclos com custo negativo).

Os K caminhos com o menor custo

Dado $K \in \mathbb{N}$ pretende-se encontrar $\mathcal{P}_K = \{p_1, \ldots, p_K\} \subseteq \mathcal{P}$, onde:

- $c(p_k) \leq c(p_{k+1})$, para todo $k \in \{1, \ldots, K-1\}$;
- $c(p_K) \leq c(p)$, para todo o caminho $p \in \mathcal{P} - \mathcal{P}_K$;
- p_k é determinado antes de p_{k+1}, para todo $k \in \{1, \ldots, K-1\}$;
- p_k é um caminho, para todo $k \in \{1, \ldots, K\}$.

Quando $K = 1$ o caminho com o menor custo é também um trajecto com o menor custo (admitindo que a rede não contém ciclos com custo negativo).

No entanto, o k-ésimo trajecto com o menor custo, com $k > 1$, pode não ser um caminho.

Adaptação de algoritmos para enumeração de trajectos

- **Algoritmos de rotulação:**
  Acrescentar teste para evitar rotular nós repetidos?

- **Algoritmos de apagamentos:**
  Nada a fazer!

- **Algoritmos de desvios:**
  Acrescentar teste para evitar escolher arcos que formem ciclos.
  ou. . .
  Modificar método para calcular caminhos ordenadamente!

Algoritmos de desvios

Seja X um conjunto de caminhos candidatos a p_k.

Algoritmo de desvios

Determinar p_1

$X \leftarrow \{p_1\}$

Para $k \in \{2, \ldots, K\}$ **Faz:**

$p_k \leftarrow$ caminho com o menor custo em X

$X \leftarrow X - \{p_k\}$

Analisar nós de p_k e gerar novos caminhos (desvios) que se armazenam em X (candidatos a p_j, com $j > k$)

Desvio de um caminho

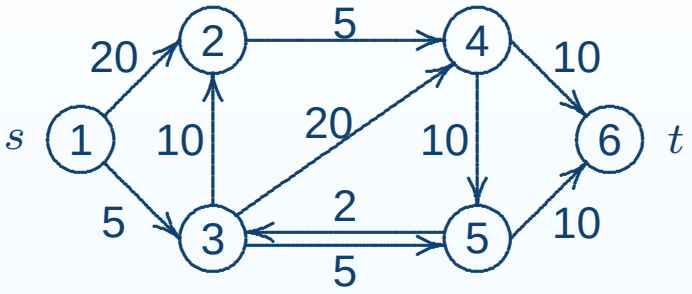
$\text{sub}_p(s, v_i)$ pode conter o nó v ou outros nós de $\mathcal{T}_t(v)$ $\Rightarrow$ $\text{sub}_p(s, v_i) \diamond \langle v_i, v \rangle \diamond \mathcal{T}_t(v)$ pode ser um trajecto com nós repetidos.

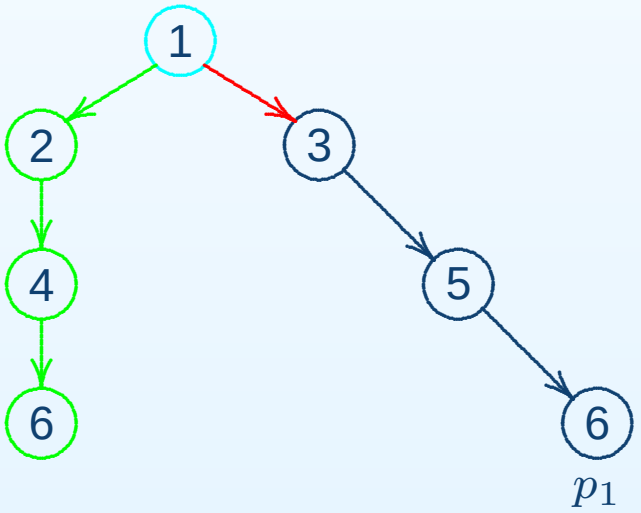
Caminho com o menor custo que desvia de $p = \langle v_1, \ldots, v_{\ell_p} \rangle$ no nó v_i:

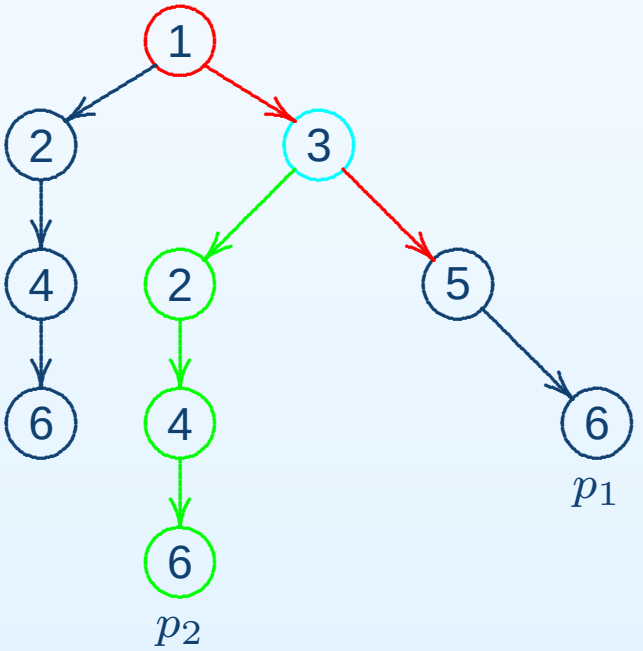
$\text{sub}_p(s, v_i) \diamond q$,
onde q é o caminho com o menor custo de v_i para t, depois de:

- apagado o arco (v_i, v_{i+1}),
- apagados os nós $v_1, \ldots, v_{i-1}$.

Algoritmos de desvios (caminhos)







Complexidade computational

Complexidade computational (pior caso)

Enumeração de trajectos:

- Calcular p_1: $\mathcal{O}(m + n \log n)$
- Substituir c por $\bar{c}$: $\mathcal{O}(m)$
- Ordenar $\mathcal{A}$: $\mathcal{O}(n \log n)$
- A análise de cada nó implica escolher o arco que se segue: $\mathcal{O}(1)$
- Se cada $p_1, \ldots, p_K$ exigir a análise de n nós: $\mathcal{O}(m + n \log n + Kn)$

Enumeração de caminhos:

- A análise de cada nó implica resolver um problema do caminho com o menor custo: $\mathcal{O}(m + n \log n)$
- Se cada $p_1, \ldots, p_K$ exigir a análise de n nós: $\mathcal{O}(Kn(m + n \log n))$

E por fim. . .

- Haverá alguma relação/ponto de contacto entre as 3 classes de métodos para enumeração de trajectos?
- Será possível usar um processo análogo ao algoritmo de desvios para enumeração de caminhos com vista a enumerar soluções de outros problemas combinatórios?
- . . .

Algumas referências

[1] Azevedo, Costa, Madeira e Martins. An algorithm for the ranking of shortest paths. *European Journal of Operational Research*, 69:97–106, 1993.

[2] Azevedo, Madeira, Martins e Pires. A shortest paths ranking algorithm. *Proc. of the Annual Conference AIRO'90, Models and Methods for Decision Support*, páginas 1001–1011. Operational Research Society of Italy, 1990.

[3] Azevedo, Madeira, Martins e Pires. A computational improvement for a shortest paths ranking algorithm. *European Journal of Operational Research*, 73:188–191, 1994.

[4] Bellman. On a routing problem. *Quarterly of Applied Mathematics*, 1:425–447, 1958.

[5] Dijkstra. A note on two problems in connection with graphs. *Numerical Mathematics*, 1:269–271, 1959.

[6] Dreyfus. An appraisal of some shortest-path algorithms. *Operations Research*, 17:395–412, 1969.

[7] Ford. Network flow theory. Relatório Interno P-923, The Rand Corp., Santa Monica, CA, 1956.

[8] Hoffman e Pavley. A method for the solution of the N th best path problem. *Journal of the Association for Computing Machinery*, 6(4):506–514, 1959.

[9] Martins. An algorithm for ranking paths that may contain cycles. *European Journal of Operational Research*, 18:123–130, 1984.

[10] Martins, Pascoal e Santos. A new algorithm for ranking loopless paths. Relatório Interno 97/001, CISUC, 1997.

[11] Martins, Pascoal e Santos. Deviation algorithms for ranking shortest paths. *The International Journal of Foundations of Computer Science*, 10(3):247–263, 1999.

[12] Martins, Pascoal e Santos. Labeling algorithms for ranking shortest paths. Relatório Interno 00/001, CISUC, 2000.

[13] Martins, Pascoal e Santos. A new improvement for a K shortest paths algorithm. *Investigação Operacional*, 21(1):47–60, 2001.

[14] Martins e Santos. A new shortest paths ranking algorithm. *Investigação Operacional*, 20(1):47–62, 2000.

[15] Moore. The shortest path through a maze. *Proceedings of the International Symposium on the Theory of Switching, Part II*, volume 30, páginas 285–292. Harvard University Press, 1959.

[16] Shier. Computational experience with an algorithm for finding the K shortest paths in a network. *Journal of Research of the NBS*, 78:139–164, 1974.

[17] Shier. Iterative methods for determining the k shortest paths in a network. *Networks*, 6:151–159, 1976.

[18] Yen. Finding the K shortest loopless paths in a network. *Management Science*, 17:712–716, 1971.